

Metodi Numerici per il Calcolo

Esercitazione 8:
 $A\mathbf{x} = \mathbf{b}$ e Fattorizzazione LU
A.A.2024/25

Scaricare dalla pagina web del corso l'archivio matlab_mnc2425_8.zip e scompattarlo nella propria home directory. Verrà creata una cartella con lo stesso nome contenente alcuni semplici script e function Matlab/Octave. Si svolga la seguente esercitazione che ha come obiettivo sperimentare la fattorizzazione LU di una matrice e la soluzione di sistemi lineari.

A. Soluzione di Sistemi Lineari

Nella cartella sono presenti alcune function che se richiamate restituiscono una matrice $n \times n$ non singolare che può essere utilizzata come matrice test per un sistema lineare $A\mathbf{x} = \mathbf{b}$. Come metodologia di lavoro si proceda definendo il vettore $\mathbf{x} \in \mathbb{R}^n$ (per esempio $\mathbf{x} = (1, 1, \dots, 1)^T$) e si determini \mathbf{b} affinché la soluzione del sistema sia il vettore \mathbf{x} , così da conoscerne la soluzione esatta.

1. Function `lu` di Matlab/Octave

Completare lo script `main_linsys.m` che definito un sistema lineare $A\mathbf{x} = \mathbf{b}$, lo risolve nei due seguenti modi:

- utilizzando l'operatore "left-division" di Matlab/Octave;
- utilizzando la function di Matlab/Octave `lu` che implementa la fattorizzazione di Gauss con scambio delle righe e perno massimo. Più precisamente siamo interessati alla chiamata:

$$[L, U, P] = \text{lu}(A)$$

(vedere `help lu`); quindi si usino le function `lsolve.m` e `usolve.m` presenti nella cartella per risolvere i sistemi

$$\begin{aligned} L\mathbf{y} &= P\mathbf{b} \\ U\mathbf{x} &= \mathbf{y}. \end{aligned}$$

2. Sulla stabilità della fattorizzazione LU

Si completi lo script `main_lufact.m` in modo che richiami la function `lu` di Matlab come nell'esercizio A.1 (fattorizzazione LU di una matrice con scambio delle righe e perno massimo) e verifichi che:

$$\max |\ell_{i,j}| \leq 1, \quad \max |u_{i,j}| \leq 2^{n-1} \max |a_{i,j}|$$

dove $\ell_{i,j}$ e $u_{i,j}$ sono gli elementi delle matrici L e U determinate. Si stampi una tabella con i seguenti valori per le matrici di esempio `mat_k`, $k=2, 3, 4, 5$ di dimensioni $n = 5, 10, 50$

mat_k	$n \times n$	$\max \ell_{i,j} $	$\max u_{i,j} $	$2^{n-1} \max a_{i,j} $
---------	--------------	---------------------	------------------	--------------------------

Si realizzi uno script `main_qrfact.m` che richiami la built-in function Matlab `qr`, del tutto simile allo script `main_lufact.m`, per confrontare i valori degli elementi delle matrici Q ed R ottenuti nella fattorizzazione QR ; si verifichi inoltre che

$$\max |q_{i,j}| \leq 1, \quad \max |r_{i,j}| \leq \sqrt{n} \max |a_{i,j}|.$$

3. Esempio sulla Stabilità della fattorizzazione LU

Le function `LUsimple.m` e `LUsimple_solve.m` implementano la fattorizzazione LU senza scambio di righe e soluzione dei due sistemi triangolare inferiore e superiore. Si completi lo script `main_stab_lufact.m` per risolvere il seguente sistema lineare $A\mathbf{x} = \mathbf{b}$ con

$$A = \begin{pmatrix} 10^{-20} & 1 \\ 1 & 1 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

utilizzando sia la fattorizzazione con scambio delle righe e perno massimo (function `lu` di Matlab) che la fattorizzazione senza scambio di righe (function `LUsimple`). Si confrontino le soluzioni ottenute nei due casi.

4. Esempio sul condizionamento di un sistema lineare

Si consideri la matrice H di Hilbert $n \times n$ (function Matlab `hilb`, vedi l'`help`) Si calcoli \mathbf{b} in modo che $H\mathbf{x} = \mathbf{b}$, dove $\mathbf{x} = (1, 1, \dots, 1)^T$. Si aggiunga a \mathbf{b} una perturbazione

$$\delta\mathbf{b}_p = 10^{-p} \text{rand}(n, 1).$$

Si risolva il sistema $H\tilde{\mathbf{x}}_p = \mathbf{b} + \delta\mathbf{b}_p$, per $p = 1, \dots, 5$ e si stampi per ogni valore di p la seguente quantità:

$$K_p = \frac{\|\mathbf{x} - \tilde{\mathbf{x}}_p\|}{\|\mathbf{x}\|} \frac{\|\mathbf{b}\|}{\|\delta\mathbf{b}_p\|}.$$

I valori K_p così ottenuti sono il valore effettivo sperimentale del numero di condizione della matrice H . Verificare che per ogni p sia

$$K_p \leq \text{cond}(H).$$

Lo script `main_hilb.m` implementa già quanto detto; eseguire per differenti dimensioni $n \times n$ e analizzare i risultati.

B. Riproduzione di disegni

I file `sppbez_flag_germ.p` e `sppbez_flag_ceco.p` sono due script Matlab in 'proprietary mode', ossia si possono eseguire in ambiente Matlab, ma non si

può accedere ai loro sorgenti. Eseguiti verrà presentato un semplice disegno (vedi Figura 2) che si vuole replicare con curve di Bézier a tratti, ottenute come join di curve di Bézier di interpolazione e/o modellazione. Si colorino quindi le regioni delimitate da tali curve per replicare, come richiesto, il disegno. Gli script si chiamano rispettivamente `sppbez_interp2d_germ.m` e `sppbez_interp2d_ceco.m`.

(Sugg. Per campionare punti della geometria disegno (vedi Figura 1) ci si avvicini ad un punto con il mouse; Matlab mostrerà le coordinate di quel punto. Dopo avere acquisito i punti necessari si proceda alla loro interpolazione e/o modellazione con curve di Bézier; si proceda poi al join di tali curve di Bézier per ottenere curve di Bézier a tratti che definiscano il bordo delle regioni colorate).

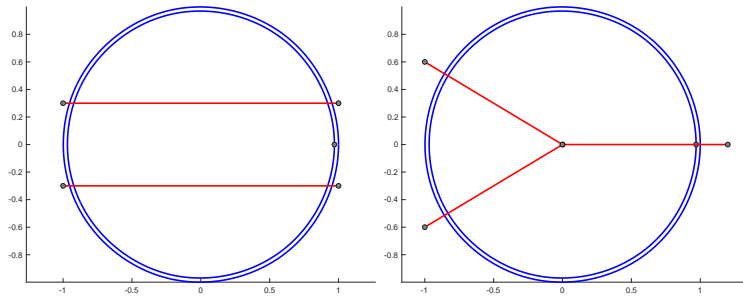


Figura 1: Figure 1 di output dei codici `sppbez_flag_germ.p` e `sppbez_flag_ceco.p`

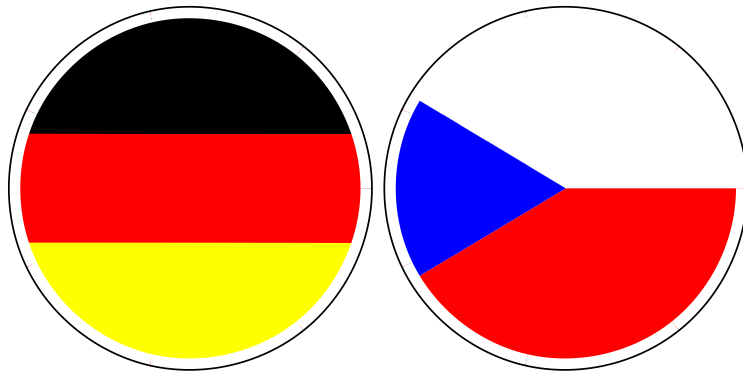


Figura 2: Figure 2 di output dei codici `sppbez_flag_germ.p` e `sppbez_flag_ceco.p`