

Metodi Numerici per il Calcolo

Esercitazione 7: Equazioni non lineari

A.A.2023/24

Scaricare dalla pagina web del corso l'archivio `matlab_mnc2324_7.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 quella di sperimentare metodi per determinare le radici di equazioni non lineari.

A. Radici di Equazioni

Si considerino le seguenti funzioni test di cui si vogliono determinare gli zeri o radici dell'equazione associata (nella cartella sono presenti le function che le implementano insieme alle function delle derivate prime):

- zfun01** $f(x) = (4x - 7)/(x - 2) \quad x \in [1, 1.9]$
- zfun02** $f(x) = 1 + 3/x^2 - 4/x^3 \quad x \in [0.5, 6]$
- zfun03** $f(x) = 1 - 2.5x + 3x^2 - 3x^3 + 2x^4 - 0.5x^5 \quad x \in [0, 3]$
- zfun04** $f(x) = x^n - 2 \quad x \in [0, 2] \quad n = 2, 3, 4$
- zfun05** $f(x) = x^3 - 3x + 2 \quad x \in [-2.5, 2]$
- zfun06** $f(x) = 1/x - 2 \quad x \in (0, 4]$

1. Metodo di bisezione

La function `main_bisez.m` fa uso della function `bisez.m` che implementa il metodo di bisezione. Viene effettuato il grafico della funzione $f(x)$ così da poter localizzare visivamente gli zeri della funzione nell'intervallo e poter definire l'intervallo di innesco del metodo.

- Analizzare i codici e sperimentare con alcune funzioni test la ricerca degli zeri modificando la tolleranza di arresto e l'intervallo di innesco.
- Modificare le function `main_bisez.m` e `bisez.m` affinché producano in stampa il numero di iterazioni effettuate. Mediamente quante iterazioni vengono compiute per la tolleranza 10^{-15} ?

2. Metodi di Newton e delle secanti

Le function `main_stangmet.m` e `main_ssecmet.m` utilizzano le function `stangmet.m` e `ssecmet.m` che implementano rispettivamente i metodi di Newton e delle secanti. Viene effettuato il grafico della funzione $f(x)$ così da poter localizzare visivamente gli zeri della funzione nell'intervallo.

- Analizzare i codici, e sperimentare con alcune funzioni test la ricerca degli zeri modificando la tolleranza di arresto.

- Dopo aver analizzato il grafico di una funzione test stimare possibili iterati iniziali.

3. Metodo di Newton e ordine di convergenza

Chiamando la funzione `main_stangmet.m` con `ftrace` ad 1 si hanno tutti gli iterati calcolati dal metodo di Newton; si modifichi il codice (lo si chiami `main_stangmet_ordconv.m`) affinché produca una tabella con le seguenti informazioni:

k	x_k	$e_k = x_k - x^*$	$ e_{k+1} / e_k $	$ e_{k+1} / e_k ^2$
-----	-------	-------------------	-------------------	---------------------

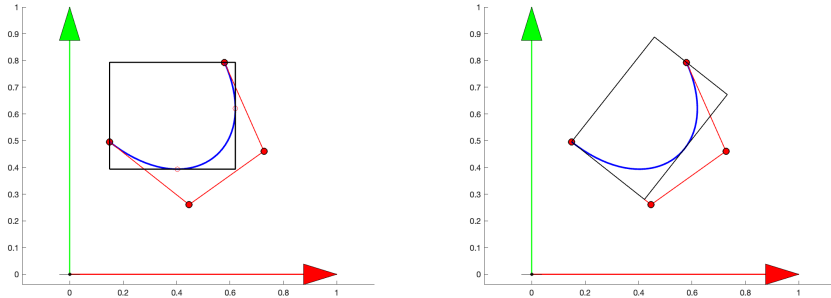
Si analizzino i risultati delle ultime due colonne nel caso di zeri semplici o multipli della funzione test.

4. Le function `bezier_clipping.m` e `lane_riesenfeld.m` implementano due differenti algoritmi per determinare tutti gli zeri di una funzione polinomiale nella base di Bernstein nell'intervallo di definizione. Si completi lo script `main_szeri_bern.m` per determinare gli zeri del polinomio `zfun05` rappresentato nella base di Bernstein. (Come si può ottenere la funzione `zfun5` nella base di Bernstein?) Si designo anche i punti di controllo della funzione.

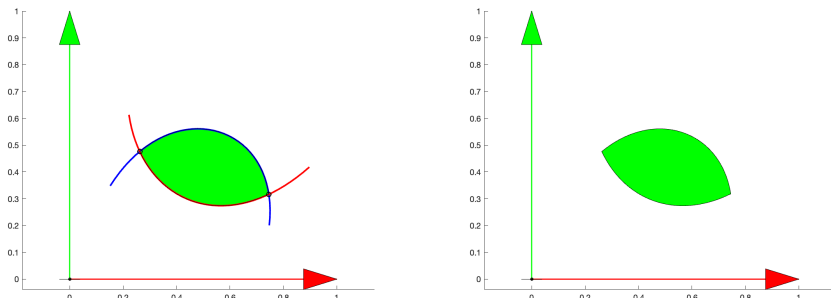
B. Interrogazione di curve 2D di Bézier

I seguenti esercizi necessitano di curve di Bézier o curve di Bézier a tratti come input. Per definirle si utilizzi l'applicazione interattiva `ppbez_design.m` del toolbox `anmglib_4.0`. Le curve progettate verranno salvate nel file di nome `ppbez_design.db` nella cartella corrente; salvare quindi una curva alla volta e rinominarla prima di salvarne un'altra.

1. Data una curva 2D di Bézier si determinino i suoi punti estremi, cioè i punti della curva a tangente verticale e orizzontale e si disegnino insieme al primo ed ultimo punto della curva. Lo script si chiami `main_bezier_estremi.m`. (Sugg. si determinino gli zeri delle derivate prime delle componenti la curva).
Dopo aver realizzato quanto richiesto si modifichi lo script affinché determini il bounding-box della curva come il più piccolo rettangolo che contiene i punti estremi e il primo ed ultimo punto della curva. Lo script si chiami `main_bezier_bbox.m`.
2. Data una curva 2D di Bézier si determini il suo tight bounding-box (vedi corso on-line su curve di Bézier). Lo script si chiami `main_bezier_tight_bbox.m`. (Sugg. si realizzi una function secondaria (`align`) che determini la trasformazione geometrica che porta il primo punto della curva nell'origine degli assi e l'ultimo punto di controllo sull'asse x ; si determini il bounding-box di questa curva aligned come fatto nell'esercizio precedente, quindi si applichi la trasformazione inversa al bounding-box trovato).



3. Si riprendano i due esercizi precedenti e si confrontino le aree dei due bounding-box per analizzare se il tight bounding-box risulta sempre più piccolo del bounding-box.
(Sugg. i due bounding box siano definiti come curve di Bézier a tratti di grado 1 e si definisca e utilizzi una function generale `curv2_ppbez_area.m` per il calcolo della loro area, sfruttando quanto visto nell'Esercitazione 6).
4. Date due curve 2D di Bézier determinare e disegnare le intersezioni. Si completi lo script `main_curve_intersect.m`.
(Sugg. si utilizzi la funzione `curv2_intersect` del toolbox `anmglib_4.0`).
Dopo aver realizzato quanto richiesto, si modifichi lo script per determinare le intersezioni fra due curve di Bézier a tratti.
5. Date due curve 2D di Bézier che si intersecano in due punti, si determini la curva di Bézier a tratti che definisce la regione chiusa fra le due e la si colori. Lo script si chiami `main_bezier_intersect.m`. (Sugg. si utilizzi la function `decast_subdiv`).
Dopo aver realizzato quanto richiesto, si modifichi lo script per gestire due curve di Bézier a tratti chiuse. Si determinino le curve di Bézier a tratti di bordo delle regioni di intersezione, unione e differenze e le si colorino.
(Sugg. si utilizzi la function `ppbezier_subdiv`).



Si osservi che le function della libreria che gestiscono curve di Bézier a tratti (suffisso `ppbezier`) possono essere utilizzate anche per curve di Bézier, a meno della function `curv2_ppbezier_load`.