Laboratorio Calcolo Numerico
14 maggio 2019

Approssimazione ai minimi quadrati

Esercizio 1

Si scriva la function `find_coeff` che calcola i coefficienti del polinomio di miglior approssimazione ai minimi quadrati di una serie di dati sperimentali posti nei vettori `x` (ascisse) e `y` (ordinate). La function richiesta deve avere come parametri di ingresso la matrice rettangolare V di dimensioni $n \times (m+1)$, essendo m il grado del polinomio di approssimazione, e il vettore y, e deve restituire il vettore c con i coefficienti $a_i,\ i = 0, 1, \ldots, m$, del polinomio di miglior approssimazione, $P_m(x) = a_m x^m + a_{m-1}x^{m-1} + \ldots + a_1 x + a_0$ ottenuto come soluzione nel senso dei minimi quadrati del sistema rettangolare

$$Vc = y \tag{1}$$

con

$$V = \begin{pmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^m \\ 1 & x_2 & x_2^2 & \ldots & x_2^m \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 & \ldots & x_n^m \end{pmatrix}, \qquad c = \begin{pmatrix} a_0 \\ a_1 \\ \ldots \\ a_m \end{pmatrix}, \qquad y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

La matrice rettangolare V è una matrice di Vandemonde, e può essere creata in Matlab tramite il comando `vander(x)`. Per trovare la soluzione di (1) si può risolvere il sistema delle equazioni normali:

$$V^T V c = V^T y$$

La sintassi della function deve essere la seguente:

```
function c=find_coeff(V,y)
```

MOLTO IMPORTANTE: come ultima istruzione della function si scriva

```
c=c(end:-1:1);
```

Esercizio 2

Si vogliono approssimare i seguenti dati sperimentali

x	1200	1201	1201.6	1202	1203	1203.5	1205
y	3	2	1.5	1.5	1	0.7	0

nel senso dei minimi quadrati con un polinomio di grado 2.

Si scriva uno script `lsqscript.m` che

1. Definisca le 7 coppie di dati, memorizzandole in due vettori colonna x e y.
2. Costruisca la matrice di Vandermonde V.
3. Calcoli i coefficienti del polinomio di approssimazione invocando la function `find_coeff`.
4. Disegni su un grafico i dati sperimentali e il polinomio approssimante ai minimi quadrati, valutato su 200 punti dell'intervallo definito dai nodi.
5. Agiunga al grafico il polinomio di grado ≤ 6 che interpola i dati sperimentali. Si spieghino i risultati ottenuti.

Risoluzione di sistemi lineari-Metodi diretti

Esercizio 3
Si scriva una function per l'implementazione della risoluzione di un sistema lineare triangolare superiore $Ux = b$ con il metodo di **sostituzione all'indietro** secondo il seguente algoritmo:

for $i = n, \ldots, 1$ **do**
  $x_i = b_i$
  for $j = i + 1, \ldots, n$ **do**
    $x_i = x_i - u_{ij}x_j$
  end for
  $x_i = \frac{x_i}{u_{ii}};$
end for

La sintassi della function deve essere la seguente:

```
function [x]= BackSolv(U,b,n)
```

Esercizio 4
Si scriva una function per l'implementazione della risoluzione di un sistema lineare triangolare inferiore $Lx = b$ con il metodo di **sostituzione in avanti** secondo il seguente algoritmo:

for $i = 1, \ldots, n$ **do**
  $x_i = b_i$
  for $j = 1, \ldots, i - 1$ **do**
    $x_i = x_i - l_{ij}x_j$
  end for
  $x_i = \frac{x_i}{l_{ii}};$
end for

La sintassi della function deve essere la seguente:

```
function [x]= ForwSolv(L,b,n)
```

Nota bene: In entrambe le function richieste nei due precedenti esercizi ci si accerti che le matrici U, L, rispettivamente, siano quadrate e che abbiano tutti gli elementi diagonali diversi da zero, altrimenti durante l'esecuzione dell'algoritmo vi sarebbe una divisione per zero.

Esercizio 5

L'algoritmo del metodo di eliminazione di Gauss si può schematizzare come segue

for $k = 1, \ldots, n - 1$ **do**
  for $i = k + 1, \ldots, n$ **do**
    $l_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}$
    for $j = k, \ldots, n$ **do**
      $a_{ij}^{(k)} = a_{ij}^{(k)} - l_{ik}a_{kj}^{(k)}$
    end for
  end for
end for

A partire dall'algoritmo precedente si scriva una `function` MATLAB che calcoli e restituisca i due fattori triangolari L ed U della fattorizzazione LU di una matrice A data. La sintassi della function deve essere la seguente:

```
function [L,U]= lugauss(A)
```

Nota bene: la matrice L deve essere inizializzata alla matrice identità dello stesso ordine di A, e alla fine la matrice U restituita è la (parte triangolare superiore della) matrice A risultante dopo le $n-1$ trasformazioni elementari di Gauss.

Esercizio 6

Si scriva uno script MATLAB che:

1. Definisca la matrice A come la matrice di Hilbert di ordine $n = 8$ e poi cambi i primi due elementi della seconda riga nel modo seguente: $A(2,1) = 2 * A(1,1)$ e $A(2,2) = 2 * A(1,2) - \epsilon$, fissato $\epsilon = 10^{-12}$.

2. Definisca il termine noto b in modo che la soluzione esatta del sistema sia il vettore con tutte le componenti pari a 1.

3. Calcoli la soluzione del sistema lineare $Ax = b$ a partire dalla fattorizzazione LU della matrice A calcolata mediante la function `lugauss`, e quindi risolvendo i due sistemi triangolari mediante il comando \ di MATLAB .

4. Visualizzi a video il vettore soluzione ottenuto. Si usi `format long` per vedere tutte le cifre significative.

5. Calcoli il residuo relativo $r_{rel} = \dfrac{||b - A\bar{x}||}{\|b\|}$ e l'errore relativo $e_{rel} = \dfrac{||x - \bar{x}||}{\|\bar{x}\|}$, dove $\bar{x}$ indica la soluzione esatta del sistema. Si usi la norma euclidea.

6. Ripeta i punti (3)-(5) ma risolvendo il sistema questa volta con la fattorizzazione LU **con pivoting** fornita dalla function MATLAB `[L,U,P] = lu(A)`.

Dopo aver analizzato i valori ottenuti del residuo e dell'errore relativi in ogni caso si risponda alle seguenti domande:

- Perché il residuo relativo nel primo caso (LU calcolata senza pivoting) non è dell'ordine della precisione di macchina?

- Com'è il residuo nel secondo caso (LU calcolata con pivoting per righe)? Perché l'errore non è dell'ordine della precisione di macchina ma molto più grande?