

PROVA DI LABORATORIO DI CALCOLO NUMERICO
INGEGNERIA MECCANICA - MATRICOLE PARI - AA 2017/18
Proff. Stefano De Marchi, Emma Perracchione
Padova, 25 Gennaio 2019

- Il candidato dovrà produrre uno script `.m` per **ogni** esercizio.
- commentare **bene** gli scripts usando il comando `%`.
- al termine della prova lasciare tutti i files nella propria cartella **home**.
- **vietato usare libri, appunti e naturalmente il cellulare**

ESERCIZI

1. Si consideri il problema di approssimare con il metodo di Newton gli zeri di funzioni. Data la seguente funzione:

$$f(x) = \left(3x - \frac{1}{2}\right)^2 e^{\left(x^4 - \frac{x}{6}\right)}.$$

si scriva uno script `Esercizio1.m` che esegua le seguenti istruzioni.

- (a) Detto z lo zero della funzione f , assegnare alla variabile `m` il valore della molteplicità di z . Plottare la funzione nell'intervallo $[0, 2]$ e cercare un'opportuna condizione iniziale `x0` per il metodo di Newton.
- (b) Calcolare inoltre la derivata di f e definirla nello script chiamandola `fp`. Calcolare quindi il valore approssimato `x1` dello zero z con il metodo di Newton `Newton.m`, presente nella propria cartella di lavoro. Calcolare inoltre in numero di iterazioni `iter1` necessarie affinché il metodo converga allo zero con tolleranza `tol=10-6`. Fissato come numero massimo di iterazioni `maxiter=100`, la chiamata alla function sarà

`[x1,iter1] = Newton(f,fp,x0,tol,maxiter);`

- (c) Scrivere una function `NewtonMod.m` che implementi il metodo di Newton modificato per una generica radice di molteplicità `m`.
- (d) Ripetere il punto (b) con la function `NewtonMod.m`. Chiamare l'approssimazione trovata `x2` e indicare con `iter2` il numero di iterazioni necessarie affinché il metodo di Newton modificato converga. In questo caso la chiamata alla function è

`[x2,iter2] = NewtonMod(f,fp,x0,tol,maxiter,m);`

- (e) Visualizzare/stampare a video `iter1` ed `iter2`. Dire quale metodo converge più velocemente e commentare adeguatamente i risultati.

2. Creare uno script denominato **Esercizio2.m** e copiare le seguenti righe di codice:

```
x = linspace(-0.2,2,3210);  
y = sin(-sqrt(2)*x)+1/9*x;  
yy = y+(10^(-1)).*randn(size(x));
```

A questo punto si trovano 3210 punti che stanno all'incirca su una funzione f . Essi infatti sono stati perturbati con un fattore random per simulare dati reali affetti da errore. Sempre sullo script **Esercizio2.m** eseguire le seguenti istruzioni.

- (a) Plottare i punti (x,yy) e dire qual è la funzione f sulla quale giacciono i punti (entro un certo errore random). Sulla stessa finestra grafica plottare in rosso anche la funzione f .
- (b) Trovare e plottare (in verde) l'approssimante ai minimi quadrati di grado $n = 6$. Usare i comandi **polyfit** e **polyval**. Suggerimento: usare il comando **polyval** come segue:

```
z = polyval(coeff,x);
```

dove **coeff** è stato trovato con **polyfit**.

- (c) Valutare tale approssimante nei punti $v = -0.12345; 0.12345; 1.2345$ e stampare a video il valore ottenuto.