

Metodi Numerici per il Calcolo

Esercitazione 5: Interpolazione Polinomiale

A.A.2022/23

Scaricare dalla pagina web del corso l'archivio matlab_mnc2223.5.zip e scom-
pattarlo 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 realizzare e sperimentare
l'interpolazione polinomiale di dati e funzioni.

A. Interpolazione polinomiale di dati nella forma di Newton

Si completi lo script `spolint_newt_dati.m` per l'interpolazione polinomiale con
base di Newton del set di dati `dataset1.txt`. Lo script faccia le seguenti cose:

- legga il file di dati;
- calcoli la matrice N per il sistema lineare $N\mathbf{c} = \mathbf{y}$;
- risolva il sistema lineare, cioè trovi i coefficienti \mathbf{c} del polinomio interpo-
lante nella base di Newton;
- faccia il grafico dei punti dati e del polinomio interpolante (valutazione su
un insieme di punti dell'intervallo).

Sugg. si utilizzino le function `newton.m`, `lsolve.m` e `newtval.m` presenti nella
cartella.

B. Interpolazione polinomiale di dati nella forma di Lagrange

Si completi lo script `spolint_lagr_dati.m` (simile al precedente), ma che utilizzi
la base di Lagrange. Sugg. per valutare il polinomio interpolante nella base di
Lagrange si utilizzi la function `lagrval2.m` presente nella cartella.

C. Interpolazione polinomiale di funzione

Si completi lo script `spolint_lagr_fun.m` per implementare l'interpolazione po-
linomiale di grado n di una funzione $f(x)$, $x \in [a, b]$ a partire da $(x_i, f(x_i))_{i=0,\dots,n}$
utilizzando la base di Lagrange:

- 1 prevedere due differenti set di punti x_i di interpolazione (equispaziati e di
Chebyshev; si veda la function `chebyshev`);
- 2 si rappresentino graficamente: la funzione test, i punti $(x_i, f(x_i))_{i=0,\dots,n}$
di interpolazione e la funzione polinomiale interpolante;

- 3 calcolare e stampare una stima del max. dell'errore di interpolazione in valore assoluto

$$\max_{x \in [a,b]} |f(x) - p(x)|$$

utilizzando i valori calcolati della funzione e dell'interpolante.

- 4 visionare il grafico dell'errore di interpolazione in scala logaritmica.

Si consideri la funzione test di Runge (function `runge.m`):

$$f(x) = 1/(1 + x^2) \quad x \in [-5, 5].$$

D. Sulla convergenza dell'interpolante polinomiale

Nella cartella sono presenti le seguenti funzioni test:

$$\begin{aligned} \text{fun1.m} \quad f(x) &= \sin(x) - \sin(2x) \quad x \in [-\pi, \pi] \\ \text{fun2.m} \quad f(x) &= \begin{cases} 0.5 & \text{se } x \geq 0 \\ -0.5 & \text{se } x < 0 \end{cases} \quad x \in [-2, 2] \\ \text{fun3.m} \quad f(x) &= e^x \quad x \in [-2, 1]. \end{aligned}$$

Si duplich lo script dell'esercizio precedente (lo si chiama `polint_lagr_fun.m`) per sperimentare l'interpolazione di queste funzioni test all'aumentare del grado n e al variare della distribuzione di punti (equispaziati e punti di Chebyshev). Lo si modifichi per trasformarlo in una funzione con i seguenti parametri:

```
polint_lagr_fun(ifun,n,tip)
```

dove `ifun` sia l'indice di una funzione test, `n` il grado polinomiale, `tip` la tipologia di punti. Per ogni funzione test eseguire il codice più volte con differenti valori del grado e tipo di distribuzione di punti; fare delle considerazioni sulla convergenza dell'interpolante alla funzione.

Suggerimenti:

- le deduzioni sulla convergenza si traggano dai grafici e dal max. dell'errore assoluto nell'intervallo di definizione;
- redigere una tabella per ogni funzione test, con gli errori max. di interpolazione ottenuti all'aumentare del grado n sia per punti equispaziati che di Chebyshev.
- rafforzare le proprie deduzioni provando alcune fra le seguenti ulteriori funzioni test:

$$\begin{aligned} \text{fun4.m} \quad f(x) &= |x| \quad x \in [-1, 1] \\ \text{polfun1.m} \quad p(x) &= 1 + x/2 + x^2/6 + x^3/24 + x^4/120 \quad x \in [-1, 3.5] \\ \text{polfun4.m} \quad p(x) &= (x + 5/4)(x + 3/4)(x + 1/4)(x - 5/4)(x - 3/4)(x - 1/4) \\ &\quad x \in [-1.25, 1.25] \\ \text{fun5.m} \quad f(x) &= e^x / \cos(x) \quad x \in [-1, 1] \\ \text{polfun5.m} \quad f(x) &= |x| + 0.5x - x^2 \quad x \in [-1, 1]. \end{aligned}$$

E. Condizionamento del problema di interpolazione polinomiale

Si consideri lo script `scond_interp.m` in cui viene calcolata una stima del numero di condizione del problema di interpolazione polinomiale data da:

$$C_{Int}(p(x)) = \sum_{i=0}^n |L_{i,n}(x)| \quad x \in [a, b]$$

e dalla costante di Lebesgue associata

$$\Lambda_n = \max_{x \in [a, b]} \{C_{Int}(p(x))\}.$$

Fra i problemi di interpolazione di funzione provati e per cui si è osservata la convergenza dell'interpolante (errore di interpolazione piccolo o nullo), vedere se alcuni presentavano Errore Inerente grande.

F. Sugli errori numerici di interpolazione polinomiale

Sulla base dello script `spolint_lagr_fun.m` si realizzi lo script `spolint_compare.m` che determini l'interpolante di una funzione test usando sia la forma di Lagrange che di Newton al fine di confrontare i risultati ottenuti con i due metodi.

Si consideri come funzione test una funzione polinomiale così che l'errore analitico o di interpolazione risulti nullo (perché?). In questa situazione, eventuali errori finali saranno solo di tipo numerico ed imputabili al metodo (errore algoritmico) o ai dati e al problema (errore inerente). Visionare graficamente l'errore assoluto fra l'interpolante ottenuto con un metodo e la funzione test; confrontare gli errori ottenuti con la forma di Lagrange e con la forma di Newton.

G. Esercizio di verifica (su errore analitico, inerente e algoritmico)

Realizzare uno script di nome `serr_polint.m` che interpoli la seguente funzione test:

$$f(x) = e^x / \cos(x) \quad x \in [-1, 1] \quad \text{fun5.m}$$

nella forma di Lagrange sia su punti equispaziati che di Chebyshev e per gradi dispari `n=1:2:70`. Per ogni distribuzione di punti si preveda un grafico in scala logaritmica con i valori $\max_{x \in [a, b]} |f(x) - p(x)|$ per i differenti interpolanti. Si commentino i risultati ottenuti.

Successivamente si modifichi lo script affinché utilizzi gradi pari `n=2:2:70` ed infine tutti i gradi `n=1:70`; si notano comportamenti particolari?

H. Esercizio di verifica (su interpolazione polinomiale di funzioni nella base di Bernstein)

Si realizzi uno script `spolint_bern_fun.m` simile a `spolint_lagr_fun.m`, ma che utilizzi la forma di Bernstein. Sugg. per definire la matrice del sistema lineare si usi la function `bernst.m` e per valutare il polinomio interpolante si utilizzi la function `decast.m` presenti nella cartella.