Laboratorio di Programmazione e Calcolo
Canale 2
Appunti del corso

Simone Cacace e Giuseppe Visconti
Dipartimento di Matematica
Sapienza Università di Roma

Anno Accademico 2025–2026

Laboratorio – Equazioni non lineari

Esercizi

E0. Bisezione con monitoraggio dell'errore

Riprendere il codice del metodo di bisezione visto a lezione e modificarlo come segue:

- Per ogni iterazione salvare su array la stima della radice x_n e l'errore $e_n = |x_n - \alpha|$ (se α è noto) oppure l'errore stimato $e_n \approx |b_n - a_n|/2$ dove $[a_n, b_n]$ è l'intervallo corrente.
- Verificare che la tolleranza fissata dall'utente è stata raggiunta all'iterazione data dalla stima $n \geq \left\lceil \log_2 \frac{b-a}{\text{Tol}} \right\rceil$.
- Salvare i seguenti dati su file (ad esempio `bisezione.txt`): `iterazione, x_n, e_n, p_n`, dove `p_n` è la velocità sperimentale di convergenza (EOC) ed è calcolato come
$$p_n = \frac{\log(e_{n+1}/e_n)}{\log(e_n/e_{n-1})}.$$
Vedi ulteriore spiegazione dopo lista di esercizi.
- Generare due grafici con `gnuplot`:
  1. `x_n` vs `iterazione`,
  2. grafico in scala log su asse y `e_n` vs `iterazione`,
  3. `p_n` vs `iterazione`.

Funzione di esempio:

$$f(x) = \cos(x) - x \qquad (\text{radice reale } \alpha \approx 0.739085\ldots)$$

Spiegazione matematica. La bisezione mantiene un intervallo $[a_n, b_n]$ contenente la radice; a ogni iterazione la lunghezza dell'intervallo si dimezza e l'errore stimato soddisfa
$$|x_n - \alpha| \leq \frac{b_n - a_n}{2} = \frac{b_0 - a_0}{2^{n+1}}.$$

Suggerimenti di implementazione.

- Modificare la funzione bisezione scritta a lezione in modo da poter restituire l'array della successione dei punti medi x_n, l'array degli errori e_n, e il numero di iterazione n quando un criterio di arresto si verifica.
- Tenere array dinamici per `x[niter]`, `err[niter]`.
- Salvare il file e tracciare con `gnuplot`, usando `set logscale y` per settare la scala logaritmica sull'asse y.

E1. Punto fisso con monitoraggio dell'errore e confronto con bisezione

Riprendere il codice del metodo del punto fisso e modificarlo come per E0:

- Per ogni iterazione salvare x_n e $e_n = |x_n - \alpha|$ (o errore relativo).
- Calcolare e salvare i dati su file (ad esempio `puntofisso.txt`).

- Effettuare un confronto qualitativo e quantitativo con i risultati ottenuti con la bisezione per la stessa funzione:
  - Confrontare le curve `x_n` vs `iterazione`, `e_n` vs `iterazione`, `p_n` vs `iterazione`, dei due metodi.

Funzione di esempio e riscrittura per punto fisso: Partendo da $f(x) = \cos(x) - x$, possiamo riscrivere $g(x) = \cos(x)$. La radice è l'intersezione della funzione $y = \cos(x)$ con la funzione $y = x$ (soluzione $\alpha \approx 0.739085\ldots$).

Spiegazione matematica. Il metodo del punto fisso usa $x_{n+1} = g(x_n)$. Una condizione sufficiente di convergenza locale è $|g'(\alpha)| < 1$. L'ordine di convergenza è lineare se $0 < |g'(\alpha)| < 1$, più elevato al decrescere di $g'(\alpha)$.

E2. Metodo di Newton con "warm–start" tramite bisezione

- Implementare l'iterazione

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

  prendendo spunto dal pseudoalgoritmo visto a lezione e presente sulle dispense.

Funzione di esempio

$$f(x) = x^3 - 2x + 2, \qquad f'(x) = 3x^2 - 2.$$

Si osserva che $f(-2) = -2 < 0$ e $f(-1) = 3 > 0$, quindi esiste una radice reale $\alpha \in [-2, -1]$. Il punto iniziale patologico proposto è $x_0 = 0$: applicando Newton si ottiene il ciclo $0 \leftrightarrow 1$ e la procedura non converge alla radice nell'intervallo $[-2, -1]$.

- **Mostrare il fallimento senza warm–start.** Partire da $x_0 = 0$, registrare le iterazioni e mostrare il ciclo o la mancata convergenza.
- **Warm–start con bisezione.** Brackettare la radice con $a = -2$, $b = -1$. Eseguire un numero piccolo m (ad esempio 3 o 5) di passi di bisezione per ottenere il punto iniziale warm:

$$x_0^{\text{warm}} = \text{midpoint dell'ultimo intervallo.}$$

  Avviare Newton da x_0^{warm} e confrontare la convergenza.

Salvataggio dati, output richiesto e grafici

- Per ogni iterazione salvare: `n, x_n, e_n, |f(x_n)|`.
- Stampa tabelle di iterazione a video.
- Calcolo della velocità di convergenza sperimentale p_n e salvataggio su file dei dati: `n, x_n, e_n, |f(x_n)|, p_n`.
- Grafico in scala log su asse y di `e_n` vs `iterazione` e di `p_n` vs `iterazione`.

Spiegazione matematica. Se $f \in C^2$ e $f'(\alpha) \neq 0$, Newton converge localmente con velocità di convergenza 2.

Calcolo dell'EOC (velocità sperimentale di convergenza). Per sequenze di errori $e_n = |x_n - \alpha|$ (con $e_n > 0$), si stima il tasso locale di convergenza tramite

$$p_n = \frac{\log(e_{n+1}/e_n)}{\log(e_n/e_{n-1})}.$$

Su un plot `EOC vs iterazione` si vede la stabilizzazione di p_n verso la velocità teorica (1 per bisezione/punto fisso lineare, 2 per Newton quadratico).

Output e plotting con gnuplot. Per ogni esercizio salvare i dati in file testuali. Esempi di comandi `gnuplot` utili:

```
# plot errore in scala log su asse y
set logscale y
set xlabel "iter"
set ylabel "errore"
plot "bisezione.txt" using 1:3 with lines title "bisezione"

# plot EOC vs iter:
set xlabel "iter"
set ylabel "EOC"
plot "bisezione.txt" using 1:4 with linespoints title "EOC"

# salvare in PNG
set terminal pngcairo size 900,600
set output "errore.png"
replot
set output
```