Esercizi Elaborato (versione 2023-12-28)

N.B.: curare attentamente la stesura delle function Matlab, che devono essere allegate all'elaborato in un file `.zip`

Esecizio 1. Dimostrare che:

$$\frac{25f(x) - 48f(x-h) + 36f(x-2h) - 16f(x-3h) + 3f(x-4h)}{12h} = f'(x) + O(h^4).$$

Esercizio 2. La funzione

$$f(x) = 1 + x^2 + \frac{\log(|3(1-x)+1|)}{80}, \qquad x \in [1, 5/3],$$

ha un asintoto in $x = 4/3$, in cui tende a $-\infty$. Graficarla in Matlab, utilizzando

$$\boldsymbol{x} = \texttt{linspace}(1, 5/3, 100001)$$

(in modo che il floating di $4/3$ sia contenuto in $\boldsymbol{x}$) e vedere dove si ottiene il minimo. Commentare i risultati ottenuti.

Esercizio 3. Spiegare in modo esaustivo il fenomeno della *cancellazione numerica*. Fare un esempio che la illustri, spiegandone i dettagli.

Esercizio 4. Scrivere una *function* Matlab che implementi in modo efficiente il metodo di bisezione.

Esercizio 5. Scrivere *function* Matlab distinte che implementino efficientemente i metodi di Newton e delle secanti per la ricerca degli zeri di una funzione $f(x)$.

Esercizio 6. Utilizzare le *function* dei precedenti esercizi per determinare una approssimazione della radice della funzione

$$f(x) = e^x - \cos\, x,$$

per $tol = 10^{-3}, 10^{-6}, 10^{-9}, 10^{-12}$, partendo da $x_0 = 1$ (e $x_1 = 0.9$ per il metodo delle secanti). Per il metodo di bisezione, usare l'intervallo di confidenza iniziale $[-0.1, 1]$. Tabulare i risultati, in modo da confrontare il costo computazionale di ciascun metodo.

Esercizio 7. Applicare gli stessi metodi e dati del precedente esercizio, insieme al metodo di Newton modificato, per la funzione

$$f(x) = e^x - \cos x + \sin x - x(x+2).$$

Tabulare i risultati, in modo da confrontare il costo computazionale e l'accuratezza di ciascun metodo. Commentare i risultati ottenuti.

Esercizio 8. Scrivere una *function* Matlab,

`function x = mialu(A,b)`

che, data in ingresso una matrice A ed un vettore $\boldsymbol{b}$, calcoli la soluzione del sistema lineare $A\boldsymbol{x} = \boldsymbol{b}$ con il metodo di fattorizzazione LU con *pivoting* parziale. Curare particolarmente la scrittura e l'efficienza della *function*, e validarla su un congruo numero di esempi significativi, che evidenzino tutti i suoi possibili *output*.

Esercizio 9. Scrivere una *function* Matlab,

`function x = mialdl(A,b)`

che, dati in ingresso una matrice sdp A ed un vettore $\boldsymbol{b}$, calcoli la soluzione del corrispondente sistema lineare utilizzando la fattorizzazione $LDL^\top$. Curare particolarmente la scrittura e l'efficienza della *function*, e validarla su un congruo numero di esempi significativi, che evidenzino tutti i suoi possibili *output*.

Esercizio 10. Scrivere una *function* Matlab,

`function [x,nr] = miaqr(A,b)`

che, data in ingresso la matrice A $m \times n$, con $m \geq n = \text{rank}(A)$, ed un vettore $\boldsymbol{b}$ di lunghezza m, calcoli la soluzione del sistema lineare $A\boldsymbol{x} = \boldsymbol{b}$ nel senso dei minimi quadrati e, inoltre, la norma, nr, del corrispondente vettore residuo. Curare particolarmente la scrittura e l'efficienza della *function*, e validarla su un congruo numero di esempi significativi, che evidenzino tutti i suoi possibili *output*.

Esercizio 11. Risolvere i sistemi lineari, di dimensione n,

$$A_n \boldsymbol{x}_n = \boldsymbol{b}_n, \qquad n = 1, \ldots, 15,$$

in cui

$$A_n = \begin{pmatrix} 1 & 1 & \ldots & \ldots & 1 \\ 10 & \ddots & \ddots & & \vdots \\ 10^2 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 1 & 1 \\ 10^{n-1} & \ldots & 10^2 & 10 & 1 \end{pmatrix} \in \mathbb{R}^{n \times n}, \qquad \boldsymbol{b}_n = \begin{pmatrix} n - 1 + \frac{10^1 - 1}{9} \\ n - 2 + \frac{10^2 - 1}{9} \\ n - 3 + \frac{10^3 - 1}{9} \\ \vdots \\ 0 + \frac{10^n - 1}{9} \end{pmatrix} \in \mathbb{R}^n,$$

la cui soluzione è il vettore $\boldsymbol{x}_n = (1, \ldots, 1)^\top \in \mathbb{R}^n$, utilizzando la *function* `mialu`. Tabulare e commentare l'accuratezza dei risultati ottenuti, dandone spiegazione esaustiva.

Esercizio 12. Fattorizzare, utilizzando la *function* `mialdlt`, le matrici sdp

$$A_n = \begin{pmatrix} n & -1 & \ldots & -1 \\ -1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & -1 \\ -1 & \ldots & -1 & n \end{pmatrix} \in \mathbb{R}^{n \times n}, \qquad n = 1, \ldots, 100.$$

Graficare, in un unico grafico, gli elementi diagonali del fattore D, rispetto all'indice diagonale.

Esercizio 13. Utilizzare la *function* `miaqr` per risolvere, nel senso dei minimi quadrati, il sistema lineare sovradeterminato

$$A\boldsymbol{x} = \boldsymbol{b},$$

in cui

$$A = \begin{pmatrix} 7 & 2 & 1 \\ 8 & 7 & 8 \\ 7 & 0 & 7 \\ 4 & 3 & 3 \\ 7 & 0 & 10 \end{pmatrix}, \qquad \boldsymbol{b} = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{pmatrix},$$

dove viene minimizzata la seguente norma *pesata* del residuo $\boldsymbol{r} = (r_1, \ldots, r_5)^T$:

$$\rho_\omega^2 := \sum_{i=1}^{5} \omega_i r_i^2,$$

con

$$\omega_1 = \omega_2 = 0.5, \quad \omega_3 = .75, \quad \omega_4 = \omega_5 = 0.25.$$

Dettagliare l'intero procedimento, calcolando, in uscita, anche ρ_ω.

Esercizio 14. Scrivere una *function* Matlab,

```
[x,nit] = newton(fun,x0,tol,maxit)
```

che implementi efficientemente il metodo di Newton per risolvere sistemi di equazioni nonlineari. Curare particolarmente il criterio di arresto. La seconda variabile, se specificata, ritorna il numero di iterazioni eseguite. Prevedere opportuni valori di *default* per gli ultimi due parametri di ingresso (rispettivamente, la tolleranza per il criterio di arresto, ed il massimo numero di iterazioni). La function `fun` deve avere sintassi: `[f,jacobian]=fun(x)`, se il sistema da risolvere è `f(x)=0`.

Esercizio 15. Usare la *function* del precedente esercizio per risolvere, a partire dal vettore iniziale nullo, il sistema nonlineare derivante dalla determinazione del punto stazionario della funzione:

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top Q\boldsymbol{x} + \boldsymbol{e}^\top \left[\cos(\alpha\boldsymbol{x}) + \beta\exp(-\boldsymbol{x})\right], \qquad \boldsymbol{e} = (1, \ldots, 1)^\top \in \mathbb{R}^{50},$$

$$Q = \begin{pmatrix} 4 & 1 & & \\ 1 & \ddots & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 4 \end{pmatrix} \in \mathbb{R}^{50\times 50}, \qquad \alpha = 2, \quad \beta = -1.1,$$

utilizzando tolleranze `tol = 1e-3, 1e-8, 1e-13` (le *function* cos e exp sono da intendersi in modo vettoriale). Graficare la soluzione e tabulare in modo conveniente i risultati ottenuti.