Esercitazione 4

Sistemi Lineari

File dell'esercitazione reperibile su:

http://campus.unibo.it/

Slide dell'esercitazione:

http://tinyurl.com/CalcoloEs4slide

Risoluzione di un Sistema Lineare:

$$Ax = b$$

Matrice

$$A \rightarrow (n \times n)$$

Termine noto

$$b \rightarrow (n \times 1)$$

Soluzione:

$$x \rightarrow (n \times 1)$$

$$x = A^{-1} \cdot b$$

in ambiente Matlab:

$$x = inv(A) \cdot b$$

$$x = A \backslash b$$

A Matrice Triangolare Inferiore:

Algoritmo di sostituzione all'avanti

$$L = \begin{bmatrix} l_{11} & 0 & \dots & \dots & 0 \\ l_{21} & l_{22} & 0 & \dots & 0 \\ l_{31} & l_{32} & l_{33} & & \\ \dots & \dots & \dots & \dots & \\ l_{n1} & l_{n2} & l_{n3} & \dots & l_{nn} \end{bmatrix}$$

$$\begin{cases} x_1 = \dfrac{b_1}{l_{11}} \\ x_2 = (b_2 - l_{21}x_1)/l_{22} \\ \dots \\ x_k = (b_k - (l_{k1}x_1 + l_{k2}x_2 + \dots l_{k,k-1}x_{k-1}))/l_{k,k} \qquad k = 2,\dots,n \end{cases}$$

```
for i=1,2,..,n
    x_i=b_i
        for j=1,2,...,i-1
            x_i=x_i-l_ij·x_j
        end for j
    x_i=x_i/a_ii
end for i
```

A Matrice Triangolare Inferiore:

Algoritmo di sostituzione all'avanti

$$L = \begin{bmatrix} 1 & 0 & 0 \\ 2 & -1 & 0 \\ 2 & 1 & 3 \end{bmatrix} \qquad b = \begin{bmatrix} 1 \\ -1 \\ 3 \end{bmatrix}$$

$$\begin{cases} \boxed{1 \cdot x_1 \qquad\qquad\qquad = \quad 1} \\ \boxed{2 \cdot x_1 \quad -1 \cdot x_2 \qquad = \quad -1} \\ 2 \cdot x_1 \quad 1 \cdot x_2 \quad 3 \cdot x_3 = \quad 3 \end{cases}$$

$$x_1 = b_1 / l_{1,1}$$

$$x_2 = (b_2 + l_{2,1} \cdot x_1) / l_{2,2}$$

A Matrice Triangolare Superiore:

Algoritmo di sostituzione all'indietro

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} & \dots & r_{1n} \\ 0 & r_{22} & r_{23} & \dots & r_{2n} \\ 0 & 0 & r_{33} & \dots & r_{3n} \\ \dots & \dots & \dots & \dots & \\ 0 & 0 & 0 & \dots & r_{nn} \end{bmatrix}$$

```
for  i=n,n-1,...,1
      x_i=b_i
          for j=i+1, ...,n
              x_i=x_i-r_ij·x_j
          end for j
x_i=x_i/a_ii
end for i
```

$$\begin{cases} x_n = \dfrac{b_n}{r_{n,n}} \\ x_{n-1} = (b_{n-1} - r_{n-1,n}x_n) / r_{n-1,n-1} \\ \dots \\ x_k = (b_k - (r_{k,k+1}x_{k+1} + r_{k,k+2}x_{k+2} + \dots r_{k,n}x_n)) / r_{k,k} \qquad k = n-1, n-2, \dots, 1 \end{cases}$$

A Matrice Triangolare Superiore:

Algoritmo di sostituzione all'indietro

$$R = \begin{bmatrix} 2 & 4 & 1 \\ 0 & -1 & 1 \\ 0 & 0 & 3 \end{bmatrix} \qquad b = \begin{bmatrix} 1 \\ -1 \\ 3 \end{bmatrix}$$

$$x_{n-1} = (b_{n-1} + r_{n-1,n} \cdot x_n) / r_{n-1,n-1}$$

$$\begin{cases} 2 \cdot x_1 \quad 4 \cdot x_2 \quad 1 \cdot x_3 = 1 \\ \boxed{-1 \cdot x_2 \quad 1 \cdot x_3 = -1} \\ \boxed{3 \cdot x_3 = 3} \end{cases}$$

$$x_n = b_n / r_{n,n}$$

Fattorizzazione di Gauss:

IDEA: Scomporre la matrice A nel prodotto di due matrici triangolari (una inferiore L e una superiore R)

$$A = L \cdot R$$

Così facendo, trovare la risoluzione del sistema lineare di partenza equivale a risolvere due sistemi triangolari:

$$Ax = b \rightarrow L\underbrace{\boxed{Rx}}_{y} = b \quad \begin{cases} Ly = b & \text{1} \\ Rx = y & \text{2} \end{cases}$$

Creazione delle Matrici L e R:

```
for k=1,...n-1
        for i=k+1,...n
            if(A(k,k)==0) stop
                L(i,k)=A(i,k)/A(k,k)
                    for j=k+1,..n
                        A(i,j)=A(i,j)-L(i,k)*A(k,j)
                    end
        end
end
```

N.B. Alla fine di questo algoritmo, la matrice R è memorizzata nel "triangolo superiore" della matrice A

$$A = \begin{bmatrix} 1 & 2 & 0 \\ \xcancel{2} & -1 & 4 \\ \xcancel{2} & \xcancel{1} & 3 \end{bmatrix}$$

Esercitazione 4:

Costruzione di 3 m-function:

1. **Risoluzione di sistemi triangolari superiori**
2. **Risoluzione di sistemi triangolari inferiori**
3. **Fattorizzazione LR di una matrice A**

Test della stabilità della fattorizzazione

Consiglio:

Creare matrici a scelta per testare le m-function e provare anche a risolvere un sistema lineare utilizzando tutte e 3 le funzioni insieme e confrontando il risultato con la soluzione di Matlab

Per visualizzare i risultati:

- *function* ***disp(x)*** *-> visualizza la variabile x nel workspace (numero o stringa)*
- *function* ***fprintf(****'Stringa da visualizzare'****)***
- *visualizza la stringa indicata*

N.B. si possono visualizzare i numeri con %i, %3.2f… ecc

```
>> fprintf('Il valore di x è %i\n',x)
     Il valore di x è 1
>> fprintf('Il valore di x è %3.2d\n',x)
     Il valore di x è 3.25e+00
>> fprintf('Il valore di x è %3.2f\n',x)
     Il valore di x è 3.25
```

help

Quali comandi vi possono servire per le relazioni?

Si accettano suggerimenti!!!

Consiglio:
Potete consultare le dispense di Matlab della Prof. Lazzaro dal materiale dell'anno scorso!