Università degli studi di Cagliari

Facoltà di Ingegneria

Dipartimento di Ingegneria Elettrica ed Elettronica

Corso di Laurea Specialistica in Ingegneria Elettrica

METODI DIRETTI PER LA RISOLUZIONE DI SISTEMI LINEARI & EQUAZIONI DIFFERENZIALI ORDINARIE

DOCENTE: Prof. Giuseppe Rodriguez

STUDENTE: Sara Pili 34876

Anno Accademico 2005-2006

INDICE

Capitolo 1

METODI DIRETTI PER LA RISOLUZIONE DI SISTEMI LINEARI

Per la risoluzione di sistemi di equazioni lineari, a differenza della quasi totalità dei problemi non lineari, sono disponibili numerosi metodi diretti. Tali metodi trasformano, con un numero finito di passi, un generico sistema lineare in un sistema equivalente, dotato di un struttura che ne rende più semplice la risoluzione.

Un generico sistema di n equazioni lineari in n incognite assume la forma:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &= b_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n &= b_n \end{aligned}$$

ma viene spesso indicato in maniera più compatta utilizzando la notazione matriciale:

$$Ax = b$$

dove $A = (a_{ij})_{i,j=1}^{n}$ è la matrice dei coefficienti, $b = (b_1,\ldots,b_n)^T$ il vettore dei termini noti e $x = (x_1,\ldots,x_n)^T$ la soluzione.

Un importante teorema afferma che un sistema lineare ammette una e una sola soluzione se e solo se è verificata una delle seguenti proprietà equivalenti:

1. il determinante di A è diverso da zero;
2. il rango di A è uguale a n;
3. il sistema omogeneo $Ax = 0$ ammette la sola soluzione banale $(x_i = 0, i = 1,\ldots,n)$.

Per semplicità, i sistemi lineari esaminati saranno sistemi lineari con coefficienti, termini noti e variabili reali.

1.1. Fattorizzazione QR

La fattorizzazione QR permette di trasformare una qualunque matrice A $m \times n$ nella forma:

$$A = QR,$$

dove Q è una matrice ortogonale $m \times m$, cioè tale che $Q^T Q = QQ^T = I$, ed R una matrice triangolare superiore con le stesse dimensioni di A e con stesso numero di condizionamento rispetto alla norma-2 della stessa matrice. Per una matrice quadrata tale fattorizzazione è anche unica.

Nei prossimi paragrafi si studierà la fattorizzazione di una data matrice A mediante diversi algoritmi di calcolo che presentano differenti caratteristiche di stabilità e complessità computazionale. Gli algoritmi implementati sono l'algoritmo di Householder, di Givens e la fattorizzazione QR ottenuta dal codice commerciale "Matlab".

1.1.1. Impostazione del problema

Come primo passo del programma implementato su Matlab si è definita la matrice dei coefficienti A.
In tal caso si è preferito prendere in esame due matrici quadrate di rango *n*: una matrice con elementi casuali (random) e una matrice i quali elementi sono fissati dalla legge $a_{ij} = 1/(i + j - 1)$ (matrice di Hilbert). La matrice di Hilbert è una matrice definita positiva e caratterizzata dalla crescita del condizionamento all'aumentare della dimensione.
In entrambi i casi, mediante l'istruzione ripetitiva *for*, si sono studiate matrici di dimensione crescente.

Matrice Random

Per la costruzione della matrice Random si è definito prima di tutto un vettore colonna *vn* costituito da elementi compresi tra 0 e 100 con passo 10. Successivamente si è implementato un ciclo for con indice di iterazione che va, con step uno, dall'unità alla prima dimensione del vettore *vn*. Tale indice si è preso successivamente come dimensione della matrice A.

Matrice di Hilbert

Per la costruzione della matrice di Hilbert il vettore colonna *vn* si è considerato costituito, invece, da elementi compresi tra 5 e 15 con passo unitario. La scelta di esaminare una matrice di dimensioni tanto inferiori rispetto alla matrice casuale è dovuto ai problemi che la caratterizzano al crescere della dimensione.

Una volta definita la matrice del sistema si sono realizzate delle "*function*" che, ricevendo in ingresso la matrice, restituissero in uscita la matrice ortogonale Q e la matrice triangolare superiore R. Tali funzioni permettono, pertanto, la fattorizzazione della matrice secondo l'algoritmo di Householder e di Givens. Per la fattorizzazione del codice commerciale si è richiamata la funzione già disponibile in esso.

Per ciascun algoritmo si è studiato l'errore tra la matrice ottenuta dal prodotto $Q \times R$ e la matrice A e l'ortogonalità della matrice Q. Per ottenere ciò, a monte del ciclo *for*, si sono inizializzati per ciascuna funzione due vettori riga costituiti da zeri e di dimensioni pari alla prima dimensione del vettore *vn*. Nel primo vettore, ad ogni iterazione del ciclo, si è memorizzata in corrispondenza della riga equivalente all'indice di iterazione l'errore di fattorizzazione calcolato, mentre nel secondo vettore l'errore relativo all'ortogonalità della matrice Q.

Al termine del ciclo *for* si sono visualizzati:

- una tabella contenente nelle colonne i valori dell'errore di fattorizzazione di ciascun algoritmo e nelle righe il numero di iterazione, ossia la dimensione della matrice;
- la rappresentazione semilogaritmica degli stessi valori dell'errore;
- una tabella contenente nelle colonne i valori dell'errore di ortogonalità della matrice Q di ciascun algoritmo e nelle righe la dimensione della matrice;
- la rappresentazione semilogaritmica degli stessi valori dell'errore.

```
% Sperimentazione sulla fattorizzazione QR
vn = [10:10:100]';                 % vettore vn per la costruzione della matrice Random
% vn = [5:1:15]';                  % vettore vn per la costruzione della matrice di Hilbert

errfh = zeros(size(vn,1),1);       % inizializzazione dei vettori errore di fattorizzazione
errfg = zeros(size(vn,1),1);
errfm = zeros(size(vn,1),1);

ortfh = zeros(size(vn,1),1);       % inizializzazione dei vettori errore di ortogonalità
ortfg = zeros(size(vn,1),1);
ortfm = zeros(size(vn,1),1);

for i = 1:size(vn,1)
    n = vn(i)
```

```
    A = rand(n);                      % matrice dei coefficienti
    % A = hilb(n);
    I = eye(n);

    [Q,R]=qrhouse(A);                 % funzione dell'algoritmo di Householder
    errfh(i) = norm(Q*R-A);           % errore di fattorizzazione
    ortfh(i) = norm((Q'*Q)-I);        % errore di ortogonalità della matrice Q

    [Q,R]=qrgivens(A);                % funzione dell'algoritmo di Givens
    errfg(i) = norm(Q*R-A);
    ortfg(i) = norm((Q'*Q)-I);

    [Q,R]=qr(A);                      % fattorizzazione del codice commerciale
    errfm(i) = norm(Q*R-A);
    ortfm(i) = norm((Q'*Q)-I);
end

                  % tabella contenente i valori dell'errore di fattorizzazione

disp('Tabella degli errori:')
disp('dim.matr    Householder    Givens      Matlab')
tabella1=[vn errfh errfg errfm];
fprintf('  %3d       %8.2e    %8.2e    %8.2e\n',tabella1')

                  % rappresentazione semilogaritmica dei valori dell'errore di fattorizzazione

figure(1)
semilogy(vn,errfh,'-rs',vn,errfg,'-gs',vn,errfm,'-ys')
legend('fatt. Householder','fatt. Givens','fatt. matlab')
title('Errore fattorizzazione QR')
xlabel('dimensione matrice')
ylabel('vettore errore')
grid

                  % tabella contenente i valori dell'errore di ortogonalità

disp('Tabella degli errori sull''ortogonalità della matrice Q:')
disp('dim.matr    Householder    Givens      Matlab')
tabella2=[vn ortfh ortfg ortfm];
fprintf('  %3d       %8.2e    %8.2e    %8.2e\n',tabella2')

                  % rappresentazione semilogaritmica dei valori dell'errore di ortogonalità

figure(2)
semilogy(vn,ortfh,'-rs',vn,ortfg,'-gs',vn,ortfm,'-ys')
legend('fatt. Householder','fatt. Givens','fatt. matlab')
title('Ortogonalità matrice Q')
xlabel('dimensione matrice')
ylabel('vettore errore')
grid
```

Fattorizzazione QR di Householder

Il primo passo dell'algoritmo di Householder consiste nello scrivere la matrice dei coefficienti in termini delle sue colonne

$$A = [a_1 \quad a_2 \quad \ldots \quad a_n].$$

La fattorizzazione di questa matrice si ottiene iterando n-1 volte, dove n è la dimensione della matrice, e utilizzando in ognuna la rispettiva matrice elementare di Householder calcolata. Questa matrice è simmetrica, ortogonale e presenta una struttura di questo tipo:

$$H = I - 2ww^T$$

dove $w \in \mathrm{R}^n$, $\|w\| = 1$.

Nello specifico, nella funzione per la fattorizzazione di Householder, pertanto, si è richiamata a sua volta una funzione che ad ogni passo dell'algoritmo costruisse la matrice elementare, tale che:

$$Ha = a - 2ww^T a = a - 2\left(w^T a\right)w = ke_1,$$

dove *a* è un vettore colonna della matrice A, e_1 il primo versore della base canonica di R^n e $k \in \mathrm{R}$.
Al termine dell'algoritmo si è ottenuta la matrice $A^{(n)}$, ossia la matrice triangolare superiore R, e la matrice ortogonale Q.

```
% Funzione dell'algoritmo di Householder

function [Q,A] = qrhouse(A)

n = size(A,1);
Q=eye(n);                              % inizzializzazione della matrice Q

for i=1:n-1
    H = housemat( A(i:n,i), n, i);     % funzione per la costruzione della matrice H
    Q = Q*H;
    A = H*A;
end
A = triu(A);


% Funzione per la costruzione della matrice elementare H

function H = housemat(x,n,i)

sigma=norm(x);                         % norma-2 del vettore colonna della matrice A
if x(1)>0
```

```
    k=-sigma;
else
    k = sigma;
end
betta=sigma*(sigma+abs(x(1)));
x(1) = x(1) - k;
H = eye(n);
H(i:n,i:n) = eye(size(x,1))-(1/betta)*x*x';
```

Fattorizzazione QR di Givens

Nella fattorizzazione QR di Givens si considerano, invece, le matrici elementari di Givens G.
Nello spazio R^m tale matrice G_{ij} (i<j) opera una rotazione che lascia invariato il sottospazio ortogonale al piano definito dai vettori della base canonica e_i ed e_j.
Essa è così fatta:

$$G_{ij} = \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & c & \cdots & s & & \\ & & \vdots & & \vdots & & \\ & & -s & \cdots & c & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix} \begin{array}{l} \\ \\ (riga-i) \\ \\ (riga-j) \\ \\ \\ \end{array}$$

con $c^2 + s^2 = 1$, ossia elementi coseno e seno.

Per annullare selettivamente le componenti sotto la diagonale della matrice è necessaria una rotazione capace di far giacere su un asse (ascissa od ordinata nulla) il punto di coordinate (x_i, x_j), dove x_i ed x_j sono l'elemento appartenente alla diagonale e l'elemento della stessa colonna ma di una riga inferiore. In tal modo si annulla il termine sottostante la diagonale.
Perché ciò si verifichi le espressioni di c ed s sono quindi:

$$c = \frac{x_i}{\sigma} \quad \text{e} \quad s = \frac{x_j}{\sigma}, \quad \text{con} \quad \sigma = \sqrt{x_i^2 + x_j^2}\ .$$

Nella funzione che implementa l'algoritmo di Givens si è introdotta pertanto una seconda funzione che calcolasse i parametri c ed s per quel determinato passo.

Successivamente una volta noti i seguenti valori si sono modificate le rispettive componenti della matrice A e si è costruita la matrice G_{ij}. Dal prodotto delle matrici di Givens G_{ij} si è ottenuta la matrice Q trasposta.

```
% Funzione dell'algoritmo di Givens

function [Q,A] = qrgivens(A)

n = size(A,1);
Q = eye(n);                          % inizializzazione della matrice Q

for k=1:n
    for i=k+1:n
        G = eye(n);                  % inizializzazione della matrice G
        [c,s] = givrot( A(k,k), A(i,k));   % funzione per il calcolo dei parametri di rotazione
        for j=k:n
            t = (c*A(k,j)) + (s*A(i,j));
            A(i,j) = (-s*A(k,j)) + (c*A(i,j));   % modifica delle componenti della matrice A
            A(k,j) = t;
        end
        G(k,k) = c;                  % costruzione della matrice G
        G(k,i) = s;
        G(i,k) = -s;
        G(i,i) = c;
        Q=(G*Q);                     % costruzione della matrice Q trasposta
    end
end
A = triu(A);
Q=Q';


% Funzione per il calcolo dei parametri di rotazione

function [c,s] = givrot( xi, xj)

if xj==0
    c=1;
    s=0;
elseif abs(xj)>abs(xi)
    t = xi/xj;
    z = sqrt(1+t^2);
    s = 1/z;
    c = t*s;
else
    t = xj/xi;
    z = sqrt(1+t^2);
    c = 1/z;
    s = t*c;
end
```

1.1.2. Analisi dei risultati

Una volta fattorizzata la matrice dei coefficienti secondo i tre algoritmi si è fatto un confronto tra i risultati ottenuti. Come previsto si sono raggiunte soluzioni diverse dallo studio di una matrice casuale e di una matrice di Hilbert.

Matrice casuale

Nel caso in cui si è esaminata come matrice A una matrice casuale si è ottenuto che gli errori derivanti dalla fattorizzazione tendono ad aumentare, per qualunque algoritmo, al crescere della dimensione della matrice. Infatti, per una matrice di dimensione 10 l'errore è dell'ordine di 10^{-15} mentre, progressivamente, si raggiunge un ordine di errore di 10^{-14}. Ciò lo si può osservare dalla tabella riportata di seguito (Tab. 1):

ERRORI DI FATTORIZAZIONE			
Dim. matrice	**Householder**	**Givens**	**Matlab**
10	1.57e-015	1.48e-015	3.02e-015
20	3.53e-015	5.73e-015	1.20e-014
30	4.23e-015	9.65e-015	8.81e-015
40	7.18e-015	1.58e-014	1.25e-014
50	1.21e-014	2.84e-014	6.70e-015
60	1.25e-014	2.20e-014	1.48e-014
70	1.94e-014	3.02e-014	1.57e-014
80	1.60e-014	1.14e-013	2.84e-014
90	2.70e-014	3.02e-014	1.85e-014
100	2.09e-014	6.36e-014	2.21e-014

Tabella 1. Errori di fattorizzazione QR su una matrice casuale.

Dalla rappresentazione semilogaritmica dell'andamento dell'errore si è osservato con maggior attenzione il comportamento di ciascun algoritmo. Per matrici di dimensione inferiore risulta più precisa la fattorizzazione ottenuta mediante l'algoritmo di Householder e di Givens, mentre al crescere di questa l'algoritmo di Householder mantiene una buona precisione e quello di Givens peggiora (Fig. 1).

L'algoritmo di Givens, rispetto a quello di Householder, risulta particolarmente conveniente se applicato a matrici sparse. Questo, infatti, è un metodo che consente di ridurre la complessità computazionale in presenza di molti zeri nel triangolo inferiore della matrice da fattorizzare. Conseguentemente si verificano anche meno errori nella fattorizzazione.

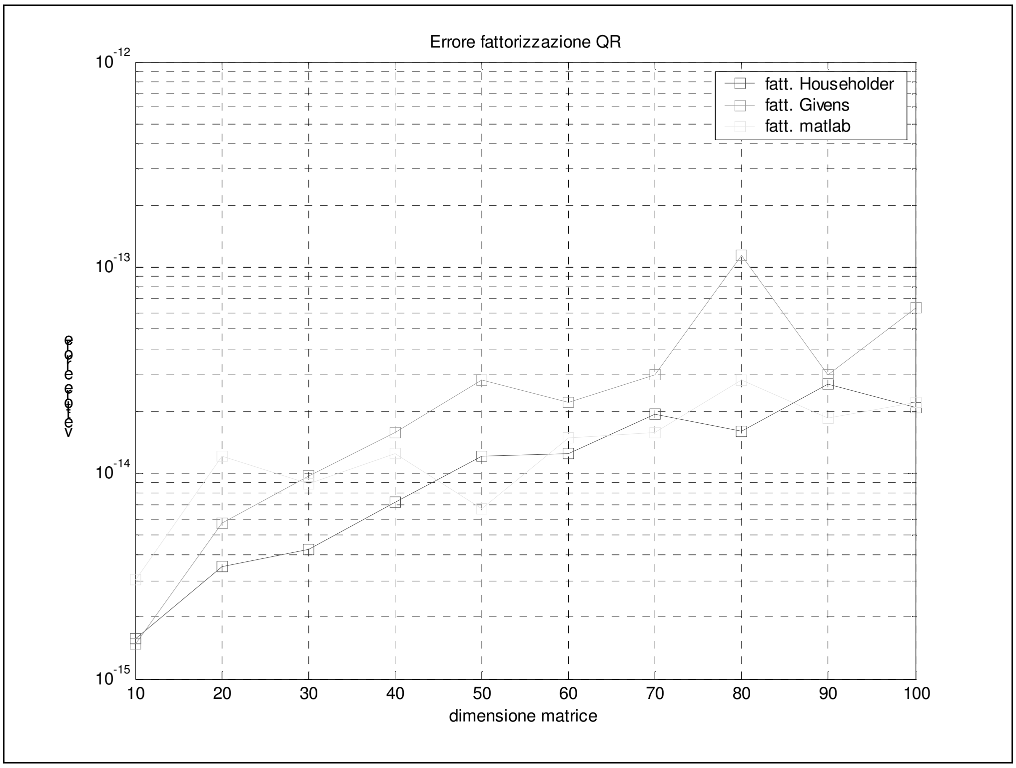

Figura 1. Andamento degli errori di fattorizzazione QR su una matrice casuale.

L'errore nell'ortogonalità della matrice Q, invece, è minore nel caso della fattorizzazione di Matlab, anche se sono minime le differenze con gli altri metodi. L'errore tende ad aumentare al crescere della dimensione della matrice (Tab. 2, Fig. 2).

ERRORI NELL'ORTOGONALITA' DELLA MATR.Q			
Dim. matrice	**Householder**	**Givens**	**Matlab**
10	1.15e-015	1.29e-015	1.48e-015
20	1.80e-015	2.12e-015	1.67e-015
30	2.08e-015	2.05e-015	1.44e-015
40	2.71e-015	2.36e-015	1.82e-015
50	3.59e-015	2.66e-015	1.80e-015
60	4.85e-015	3.27e-015	1.88e-015

70	5.61e-015	3.23e-015	2.25e-015
80	6.98e-015	4.06e-015	2.14e-015
90	5.78e-015	4.20e-015	2.33e-015
100	6.49e-015	4.41e-015	2.37e-015

Tabella 2. Errori nell'ortogonalità della matrice Q su una matrice casuale.

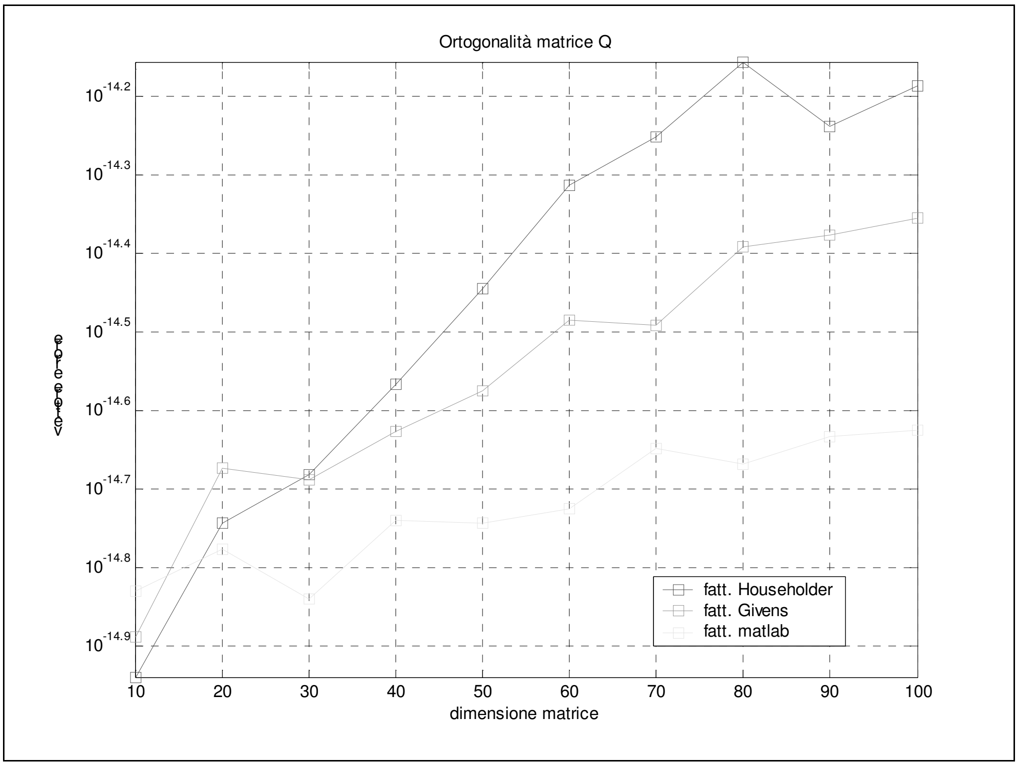

Figura 2. Andamento degli errori nell'ortogonalità della matrice Q su una matrice casuale.

Matrice di Hilbert

Considerata come matrice A la matrice di Hilbert si sono ottenuti risultati piuttosto differenti.
Nello studio dell'errore di fattorizzazione si hanno ordini di grandezza di $10^{-16} \div 10^{-15}$ per dimensioni della matrice A molto inferiori rispetto al caso precedente (Tab. 3). In generale, l'ordine dell'errore per le matrici di Hilbert tende ad aumentare molto velocemente col crescere della dimensione della matrice.

ERRORI DI FATTORIZZAZIONE			
Dim. matrice	**Householder**	**Givens**	**Matlab**
5	1.40e-015	3.10e-016	4.12e-016
6	1.63e-016	3.25e-016	3.77e-016
7	9.50e-016	3.22e-016	6.55e-016
8	5.81e-016	2.79e-016	2.76e-016
9	1.20e-015	2.04e-016	3.77e-016
10	3.85e-016	3.97e-016	1.01e-015
11	1.93e-016	3.41e-016	3.92e-016
12	6.42e-016	3.25e-016	3.67e-016
13	2.34e-016	5.31e-016	5.38e-016
14	5.68e-016	5.58e-016	7.46e-016
15	8.54e-016	1.18e-015	5.98e-016

Tabella 3. Errori di fattorizzazione QR su una matrice di Hilbert.

Dall'andamento dell'errore si è osservato che la fattorizzazione risulta più precisa se eseguita con il metodo di Givens rispetto al codice commerciale (Fig. 3). Nel caso della fattorizzazione di Householder il risultato dipende dalla dimensione della matrice considerata.

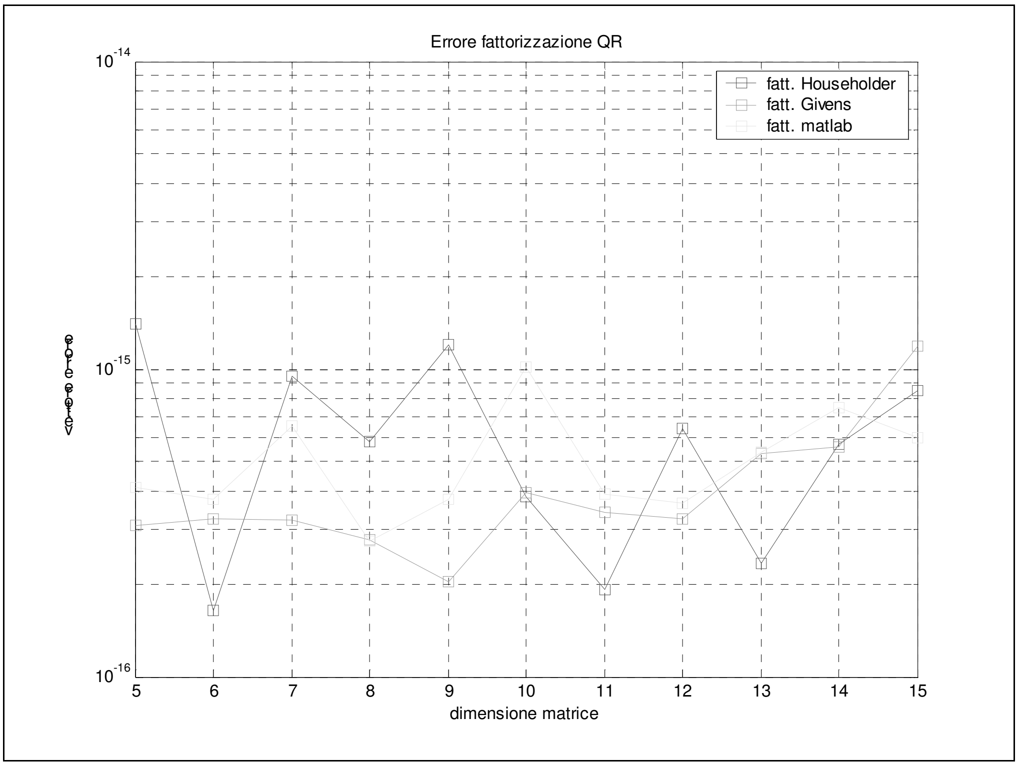

Figura 3. Andamento degli errori di fattorizzazione QR su una matrice di Hilbert

Anche dall'analisi dell'ortogonalità della matrice Q non si distingue un metodo maggiormente preciso perché dipende dalla dimensione della matrice (Tab. 4, Fig. 4).

ERRORI NELL'ORTOGONALITA' DELLA MATR.Q			
Dim. matrice	**Householder**	**Givens**	**Matlab**
5	8.87e-016	5.81e-016	6.44e-016
6	9.73e-016	5.60e-016	7.46e-016
7	1.43e-015	5.92e-016	7.12e-016
8	5.34e-016	6.59e-016	5.58e-016
9	7.65e-016	6.19e-016	9.45e-016
10	1.04e-015	7.48e-016	1.00e-015
11	7.28e-016	8.00e-016	8.09e-016
12	1.13e-015	8.80e-016	1.08e-015
13	1.33e-015	1.04e-015	1.43e-015
14	1.21e-015	1.21e-015	1.03e-015
15	1.48e-015	1.08e-015	8.16e-016

Tabella 4. Errori nell'ortogonalità della matrice Q su una matrice di Hilbert.

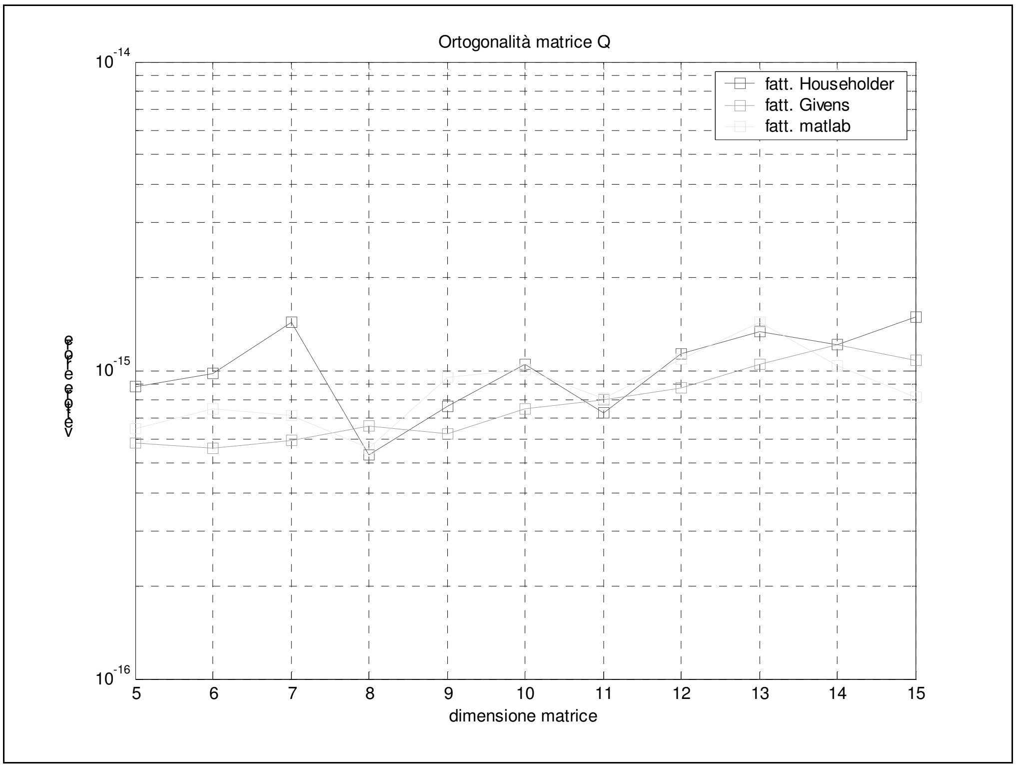

Figura 4. Andamento degli errori nell'ortogonalità della matrice Q su una matrice di Hilbert.

1.2. Risoluzione di sistemi lineari

Considerato il sistema lineare

$$Ax = b$$

se la matrice dei coefficienti A è una matrice quadrata non singolare, la fattorizzazione QR può essere utilizzata per la risoluzione del sistema.

Sostituendo alla matrice A la sua fattorizzazione QR, si ottengono infatti i due sistemi

$$\begin{aligned} Qy &= b \\ Rx &= y \end{aligned}$$

la cui soluzione è

$$x = R \setminus \left(Q^T b\right).$$

La fattorizzazione QR, quindi, trasforma un sistema lineare in un sistema triangolare ad esso equivalente di più facile risoluzione, il quale è dotato dello stesso numero di condizionamento.

Allo stesso modo della fattorizzazione QR, anche il metodo di Gauss consente di trasformare un generico sistema lineare in un sistema triangolare superiore, che può essere risolto con l'algoritmo di sostituzione all'indietro.

Nei prossimi paragrafi si studierà la risoluzione di sistemi lineari sia utilizzando le matrici Q – R, ottenute con i tre algoritmi di fattorizzazione QR esaminati in precedenza, sia secondo l'algoritmo di Gauss.

1.2.1. Impostazione del problema

Per la risoluzione del sistema lineare si sono aggiunte semplicemente alcune righe di programma al programma già realizzato in precedenza. Oltre ai tre algoritmi già implementati si è aggiunto l'algoritmo di Gauss e per tutti e quattro si sono costruite delle funzioni che calcolassero la soluzione del problema.
Una volta note le soluzioni si è verificata la precisione di ciascun metodo confrontando queste con la soluzione reale.

Successivamente si sono visualizzati:

- una tabella contenente nelle colonne i valori dell'errore nella soluzione secondo i quattro algoritmi e nelle righe la dimensione della matrice;
- la rappresentazione semilogaritmica degli stessi valori dell'errore.

1.2.2. Analisi dei risultati

Anche in questo caso si è studiato il problema considerando matrici casuali e matrici di Hilbert di diverse dimensioni.

Matrice casuale

Studiando una matrice casuale, l'errore tra la soluzione calcolata e la soluzione reale tende ad aumentare al crescere della dimensione della matrice. Da un ordine di 10^{-13} per una matrice di dimensione 10 si arriva a un ordine di errore di 10^{-11}. Come dimostra la tabella sottostante e la rappresentazione semilogaritmica, la soluzione più precisa si ottiene con l'algoritmo di Gauss, che per qualunque dimensione commette un errore di circa un ordine di grandezza inferiore rispetto agli altri metodi (Tab. 5). Con la fattorizzazione LU, al contrario, l'errore del sistema finale può aumentare in maniera significativa, ciò rende preferibile l'uso della fattorizzazione QR nella risoluzione di sistemi lineari malcondizionati.

ERRORI NELLA SOLUZIONE				
Dim. matrice	**Householder**	**Givens**	**Matlab**	**Gauss**
10	1.37e-013	7.74e-014	8.64e-014	6.91e-015
20	9.83e-014	7.38e-014	6.91e-014	1.77e-014
30	1.75e-013	9.12e-014	1.11e-013	7.67e-014
40	6.07e-013	3.28e-013	2.56e-013	1.42e-013
50	7.10e-012	1.77e-012	2.67e-012	1.51e-012
60	6.07e-013	4.62e-013	2.90e-013	1.74e-013
70	1.37e-011	5.37e-012	3.14e-012	1.34e-012
80	1.40e-011	6.41e-012	1.32e-011	4.28e-012
90	3.67e-012	1.22e-012	1.03e-012	2.22e-013
100	1.13e-011	3.54e-012	5.31e-012	2.39e-012

Tabella 5. Errori nella soluzione su una matrice casuale.

Graficamente si deduce che la soluzione meno esatta si ottiene dall'algoritmo di Householder, mentre la fattorizzazione QR del codice commerciale e Givens sono di pari precisione (Fig. 5).

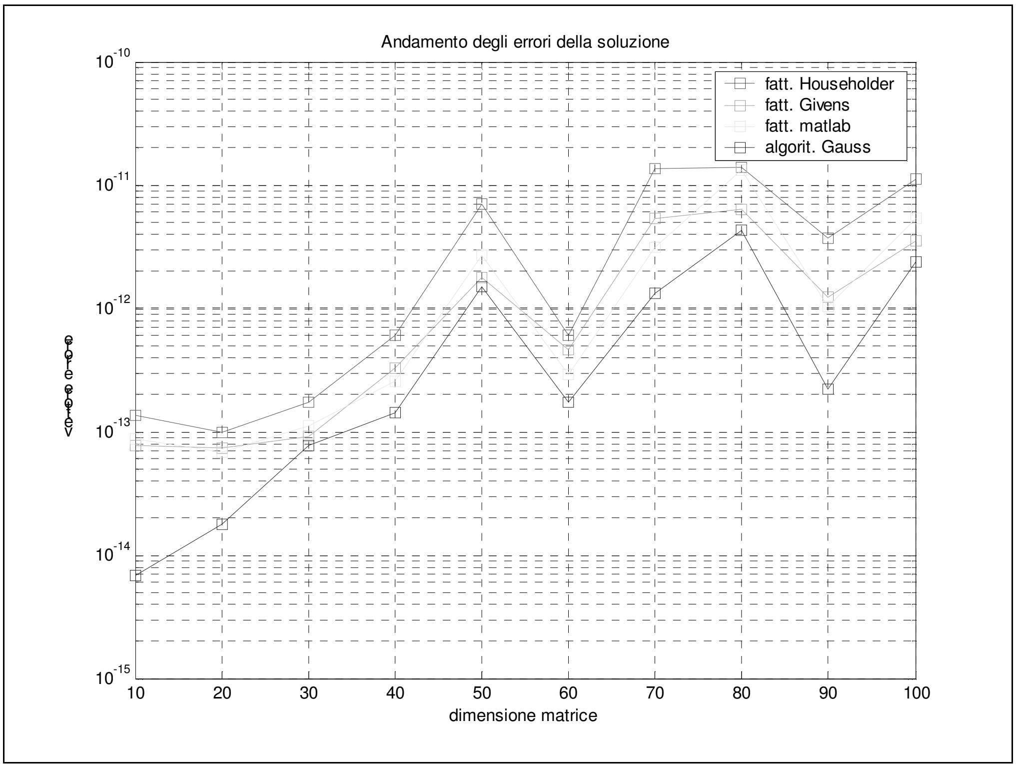

Figura 5. Andamento degli errori nella soluzione su una matrice casuale.

Matrice di Hilbert

Nel caso di una matrice di Hilbert la precisione della soluzione è nettamente inferiore. Da un ordine di errore di 10^{-12} per una matrice di dimensione 5 si arriva a un ordine di errore di 10^{5}, dubitando anche dell'attendibilità di questi risultati (Tab. 6).

Ciò, come anticipato precedentemente, deriva dalle caratteristiche della matrice, nella quale aumentando la dimensione peggiora velocemente il condizionamento.

ERRORI NELLA SOLUZIONE				
Dim. matrice	**Householder**	**Givens**	**Matlab**	**Gauss**
5	3.64e-011	4.20e-011	6.28e-011	1.96e-012
6	2.26e-009	3.05e-009	1.47e-009	6.11e-010
7	6.59e-008	1.43e-007	9.28e-008	3.52e-008
8	5.08e-006	6.84e-006	7.36e-006	6.55e-007
9	1.87e-004	1.56e-004	3.94e-004	3.19e-005
10	4.01e-003	7.31e-003	8.83e-003	7.23e-004
11	5.93e-001	8.52e-001	8.23e-001	1.75e-002
12	1.31e+001	8.80e+000	3.20e+001	5.82e-001
13	4.17e+002	3.80e+003	9.96e+002	6.79e+000
14	4.26e+003	3.54e+003	8.93e+002	5.02e+001
15	1.26e+003	1.50e+005	2.15e+004	1.19e+001

Tabella 6. Errori nella soluzione su una matrice di Hilbert.

Come nel caso di una matrice casuale, il metodo di risoluzione più preciso è l'algoritmo di Gauss, attraverso cui si ha un'accuratezza circa un ordine di grandezza superiore (Fig. 6).

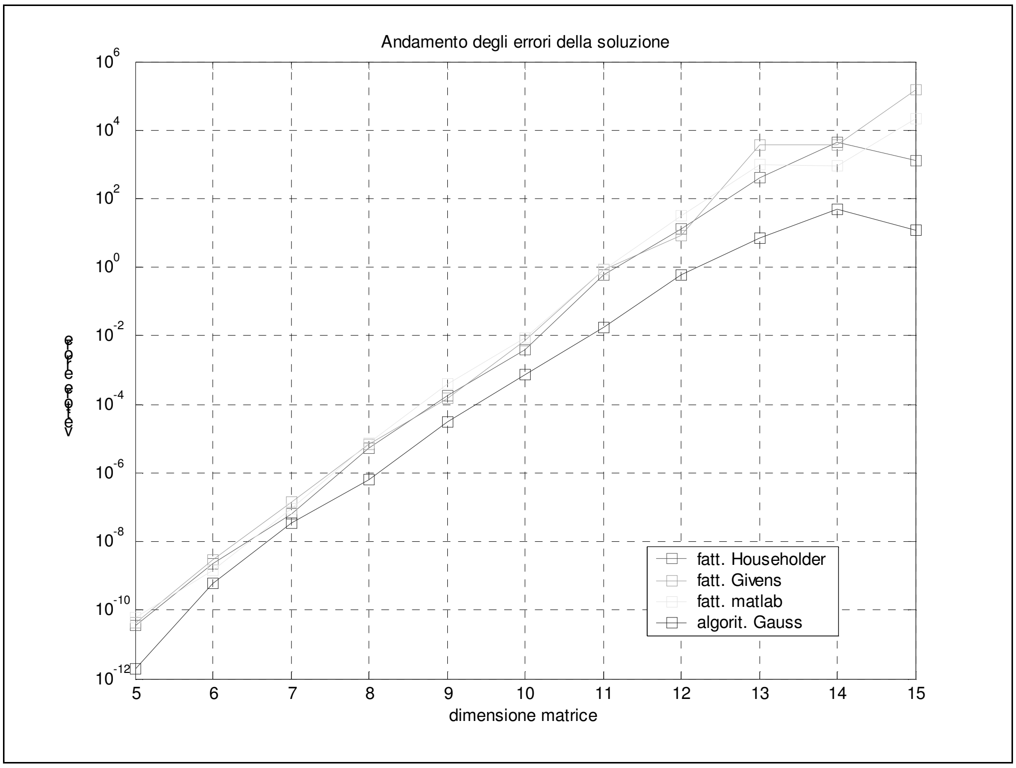

Figura 6. Andamento degli errori nella soluzione su una matrice di Hilbert.

Capitolo 2

EQUAZIONI DIFFERENZIALI ORDINARIE

Le equazioni differenziali rappresentano "un importante strumento matematico" che si impiega in svariati campi applicativi, dalla meccanica, all'astronomia, alle scienze economiche, etc.

Le equazioni differenziali sono connesse a un particolare tipo di problema chiamato "problema di Cauchy". Questo problema è comunemente espresso in questa forma:

$$\begin{aligned} y'(x) &= f(x, y(x)) \\ y(x_o) &= y_o \end{aligned}$$

dove

$$\begin{aligned} f \ &: \ I_o \times \mathrm{R} \to \mathrm{R} \\ (x, y) &\to f(x, y) \end{aligned}$$

e di soluzione y:

$$\begin{aligned} y \ &: \ I_o \to \mathrm{R} \\ x &\to y(x). \end{aligned}$$

La soluzione al problema è di classe $C^1(I_o)$ ed è tale che la sua derivata prima coincide con la funzione $f(x, y(x))$ in ciascun punto dell'intervallo I_o e assume il valore y_o nel punto iniziale x_o.
Il punto iniziale (x_o, y_o) consente di selezionare la soluzione tra le infinite curve. La derivata, invece, esprime semplicemente l'andamento locale della generica curva integrale.

Mediante le formule alle differenze finite è possibile determinare una soluzione discreta del problema, approssimando numericamente la soluzione su un insieme discreto di punti

$$\eta_j \cong y(x_j) \ , \ j = 0,1,2,...$$

I valori η_j possono essere ottenuti approssimando in vario modo la derivata della soluzione $y(x)$, esistono infatti diverse formule che approssimano in modo più o meno preciso.

Nei prossimi paragrafi si studieranno alcuni metodi per la risoluzione dei problemi di Cauchy e si confronteranno le soluzioni ottenute.

2.1. Impostazione del problema

Come primo passo per l'implementazione del programma si sono definite le equazioni differenziali che si aveva interesse di risolvere, indicando la funzione, il punto iniziale, il punto finale, il numero di iterazioni e la soluzione reale del problema. Nello specifico si sono prese in esame due equazioni.

Successivamente si sono realizzate delle funzioni che calcolassero la soluzione discreta secondo diversi metodi e si sono confrontate le soluzioni.

In particolare si è realizzato:

- un primo caso in cui si sono confrontati tre metodi monostep espliciti;
- un secondo caso in cui si sono confrontati un metodo del primo ordine, uno del secondo, uno del terzo e uno del quarto;
- un ultimo caso in cui si è fatto un confronto tra due metodi del quarto ordine, di cui uno esplicito e uno implicito.

2.2. Confronto tra metodi monostep espliciti

2.2.1 Impostazione del problema

Per metodi monostep si intendono quei metodi in cui la valutazione di ogni punto discreto η_{j+1} coinvolge solo il punto η_j e non le approssimazioni precedenti $\eta_{j-1}, \eta_{j-2}, \ldots$ Per metodi espliciti, invece, si intendono tutti quei metodi dove il valore da calcolare ad ogni passo η_{j+1} compare solo a sinistra dell'uguale, e non tra gli argomenti della funzione *f*.

Tra i metodi monostep espliciti si sono esaminati il metodo di Eulero-Cauchy, di Heun e di Eulero modificato.

La formula di Eulero-Cauchy è il metodo di risoluzione più semplice e consiste nell'approssimare la derivata nel punto x_j col rapporto incrementale in avanti

$$f(x_j, y(x_j)) \cong \frac{y_{j+1} - y_j}{h}.$$

Sostituendo i valori incogniti $y_j = y(x_j)$ con le approssimazioni η_j, si ottiene lo schema numerico

$$\begin{array}{l} \eta_{j+1} = \eta_j + hf(x_j, \eta_j) \\ \eta_o = y_o \end{array}.$$

Il metodo di Heun, invece, si ottiene rendendo esplicita la formula di Crank-Nicholson. Questo si esprime nella forma:

$$\begin{array}{l} \eta_{j+1} = \eta_j + \frac{h}{2}\left[f(x_j, \eta_j) + f(x_{j+1}, \eta_j + hf(x_j, \eta_j))\right] \\ \eta_o = y_o \end{array}.$$

Per ultimo, il metodo di Eulero modificato:

$$\begin{array}{l} \eta_{j+1} = \eta_j + hf\left(x_j + \frac{h}{2}, \eta_j + \frac{h}{2} f(x_j, \eta_j)\right) \\ \eta_o = y_o \end{array}.$$

Per ciascun metodo si è realizzata una funzione dove si sono implementate le formule di risoluzione scritte sopra. Dopo di che, noti i valori di x e y di ciascun metodo, si sono visualizzati:

- la soluzione del problema, confrontando quella reale con le soluzioni ottenute con i tre metodi;
- una tabella contenete gli errori nella soluzione;
- l'andamento dell'errore.

```
% Risoluzione con tre metodi monostep espliciti: Eulero-Cauchy, Heun ed Eulero modificato

% funz = @test1;                  % problema di Cauchy in esame
                                  % funzione richiamante l'equazione differenziale test
% x0 = -2;
% xf = 2;
% soluz = @soluz1;                % funzione richiamante la soluz. dell'equazione test

funz = @test2;
```

```
x0 = 0;
xf = 10;
soluz = @soluz2;

n = input('n = ');                          % numero di iterazioni
y0 = feval(soluz,x0);

[x,y]=euler_espl(funz,x0,y0,xf,n);          % funzione del metodo di Eulero-Cauchy eplicito
[x,y1]=heun(funz,x0,y0,xf,n);               % funzione del metodo di Heun
[x,y2]=euler_modif(funz,x0,y0,xf,n);        % funzione del metodo di Eulero modificato

                    % Visualizzazione andamento della curva integrale

figure(1)
s = feval(soluz,x);
plot(x,s,'-r',x,y,'-g',x,y1,'-y',x,y2,'-m')
legend('soluzione','Eulero-Cauchy','Heun','Eulero modificato')
title('Risoluzione del problema di Cauchy con metodi monostep espliciti')
xlabel('punti valutati')
ylabel('soluzione')
grid

                    % Tabella contenete gli errori nella soluzione

disp('Tabella degli errori:')
disp('p. valutati  Eulero-Cauchy     Heun      Eulero modif.')
tabella=[x abs(s-y) abs(s-y1) abs(s-y2)];
fprintf('  %6.1f      %8.2e     %8.2e     %8.2e\n',tabella')

                    % Visualizzazione andamento degli errori nella soluzione

figure(2)
plot(x,abs(s-y),'-g',x,abs(s-y1),'-y',x,abs(s-y2),'-m')
legend('Eulero-Cauchy','Heun','Eulero modificato')
title('Andamento dell''errore')
xlabel('punti valutati')
ylabel('vettore errore')
grid


% funzione richiamante l'equazione differenziale test1

%  ODE, soluzione: y=exp(-x.^2/2), y(-2)=exp(2), x in [-2,2]
%  dim = 1
function yp = f( x, y)
yp = -x .* y;


% funzione richiamante la soluz. dell'equazione test1

function y = soluz(x)
y=exp(-x.^2/2);
```

% funzione richiamante l'equazione differenziale test2

```
%  ODE, soluzione: y=atan(exp(x)), y(0)=atan(1), x in [0,10]
%  dim = 1
function yp = f( x, y)
yp = exp(x) .* cos(y) .* cos(y);
```

% funzione richiamante la soluz. dell'equazione test2

```
function y = soluz(x)
y=atan(exp(x));
```

% funzione del metodo di Eulero-Cauchy esplicito

```
function [x,y]=euler_espl(funz,x0,y0,xf,n)

h = (xf - x0)/ n;                    % calcola il passo h
x = [x0; zeros(n,1)];                % inizializza i vettori colonna x e y
y = [y0; zeros(n,1)];

for i=1:n                            % calcola i vettori x e y
  x(i+1) = x(i)+h;
  y(i+1) = y(i)+h*feval(funz,x(i),y(i));
end
```

% funzione del metodo di Heun

```
function [x,y]=heun(funz,x0,y0,xf,n)

h = (xf - x0)/ n;
x = [x0; zeros(n,1)];
y = [y0; zeros(n,1)];

for i=1:n
  x(i+1) = x(i)+h;
  k1 = feval(funz,x(i),y(i));
  k2 = feval(funz,x(i+1),y(i)+h*k1);
  y(i+1) = y(i) + h/2*[k1+k2];
end
```

% funzione del metodo di Eulero modificato

```
function [x,y]=euler_modif(funz,x0,y0,xf,n)

h = (xf - x0)/ n;
x = [x0; zeros(n,1)];
y = [y0; zeros(n,1)];

for i=1:n
  x(i+1) = x(i)+h;
```

```
    r1 = x(i) + h/2;
    r2 = y(i) + [h/2 * feval(funz,x(i),y(i))];
    y(i+1) = y(i) + h*feval(funz,r1,r2);
  end
```

2.2.2. Analisi dei risultati

Confrontando le soluzioni ottenute con i tre metodi si osserva che il metodo che meglio approssima la soluzione è il metodo di Eulero modificato (Fig. 7). Tale considerazione deriva dal fatto che, così come il metodo di Heun, il metodo di Eulero modificato richiede ad ogni passo due valutazioni della funzione f. Ciò aumenta la complessità computazionale, ma permette di ottenere una migliore approssimazione. La soluzione del metodo di Eulero-Cauchy, invece, è quella meno precisa.

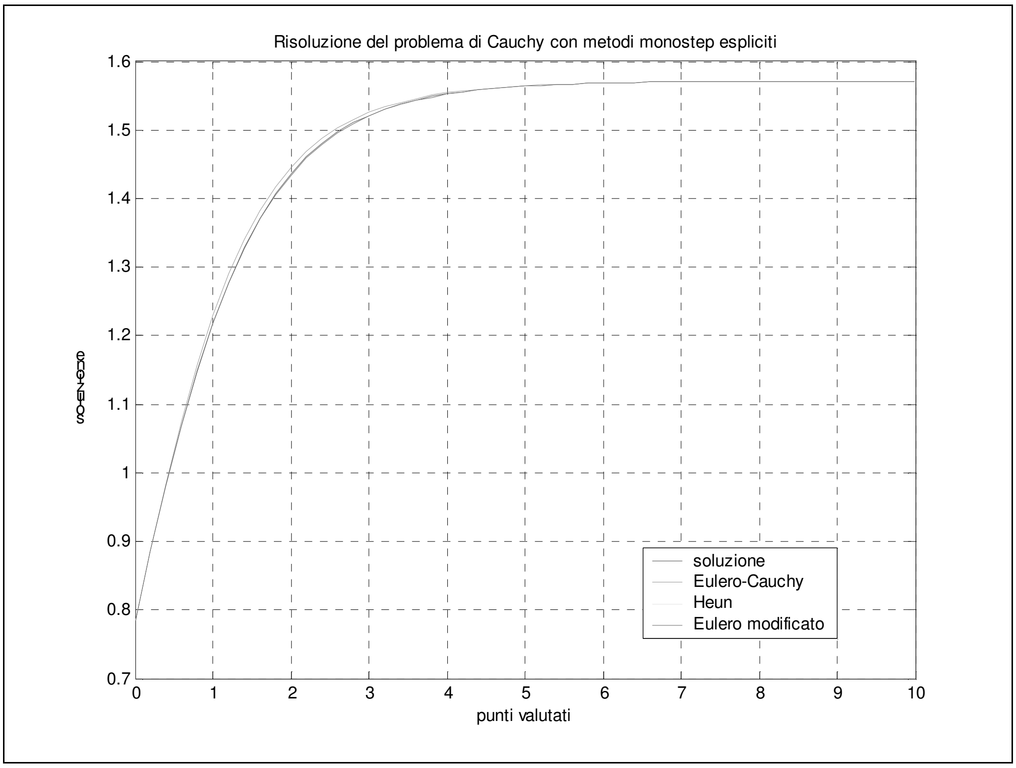

Figura 7. Risoluzione del problema di Cauchy con metodi monostep espliciti.

Per rende più visibile l'andamento delle curve si è fatto un ingrandimento delle soluzioni del problema in un generico intervallo (Fig. 8).

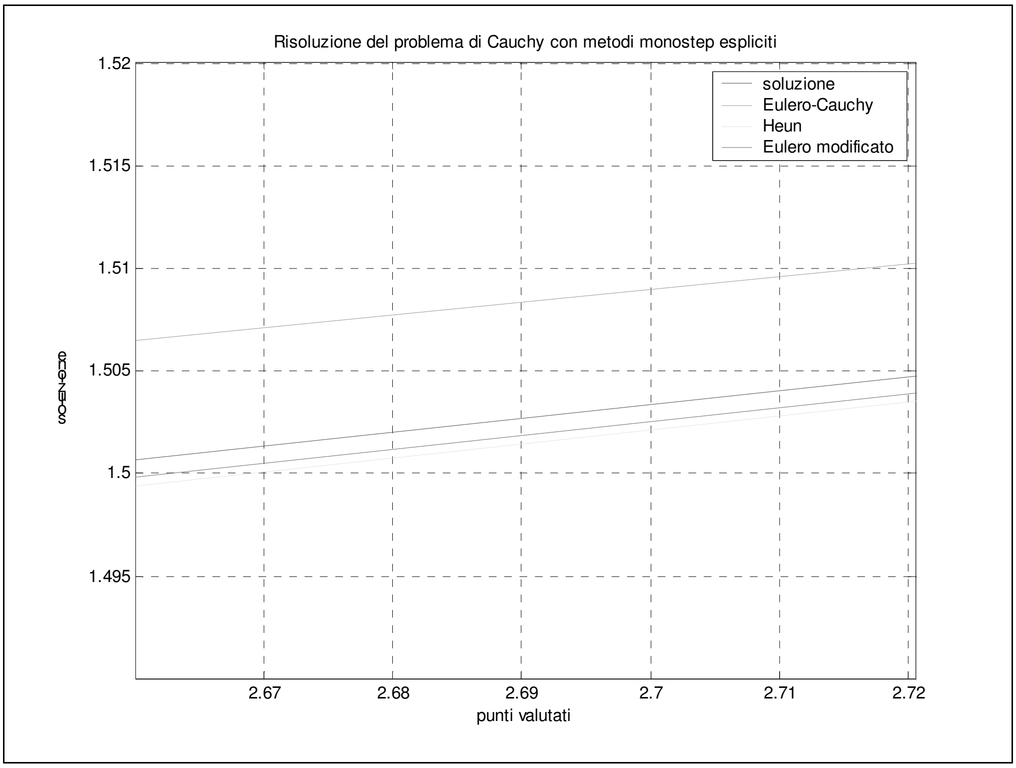

Figura 8. Ingrandimento della risoluzione del problema di Cauchy con metodi monostep espliciti.

Successivamente si è valutato per ogni metodo l'errore nella risoluzione del problema.
Su una tabella e una rappresentazione grafica si è rappresentata la differenza tra la soluzione reale e quella calcolata (Tab. 7, Fig. 9).
Come ci si aspettava la curva con errore inferiore è quella relativa al metodo di Eulero modificato.

ERRORI NELLA SOLUZIONE			
P. valutati	**Eul.-Cauchy**	**Heun**	**Eulero modif.**
0.0	0.00e+000	0.00e+000	0.00e+000
0.2	6.60e-004	4.03e-004	1.44e-004
0.4	3.00e-003	9.21e-004	9.95e-005
0.6	6.08e-003	1.46e-003	1.13e-004
0.8	8.94e-003	1.92e-003	4.18e-004
1.0	1.10e-002	2.25e-003	7.29e-004
1.2	1.20e-002	2.41e-003	9.81e-004

1.4	1.21e-002	2.44e-003	1.15e-003
1.6	1.16e-002	2.35e-003	1.22e-003
1.8	1.07e-002	2.18e-003	1.22e-003
2.0	9.52e-003	1.98e-003	1.17e-003
2.2	8.33e-003	1.76e-003	1.08e-003
2.4	7.17e-003	1.53e-003	9.71e-004
2.6	6.10e-003	1.32e-003	8.58e-004
2.8	5.14e-003	1.13e-003	7.47e-004
3.0	4.31e-003	9.56e-004	6.42e-004
3.2	3.59e-003	8.05e-004	5.47e-004
3.4	2.98e-003	6.74e-004	4.63e-004
3.6	2.47e-003	5.62e-004	3.89e-004
3.8	2.04e-003	4.67e-004	3.26e-004
4.0	1.68e-003	3.87e-004	2.71e-004
4.2	1.38e-003	3.20e-004	2.25e-004
4.4	1.13e-003	2.64e-004	1.87e-004
4.6	9.31e-004	2.18e-004	1.54e-004
4.8	7.64e-004	1.79e-004	1.27e-004
5.0	6.27e-004	1.48e-004	1.05e-004
5.2	5.14e-004	1.21e-004	8.65e-005
5.4	4.21e-004	9.97e-005	7.11e-005
5.6	3.45e-004	8.18e-005	5.84e-005
5.8	2.83e-004	6.71e-005	4.80e-005
6.0	2.32e-004	5.51e-005	3.94e-005
6.2	1.90e-004	4.51e-005	3.23e-005
6.4	1.55e-004	3.70e-005	2.65e-005
6.6	1.27e-004	3.03e-005	2.17e-005
6.8	1.04e-004	2.49e-005	1.78e-005
7.0	8.54e-005	2.04e-005	1.46e-005
7.2	6.99e-005	1.67e-005	1.20e-005
7.4	5.72e-005	1.37e-005	9.81e-006
7.6	4.69e-005	1.12e-005	8.03e-006

7.8	3.84e-005	9.17e-006	6.58e-006
8.0	3.14e-005	7.51e-006	5.39e-006
8.2	2.57e-005	6.15e-006	4.41e-006
8.4	2.11e-005	5.03e-006	3.61e-006
8.6	1.72e-005	4.12e-006	2.96e-006
8.8	1.41e-005	3.38e-006	2.42e-006
9.0	1.16e-005	2.76e-006	1.98e-006
9.2	9.46e-006	2.26e-006	1.63e-006
9.4	7.75e-006	1.85e-006	1.33e-006
9.6	6.34e-006	1.52e-006	1.09e-006
9.8	5.19e-006	1.24e-006	8.92e-007
10.0	4.25e-006	1.02e-006	7.31e-007

Tabella 7. Errore nella soluzione ottenuta con metodi monostep espliciti.

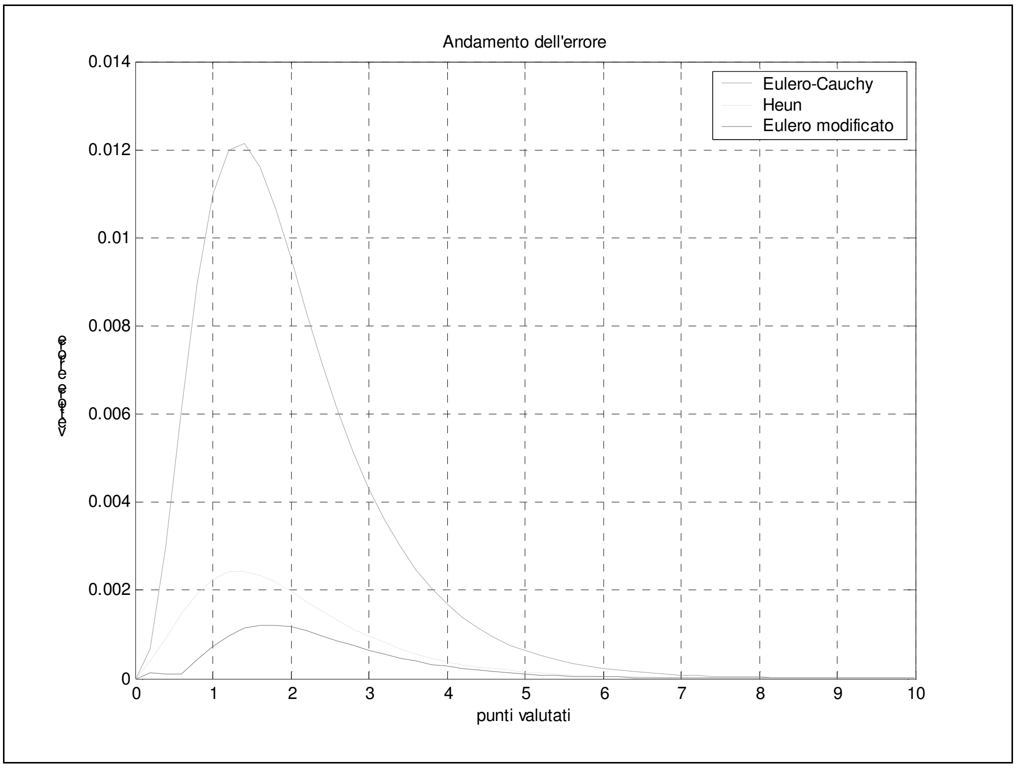

Figura 9. Andamento dell'errore nella soluzione con metodi monostep espliciti.

2.3. Confronto tra metodi del primo, secondo, terzo e quarto ordine

2.3.1 Impostazione del problema

Dai risultati sui metodi monostep espliciti si è osservato che mediante i metodi di secondo ordine Eulero modificato e Heun si ottenere una soluzione più precisa rispetto al metodo di primo ordine di Eulero-Cauchy.
Con opportune tecniche è possibile ricavare formule di ordine superiore, quali ad esempio la formule di Runge-Kutta del terzo e del quarto ordine.

In questa seconda fase si sono confrontati i metodi alle differenze finite di primo, secondo, terzo e quarto ordine. Nello specifico il metodo di Eulero-Cauchy, di Heun, ERK3 e ERK4.

```
% funzione del metodo di Runge-Kutta di ordine 3

function [x,y]=erk3(funz,x0,y0,xf,n)

h = (xf - x0)/ n;
x = [x0; zeros(n,1)];
y = [y0; zeros(n,1)];

for i=1:n
  x(i+1) = x(i)+h;
  c1 = y(i);
  c2 = y(i) + [h/2 * feval(funz,x(i),c1)];
  c3 = y(i) - [h*feval(funz,x(i),c1)] + [2*h*feval(funz,x(i)+h/2,c2)];
  y(i+1) = y(i) + h/6*[feval(funz,x(i),c1) + 4*feval(funz,x(i)+h/2,c2) + feval(funz,x(i)+h,c3)];
end
```

```
% funzione del metodo di Runge-Kutta di ordine 4

function [x,y]=erk4(funz,x0,y0,xf,n)

h = (xf - x0)/ n;
x = [x0; zeros(n,1)];
y = [y0; zeros(n,1)];

for i=1:n
  x(i+1) = x(i)+h;
  c1 = y(i);
  c2 = y(i) + [h/2 * feval(funz,x(i),c1)];
  c3 = y(i) + [h/2 * feval(funz,x(i)+h/2,c2)];
  c4 = y(i) + h*feval(funz,x(i)+h/2,c3);
  y(i+1) = y(i)+ h/6*[feval(funz,x(i),c1) + 2*feval(funz,x(i)+h/2,c2) + 2*feval(funz,x(i)+h/2,c3) +
feval(funz,x(i)+h,c4)];
end
```

2.3.2. Analisi dei risultati

Dalla rappresentazione grafica delle soluzioni ottenute dai quattro metodi si è dimostrato ciò che già si è osservato nel caso precedente. La curva che approssima con minore precisione la soluzione reale è quella relativa al metodo di primo ordine di Eulero-Cauchy, mentre la più accurata è la curva del metodo di Runge-Kutta del quarto ordine (Fig. 10).

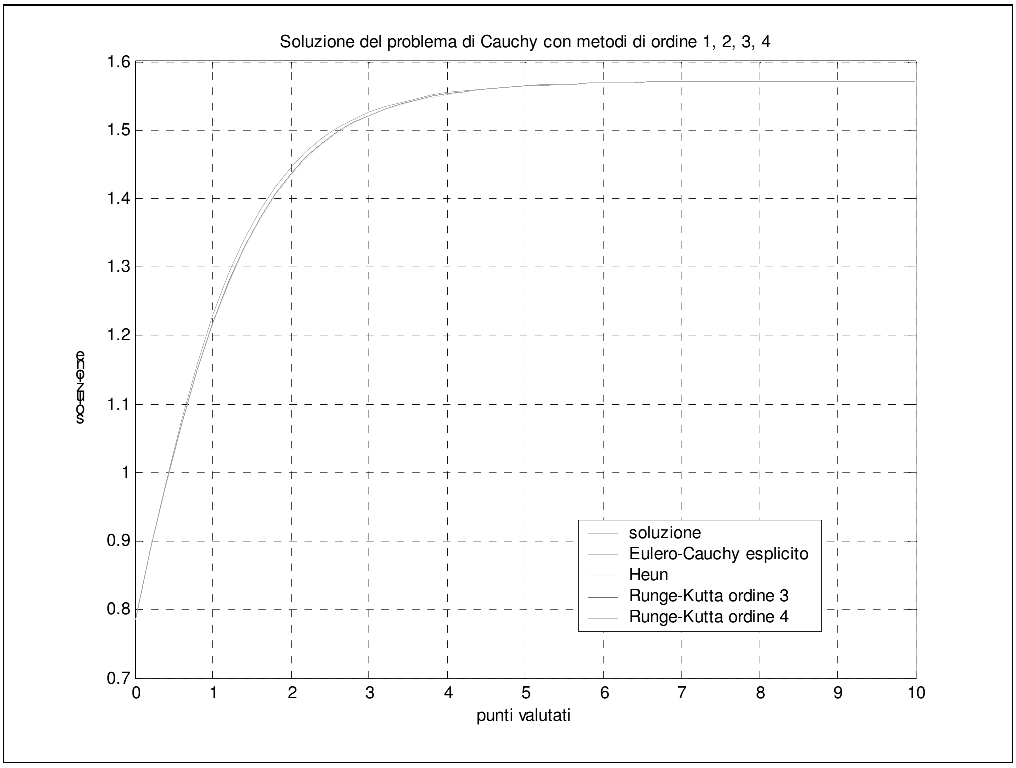

Figura 10. Soluzione del problema di Cauchy con metodi di ordine 1, 2, 3, 4.

Anche ingrandendo l'immagine in un generico intervallo la soluzione ottenuta con il metodo del quarto ordine sembra decalcare la soluzione reale (Fig. 11).

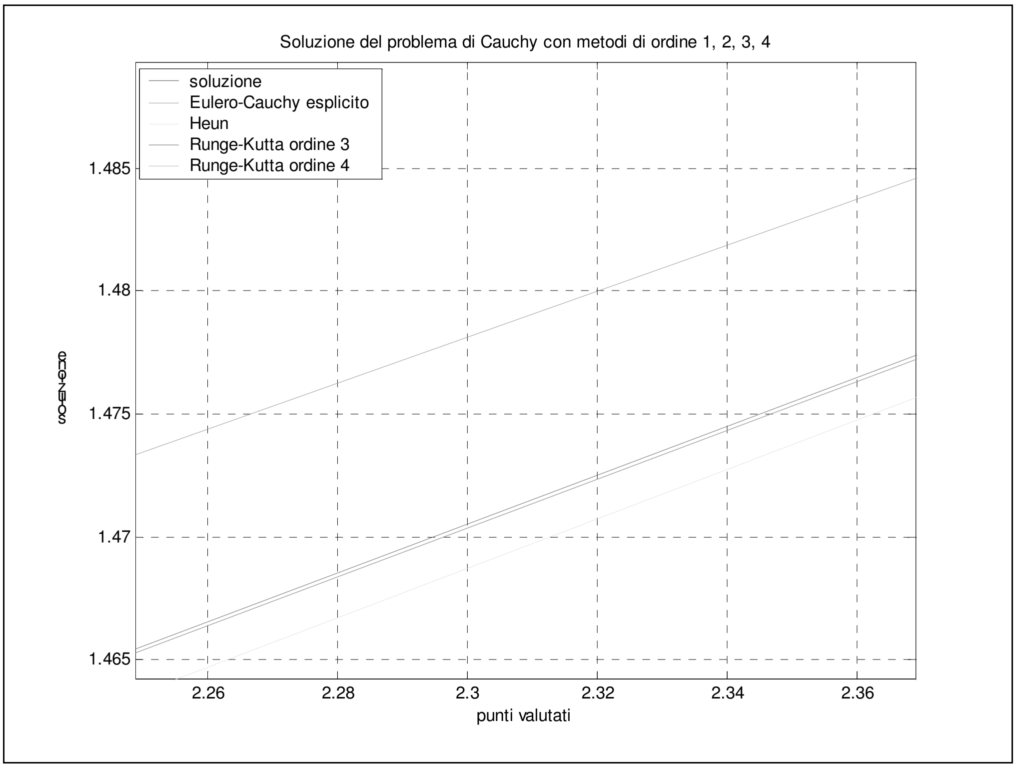

Figura 11. Ingrandimento della soluzione del problema di Cauchy con metodi di ordine 1, 2, 3, 4.

Infine, si è analizzato l'errore tra la soluzione reale e quella raggiunta con i quattro metodi.
In questo secondo caso si è visualizzata una tabella in cui nelle colonne si sono indicati i metodi di risoluzione e nelle ascisse i punti di iterazione. Per semplicità si sono riportati solo parte dei punti, ma anche in questo caso si è imposto un numero di iterazione pari a 50 (tab. 8).

ERRORI NELLA SOLUZIONE				
P. valutati	**Eul.-C. esplic**	**Heun**	**ERK3**	**ERK4**
0.0	0.00e+000	0.00e+000	0.00e+000	0.00e+000
0.2	6.60e-004	4.03e-004	2.93e-006	1.01e-006
0.4	3.00e-003	9.21e-004	2.28e-005	2.70e-006
0.6	6.08e-003	1.46e-003	5.55e-005	4.96e-006
0.8	8.94e-003	1.92e-003	9.29e-005	7.38e-006
1.0	1.10e-002	2.25e-003	1.27e-004	9.51e-006
1.2	1.20e-002	2.41e-003	1.51e-004	1.10e-005

1.4	1.21e-002	2.44e-003	1.65e-004	1.17e-005
1.6	1.16e-002	2.35e-003	1.68e-004	1.18e-005
1.8	1.07e-002	2.18e-003	1.62e-004	1.13e-005
2.0	9.52e-003	1.98e-003	1.52e-004	1.05e-005
2.2	8.33e-003	1.76e-003	1.38e-004	9.50e-006
2.4	7.17e-003	1.53e-003	1.22e-004	8.41e-006
2.6	6.10e-003	1.32e-003	1.07e-004	7.33e-006
2.8	5.14e-003	1.13e-003	9.20e-005	6.31e-006
3.0	4.31e-003	9.56e-004	7.85e-005	5.38e-006
3.2	3.59e-003	8.05e-004	6.65e-005	4.54e-006
3.4	2.98e-003	6.74e-004	5.59e-005	3.82e-006
3.6	2.47e-003	5.62e-004	4.68e-005	3.19e-006
3.8	2.04e-003	4.67e-004	3.90e-005	2.66e-006
4.0	1.68e-003	3.87e-004	3.24e-005	2.21e-006
4.2	1.38e-003	3.20e-004	2.68e-005	1.83e-006
4.4	1.13e-003	2.64e-004	2.22e-005	1.51e-006
4.6	9.31e-004	2.18e-004	1.83e-005	1.25e-006
4.8	7.64e-004	1.79e-004	1.51e-005	1.03e-006
5.0	6.27e-004	1.48e-004	1.24e-005	8.45e-007
…	…	…	…	…

Tabella 8. Errore nella soluzione ottenuta con metodi monostep espliciti.

Dall'andamento dell'errore si ottengono i risultati aspettati: la curva relativa a ERK4 giace quasi totalmente sull'asse delle ascisse (Fig. 12).

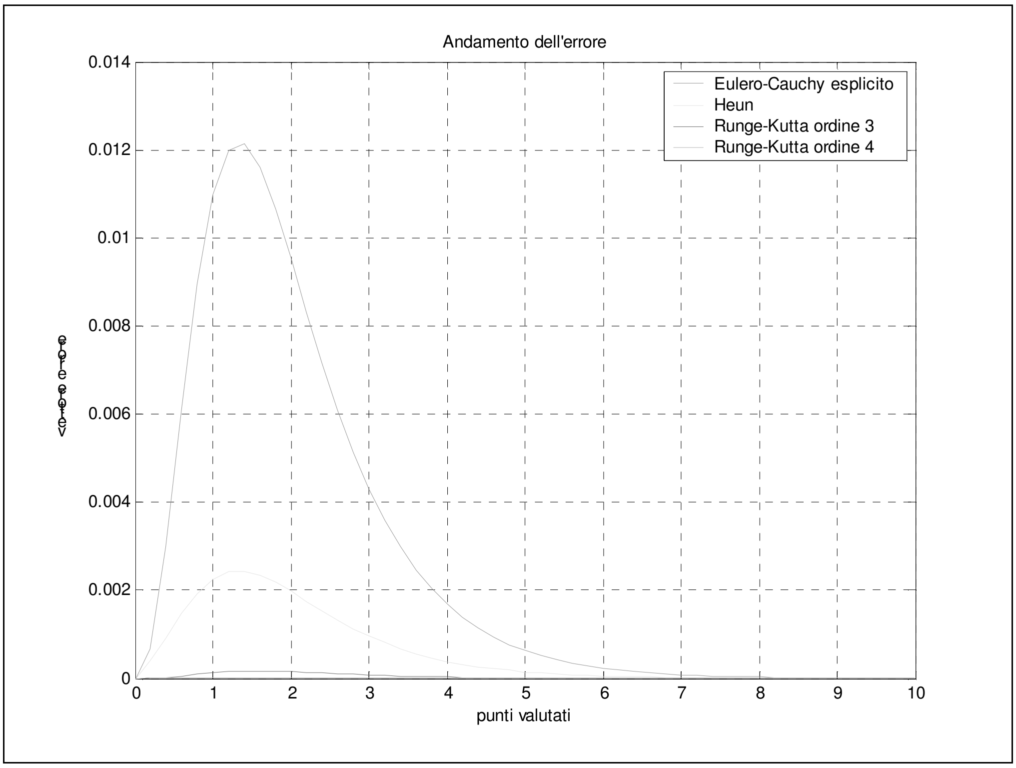

Figura 12. Andamento dell'errore nella soluzione con metodi di ordine 1, 2, 3, 4.

2.4. Confronto tra metodi del quarto ordine: espliciti ed impliciti

2.4.1 Impostazione del problema

Come ultima prova si sono confrontati due metodi del quarto ordine: il metodo esplicito ERK4 e il metodo implicito Predictor-Corrector.

Nei metodi Predictor-Corrector vengono utilizzate due formule di pari ordine, una esplicita e una implicita. Mediante la formula esplicita viene calcolata una prima stima del valore della soluzione che viene poi raffinata mediante una o più valutazioni della formula implicita.

Un esempio è fornito dal metodo di Milne, implementato di seguito. Trattandosi di un metodo multistep, oltre che implicito, esso deve essere inizializzato col calcolo dei valori η_1, η_2 e η_3 mediante una formula monostep esplicita del quarto ordine.

Per questo motivo all'interno della funzione che calcola la soluzione si è realizzata una seconda funzione che fornisce i primi tre valori valutati. Per il calcolo di questi valori si è considerato il metodo ERK4. Pertanto, oltre all'inizializzazione dei vettori dove allocare i valori di x e y calcolati ad ogni passo, è stato necessario inizializzare anche un vettore colonna per i valori della funzione.

```
% funzione del metodo Predictor-Corrector

function [x,y]=pre_cor(funz,x0,y0,xf,n)

h = (xf - x0)/ n;
x = [x0; zeros(n,1)];
y = [y0; zeros(n,1)];
f = [feval(funz,x0,y0); zeros(n,1)];          % inizializza il vettore colonna f
[x,y,f]=erk4_aiuto(funz,x,y,f,h);             % funzione del metodo ERK4 per i primi 4 punti

for i=4:n
   x(i+1) = x(i)+h;
   ynew = y(i-3) + 4/3*h*[2*f(i-2)-f(i-1)+2*f(i)];
   y(i+1) = y(i-1) + 1/3*h*[f(i-1)+4*f(i)+feval(funz,x(i+1),ynew)];
   f(i+1) = feval(funz,x(i+1),y(i+1));
end


% funzione del metodo ERK4 per i primi 4 punti

function [x,y,f]=erk4_aiuto(funz,x,y,f,h)

for i=1:3
   x(i+1) = x(i)+h;
   c1 = y(i);
   c2 = y(i) + [h/2 * feval(funz,x(i),c1)];
   c3 = y(i) + [h/2 * feval(funz,x(i)+h/2,c2)];
   c4 = y(i) + h*feval(funz,x(i)+h/2,c3);
   y(i+1)  =  y(i)  +  h/6*[feval(funz,x(i),c1)  +  2*feval(funz,x(i)+h/2,c2)  +  2*feval(funz,x(i)+h/2,c3)  +
feval(funz,x(i)+h,c4)];
   f(i+1) = feval(funz,x(i+1),y(i+1));
end
```

2.4.2. Analisi dei risultati

Anche in questo caso si è visualizzato l'andamento delle soluzioni approssimate e della soluzione reale.

Per rendere più chiari i risultati si sono riportate le curve soluzione dei due metodi su due grafici distinti, così da dimostrare l'alta precisione di entrambi i metodi (Fig. 13). Anche con un ingrandimento, entrambe le curve sembra decalchino la soluzione.

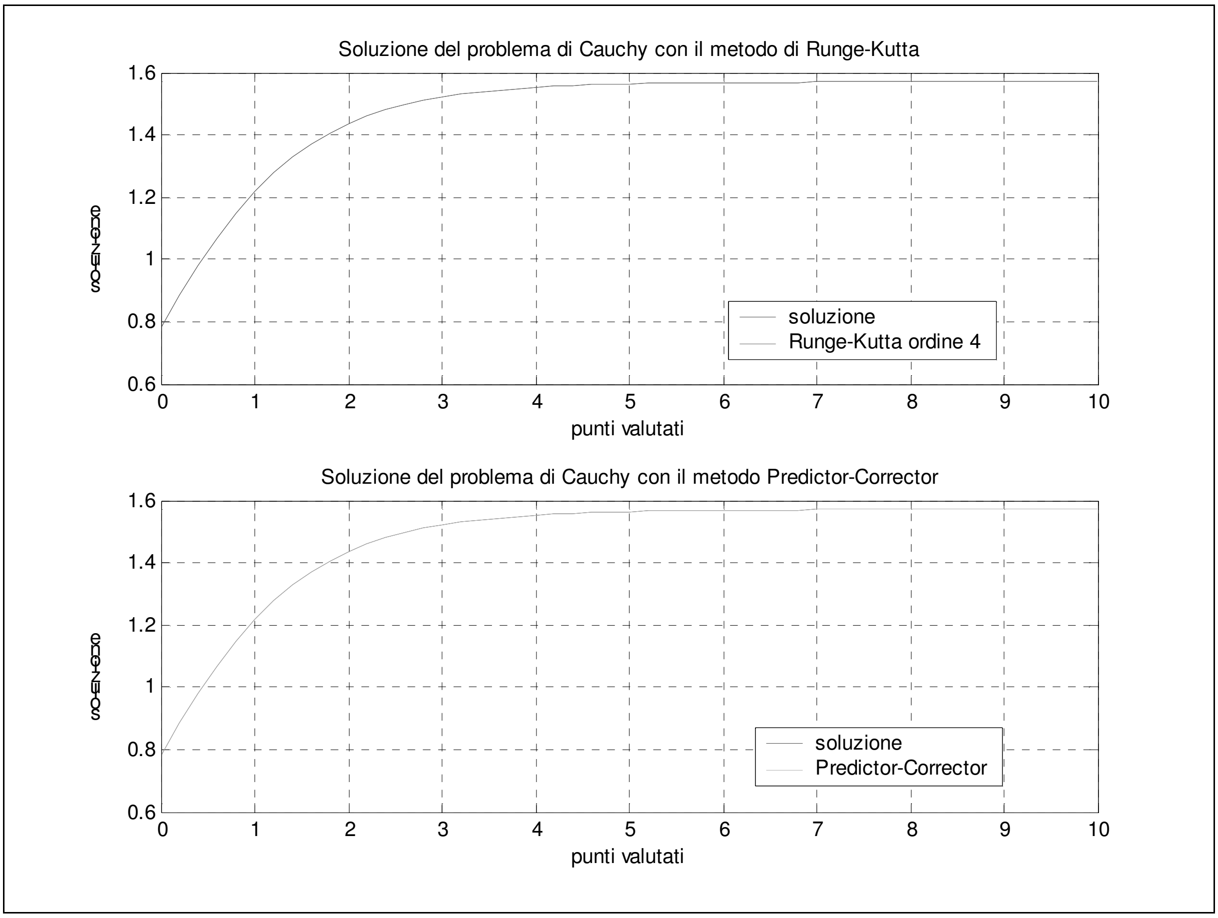

Figura 13. Soluzione del problema di Cauchy con il metodo di ERK4 e Predictor-Corrector.

Nella tabella sottostante si riportano gli errori nella soluzione di entrambi i metodi (Tab. 9).

ERRORI NELLA SOLUZIONE		
P. valutati	**ERK4**	**Pre-Cor**
0.0	0.00e+000	0.00e+000
0.2	1.01e-006	1.01e-006
0.4	2.70e-006	2.70e-006
0.6	4.96e-006	4.96e-006
0.8	7.38e-006	5.32e-006
1.0	9.51e-006	1.40e-005
1.2	1.10e-005	5.68e-006
1.4	1.17e-005	2.34e-005
1.6	1.18e-005	1.01e-005
1.8	1.13e-005	2.17e-005

2.0	1.05e-005	6.44e-006
2.2	9.50e-006	1.52e-005
2.4	8.41e-006	1.47e-006
2.6	7.33e-006	9.37e-006
2.8	6.31e-006	1.78e-006
3.0	5.38e-006	5.64e-006
3.2	4.54e-006	3.20e-006
3.4	3.82e-006	3.59e-006
3.6	3.19e-006	3.49e-006
3.8	2.66e-006	2.52e-006
4.0	2.21e-006	3.24e-006
4.2	1.83e-006	1.95e-006
4.4	1.51e-006	2.81e-006
4.6	1.25e-006	1.61e-006
4.8	1.03e-006	2.35e-006
5.0	8.45e-007	1.38e-006
5.2	6.94e-007	1.94e-006
5.4	5.70e-007	1.20e-006
5.6	4.68e-007	1.59e-006
5.8	3.84e-007	1.04e-006
6.0	3.15e-007	1.31e-006
6.2	2.58e-007	9.03e-007
6.4	2.12e-007	1.07e-006
6.6	1.74e-007	7.79e-007
6.8	1.42e-007	8.84e-007
7.0	1.17e-007	6.69e-007
7.2	9.55e-008	7.30e-007
7.4	7.82e-008	5.72e-007
7.6	6.40e-008	6.05e-007
7.8	5.25e-008	4.88e-007
8.0	4.30e-008	5.02e-007

8.2	3.52e-008	4.14e-007
8.4	2.88e-008	4.18e-007
8.6	2.36e-008	3.51e-007
8.8	1.93e-008	3.49e-007
9.0	1.58e-008	2.97e-007
9.2	1.29e-008	2.92e-007
9.4	1.06e-008	2.50e-007
9.6	8.68e-009	2.44e-007
9.8	7.11e-009	2.11e-007
10.0	5.82e-009	2.05e-007

Tabella 9. Errore nella soluzione ottenuta con i metodi di ERK4 e Predictor-Corrector.