

Metodi Numerici per il Calcolo

Esercitazione 4:
Script, Function e Funzioni Polinomiali
A.A.2022/23

Scaricare dalla pagina web del corso l'archivio `matlab_mnc2223_4.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 approfondire la propria conoscenza dell'ambiente Matlab capendo, usando e modificando piccoli script e function sulla rappresentazione grafica di dati e funzioni polinomiali.

A. Function plot()

Dopo aver visto le potenzialità della grafica Matlab, realizzare lo script `sgrafici1.m` che rappresenti graficamente in una stessa **figure** le due funzioni

$$y = \cosh(x), \quad y = 0.5 \exp(x), \quad \text{con} \quad x \in [-2, 2],$$

utilizzando tipi di linee e/o colori differenti, una legenda, un titolo, etichette sugli assi coordinati e una griglia.

B. Ancora sulla Function plot()

Realizzare lo script `sgrafici2.m` per rappresentare graficamente le seguenti funzioni, ciascuna in una differente **figure**:

$$y = x^3 - 4x \quad x \in [-3, 3],$$

$$y = 0.2x^4 + x^3 - 0.4x^2 - 3x + 1 \quad x \in [-6, 6],$$

$$y = 3 \cos(2x) - 2 \cos(x) \quad x \in [0, 2\pi],$$

$$y = \frac{\sin(2x)}{x} \quad x \in [-6\pi, 6\pi].$$

C. Valutazione funzione polinomiale e derivata prima

Si realizzi uno script che utilizzi le function Matlab `polyval` e `polyder` per valutare un polinomio e la sua derivata prima in corrispondenza di una discretizzazione dell'intervallo di definizione. Lo script si chiami `spoly_eval_der.m`, consideri i polinomi dell'esercizio precedente e li rappresenti graficamente insieme alle loro derivate prime.

D. Errore Algoritmico nella Valutazione polinomiale

Completare la function `poly_eval.m` che implementi l'algoritmo di Horner per valutare un polinomio in corrispondenza di un vettore di punti. Si utilizzi lo script `spoly_eval.m` che richiama tale function e valuta il seguente polinomio

$$p(x) = x^3 - 39x^2 + 504x - 2158 \quad x \in [10, 16]$$

sia in precisione single che double. Considerando il risultato ottenuto in precisione double come esatto, si calcoli e rappresenti graficamente l'errore algoritmico.

Si analizzi il risultato e si individuino in corrispondenza di quali punti si hanno i valori di maggior errore algoritmico; si dia una spiegazione a quanto trovato. (**Sugg.** si valuti il polinomio nell'intervallo indicato in punti che siano numeri finiti; questo, per il fatto che i coefficienti del polinomio sono numeri interi, farà sì che gli eventuali errori siano di tipo algoritmico).

E. Errore Inerente nella Valutazione polinomiale

Si utilizzi lo script `spoly_eval2.m` che richiama la function `poly_eval.m` e valuta il seguente polinomio

$$p(x) = -x + 100 \quad x \in [100, 101]$$

in precisione double, ma sia a partire da dati double che convertiti in single. Considerando i risultati ottenuti a partire dai dati in double come esatti e quelli ottenuti a partire dai dati in single come quelli calcolati, si determina e rappresenta graficamente l'errore inerente. (**Sugg.** si valuti il polinomio nell'intervallo indicato in punti che siano numeri approssimati a finiti; si modifichino poi i coefficienti del polinomio per essere numeri approssimati a finiti. Questo, per farà sì che gli eventuali errori siano di tipo inerente).

F. Sviluppo di Taylor ed Errore Analitico

La function `taylor_sin` implementa lo sviluppo polinomiale di Taylor della funzione `sin(x)` centrato in un punto x_0 e di grado n . Analizzare lo script per capire cosa è implementato, quindi eseguirlo più volte per differenti punti x_0 e gradi n al fine di comprendere il comportamento dell'approssimazione di Taylor grazie alla rappresentazione grafica delle funzioni e alla stampa dell'errore.

G. Basi polinomiali e loro rappresentazione grafica

La function `base_plot.m` implementa la visualizzazione grafica di differenti basi polinomiali definite su un intervallo $[a, b]$. In particolare permette la rappresentazione della base canonica, della base di Bernstein e della base con centro

$$\{1, (x - c), (x - c)^2, \dots, (x - c)^n\}$$

con c un punto dell'intervallo di definizione. Dopo aver visionato il codice e le function ivi richiamate, si modifichi la function `base_plot.m` per visualizzare una base alla volta e di ogni base una funzione alla volta. Si modifichi il codice della base con centro in modo che il centro c sia a scelta l'estremo sinistro a , l'estremo destro b o il punto medio $(a+b)/2$ dell'intervallo.

H. Approssimazione di Weierstrass e Bernstein

Si realizzi uno script `sapprox.unif.m` che considerata una funzione continua $y = f(x)$ per $x \in [a, b]$, la approssimi con un polinomio nella base di Bernstein di grado n con coefficienti i valori

$$c_i = f(\xi_i) \quad i = 0, \dots, n$$

dove

$$\xi_i = a + \frac{i}{n}(b - a).$$

Si sperimenti questo tipo di approssimazione sulle funzioni degli esercizi A. e B. e per $n \rightarrow \infty$. Si rappresenti graficamente sia la funzione che il polinomio approssimante e in una seconda figure la funzione errore assoluto. Cosa si osserva all'aumentare di n .

I. Esercizio di verifica (su valutazione polinomiale e derivata prima con Ruffini)

Completare la function `poly_eval_der` che implementa l'algoritmo di Ruffini per valutare un polinomio e la sua derivata prima in corrispondenza di un vettore di punti (Sugg. ci si riferisca al pseudocodice presente sulla dispensa del corso). Si realizzi uno script `spoly_eval_der2.m` simile a quello dell'esercizio C. che richiami la function `poly_eval_der` per valutare e quindi rappresentare graficamente gli stessi polinomi e le loro derivate prime.

L. Esercizio di verifica (su algoritmo di de Casteljau)

Si completi la function `decast.m` per implementare l'algoritmo di de Casteljau per valutare un polinomio nella base di Bernstein. (Sugg. ci si riferisca al codice Matlab presente sulla dispensa del corso). Realizzare poi uno script `sdecast.m` per valutare e rappresentare graficamente i polinomi nella base di Bernstein i cui coefficienti sono definiti nella function `def_pol.m`.