CALCOLO NUMERICO

Ing. chimica e dei materiali, Canale A - A.A. 2023-24

Laboratorio 10 - 15 maggio 2024

- ***ESERCIZIO 1.*** È noto che l'inversa di una matrice triangolare superiore (rispettivamente inferiore) è ancora triangolare superiore (rispettivamente inferiore) con gli elementi diagonali uguali ai reciproci degli elementi diagonali della matrice data. Si può sfruttare tale particolarità per definire un algoritmo per il calcolo dell'inversa adattato a tali matrici. Tale algoritmo risolve ancora n sistemi triangolari, ma anzichè essere tutti di ordine n, sono di ordine i, per $i = 1, \dots, n$.

  Per le matrici triangolari superiori, indicando $U = (u_{ij})$ e la matrice inversa $U^{-1} = Z = (z_{ij})$, possiamo calcolare gli elementi non nulli con le seguenti formule:

$$\left.\begin{aligned} & z_{ii} = u_{ii}^{-1}, \\ & z_{ji} = -u_{jj}^{-1} \sum_{k=j+1}^{i} u_{jk} z_{ki}, \qquad j = i-1, i-2, \dots, 1 \end{aligned}\right\} i = n, n-1, \dots, 1.$$

  – Si scriva un algoritmo che, data una matrice U e la sua dimensione n, implementi tali formule e restituisca la matrice $Z = U^{-1}$. (Iniziare scrivendo lo pseudocodice)
  L'algoritmo dovrà avere la seguente intestazione:

$$Z = \textbf{trisup\_inv}(U, n)$$

  – Si inseriscano all'inizio dell'algoritmo tre controlli (ovvero tre strutture **if** separate) per verificare che la matrice sia quadrata, che la matrice sia non singolare ($\det(U) \neq 0$) ed infine che sia effettivamente triangolare superiore. In caso positivo si esegua il calcolo, altrimenti si arresti l'esecuzione.

- ***ESERCIZIO 2.*** (function lab. 9: `triinf`, `trisup`; comandi matlab: `isequal`, `diag`, `all`, `chol`, `format`, `norm`, uso della variabile `flag`, istruzioni condizionali di controllo)

  Facendo riferimento all'esercizio 2 del laboratorio 9 (sistema idraulico), si consideri il sistema lineare $A\mathbf{p} = \mathbf{b}$, dove:

$$A = \begin{pmatrix} -0.370 & 0.050 & 0.050 & 0.070 \\ 0.050 & -0.116 & 0 & 0.050 \\ 0.050 & 0 & -0.116 & 0.050 \\ 0.070 & 0.050 & 0.050 & -0.202 \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} -2 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

  Si vuole determinare la soluzione del sistema usando il *metodo di Cholesky*.

  Si noti, però, che la matrice A, pur essendo *simmetrica*, **NON PUÒ** essere *definita positiva*, in quanto si può dimostrare il seguente teorema (che fornisce una condizione necessaria ma NON sufficiente):

  > Se una matrice A reale e simmetrica è *definita positiva*, allora **necessariamente TUTTI** i suoi *elementi diagonali* devono essere **positivi**.

  – Si modifichino *opportunamente* la matrice A ed il vettore $\mathbf{b}$ in modo da trasformare il sistema

$$A\mathbf{p} = \mathbf{b}$$

  nel sistema **equivalente**

$$\widetilde{A}\mathbf{p} = \widetilde{\mathbf{b}},$$

  in cui la matrice $\widetilde{A}$ sia *simmetrica* e *definita positiva* (ovvero abbia tutti gli elementi diagonali positivi).

  – Si costruisca uno script `idracholscript.m` che

  * definisca la matrice A ed il vettore termine noto $\mathbf{b}$ dopo aver letto una matrice aumentata contenuta nel file di testo `idra.dat`, fornito dai docenti, (usare il comando `load`)
  * trasformi la matrice `A` (=A) nella matrice `A1` (=$\widetilde{A}$) e il vettore `b` (=$\mathbf{b}$) nel vettore `b1` (=$\widetilde{\mathbf{b}}$)
  * controlli che la matrice sia simmetrica (come fare?)
  * controlli che tutti gli elementi diagonali della matrice siano positivi (senza usare cicli `for` - ci sono molti modi per fare questo test)
  * cerchi di calcolare la matrice `L` della fattorizzazione di Cholesky con il comando `[L,flag]=chol(A1,'lower')`, che crea la matrice *triangolare inferiore* `L` e la *variabile di controllo* `flag`
  * se `flag` $\neq$ `0` esca con un errore;
    se `flag` = `0` la matrice `A1` è *simmetrica* e *definita positiva*, ed L è la matrice della fattorizzazione. Quindi, si può continuare
    (per maggiori informazioni, si digiti `help chol` nella finestra dei comandi di Matlab)
  * calcoli la matrice `LLT` = $L \times L^T$ e la confronti visivamente con la matrice `A1` = $\widetilde{A}$, scrivendole (una dopo l'altra), entrambe in doppia precisione, in un file di testo esterno di nome `matrici.txt` (si usi il comando Matlab `save`). Dopo aver guardato il contenuto del file, cosa si puońotare? perchè?
  * visualizzi anche sulla Command Window (si usi un `format` adeguato) la matrice `A1` e la matrice `LLT`;
  * risolva il sistema $\widetilde{A}\mathbf{p} = \widetilde{\mathbf{b}}$ utilizzando tale fattorizzazione e le due function `triinf.m` e `trisup.m` create come soluzione degli esercizi del laboratorio 9, calcolando la soluzione $\mathbf{p}$
  * calcoli la soluzione dello stesso sistema (assegnandola ad un altro vettore) anche con il comando Matlab `A\b`

    * visualizzi il vettore `b1`, la dimensione del sistema, la matrice `L` della fattorizzazione, il vettore soluzione ottenuto utilizzando la risoluzione dei sistemi triangolari ed il vettore soluzione ottenuto utilizzando l'operatore Matlab \;
    * visualizzi la norma euclidea del vettore differenza tra le due soluzioni (quella ottenuta con i due sistemi triangolari e quella ottenuta con il comando Matlab \)

- ***ESERCIZIO 2b.*** Con riferimento all'esercizio 2, verificare:
  - che il controllo previsto sulla condizione necessaria espressa dal teorema sia corretto: cercare di fattorizzare la matrice A che, pur essendo simmetrica non è definita positiva (utilizzare lo script precedente per effettuare questo controllo inserendo un'istruzione temporanea)
  - che il controllo sulla simmetria della matrice sia corretto (nello script precedente modificare temporaneamente un solo elemento al di fuori della diagonale di `A1`, ponendolo, ad esempio, a zero, e poi tale istruzione potrà essere cancellata)

- **CONSEGNA:** Si crei un file .zip con i codici creati per l'esercizio 2 e il file `matrici.txt`. Si rinomini lo zip (**cognome_nome_lab10.zip**) e si carichi su moodle.

- ***ESERCIZIO 3a.*** (comandi `for`, `hilb`, `pascal`, `vander`, `rand`, `cond`, `condest`, `rcond`)

  Questo esercizio serve a comprendere meglio il problema del malcondizionamento.

  Si crei uno script `provamc.m` che:

  - crei 3 matrici quadrate di Hilbert di dimensione 5^i per $i = 1, \ldots, 3$, visualizzi la loro dimensione, il loro numero di condizionamento (ottenuto con il comando `cond`), la stima del loro numero di condizionamento (ottenuto con il comando `condest`), ed la stima del suo reciproco (ottenuto con il comando `rcond`).
  - crei 3 matrici quadrate di Pascal di dimensione 5^i per $i = 1, \ldots, 3$, visualizzi la loro dimensione, il loro numero di condizionamento (ottenuto con il comando `cond`), la stima del loro numero di condizionamento (ottenuto con il comando `condest`), ed la stima del suo reciproco (ottenuto con il comando `rcond`).
  - crei 3 matrici quadrate di Vandermonde di dimensione 5^i per $i = 1, \ldots, 3$, visualizzi la loro dimensione, il loro numero di condizionamento (ottenuto con il comando `cond`), la stima del loro numero di condizionamento (ottenuto con il comando `condest`), ed la stima del suo reciproco (ottenuto con il comando `rcond`). Si usi il comando `help`, in quanto per creare queste matrici NON basta dare la dimensione desiderata come valore in ingresso, ma bisogna prima definire un vettore `v` che abbia la dimensione desiderata e che contenga (solo come esempio per scrivere lo script) componenti equispaziate nell'intervallo $[1, 3]$. Tale vettore deve essere dato come argomento al comando `vander`.
  - crei 3 matrici quadrate random di dimensione 5^i per $i = 1, \ldots, 3$, visualizzi la loro dimensione, il loro numero di condizionamento (ottenuto con il comando `cond`), la stima del loro numero di condizionamento (ottenuto con il comando `condest`), ed la stima del suo reciproco (ottenuto con il comando `rcond`).

  Cosa si nota?
  Solo per verificare l'esattezza dello script predisposto di seguito vengono riportati **solo alcuni** risultati (dovrebbero coincidere con quelli degli studenti a parte per la matrice `random`):

```
MATRICI DI HILBERT
dimensione = 5
numero condizionamento = 476607.2502
stima numero condizionamento = 943656
stima reciproco numero condizionamento = 1.0597e-06

MATRICI DI PASCAL
dimensione = 25
numero condizionamento = 4.393649988032901e+22
stima numero condizionamento = 1.498146050895393e+22
stima reciproco numero condizionamento = 6.6749e-23

MATRICI DI VANDERMONDE
dimensione = 125
numero condizionamento = 1.010179490133965e+74
stima numero condizionamento = 2.89506093242887e+75
stima reciproco numero condizionamento = 3.4542e-76

MATRICE RANDOM
dimensione = 125
numero condizionamento = 5491.5101
```

```
stima numero condizionamento = 12752.3469
stima reciproco numero condizionamento = 7.8417e-05
```

- ***ESERCIZIO 3b.***

  Si crei uno script `provasysmc.m` che:

  - crei una matrice quadrata A di Pascal di dimensione 40
  - visualizzi il numero di condizionamento (ottenuto con il comando `cond`)
  - assegnato il vettore $u = (1, \ldots, 1)^T$, calcoli il termine noto b come $b = Au$ (in tal modo la soluzione esatta del sistema coincide con il vettore unitario)
  - risolva il sistema $Ax = b$ con l'operatore Matlab $\backslash$
  - visualizzi la norma euclidea del vettore differenza tra la soluzione ottenuta con il comando Matlab e quella esatta $\|u - x\|$ (dovrebbe risultare circa `1.3046e+08`; cosa significa?)