Metodi Numerici per il Calcolo

Esercitazione 5: Interpolazione Polinomiale

A.A.2024/25

Scaricare dalla pagina web del corso l'archivio matlab_mnc2425_5.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 quello di sperimentare l'interpolazione polinomiale di dati e funzioni, quindi di punti e curve 2D.

A. Interpolazione polinomiale

1. **Interpolazione polinomiale di dati: forma di Newton**
   Si completi lo script `spolint_newt_dati.m`, per l'interpolazione polinomiale nella base di Newton, del set di dati `dataset1.txt`. (Sugg. si utilizzino le function `newton.m`, `lsolve.m` e `newtval.m` presenti nella cartella).

2. **Interpolazione polinomiale di dati: forma di Lagrange e di Bernstein**
   Si ripeta l'esercizio precedente completando gli script `spolint_lagr_dati.m` e `spolint_bern_dati.m`, per l'interpolazione polinomiale nelle basi di Lagrange e Bernstein. Si utilizzi la function `lagrval2.m` (presente nella cartella) e le function `bernst_val.m` e `decast_val.m` entrambe del toolbox `anmglib_4.1`.

3. **Interpolazione polinomiale di funzione: base di Lagrange**
   Si completi lo script `spolint_lagr_fun.m` per implementare l'interpolazione polinomiale di grado n di una funzione $f(x)$, $x \in [a, b]$ a partire da $(x_i, f(x_i))_{i=0,\ldots,n}$ utilizzando la base di Lagrange. Prevedere due differenti set di punti x_i di interpolazione (equispaziati e di Chebyshev; si utilizzi la function `chebyshev2.m` presente nella cartella). Eseguire più volte per sperimentare l'interpolazione di funzione all'aumentare del grado n e al variare della distribuzione dei punti (equispaziati e punti di Chebyshev):

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

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

4. **Interpolazione polinomiale di funzioni: base di Bernstein**
   Si completi lo 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_val.m` e per valutare il polinomio interpolante si utilizzi la function `decast_val.m` del toolbox `anmglib_4.1`).

5. **Sulla convergenza dell'interpolante polinomiale**
   Si modifichi lo script dell'esercizio A.3 (lo si chiami `spolint_lagr_fun_test.m`) per sperimentare l'interpolazione delle seguenti funzioni test all'aumentare del grado n e al variare della distribuzione di punti (equispaziati e punti di Chebyshev).

   `fun1.m` $f(x) = \sin(x) - \sin(2x) \qquad x \in [-\pi, \pi]$

   `fun2.m` $f(x) = \begin{cases} 0.5 & se \quad x \geq 0 \\ -0.5 & se \quad x < 0 \end{cases} \qquad x \in [-2, 2]$

   `fun3.m` $f(x) = e^x \qquad x \in [-2, 1].$

   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.

B. Curve di Bézier e di Bézier a tratti di interpolazione

1. Dato un set di punti 2D lo si vuole interpolare con una curva 2D di Bézier. Si completi lo script `sbezierinterp_p2d.m` per realizzare quanto richiesto; si utilizzi la function `curv2_bezier_interp` del toolbox `anmglib_4.1`.

2. Data una curva 2D in forma parametrica (vedi function `c2_curv3_pol`) la si vuole interpolare con una curva di Bézier. Si completi lo script `sbezierinterp_curv2d.m` utilizzando come parametri di interpolazione sia punti equispaziati che punti di Chebishev. Si determini e analizzi l'errore di interpolazione all'aumentare del numero di punti con cui si campiona la curva analitica.

3. Dato un set di punti 2D (vedi file `paperino.txt`) lo si vuole interpolare con una curva di Bézier cubica a tratti (interpolazione di Hermite). Si consideri la function `curv2_ppbezierCC1_interp` del toolbox `anmglib_4.1` e la si utilizzi per ottenere quanto richiesto. (Sugg. si completi lo script `sppbezierinterp_p2d.m`)

4. Data una curva 2D in forma parametrica la si vuole interpolare con una curva di Bézier cubica a tratti C^1 (interpolazione di Hermite), campionando valori e valori di derivata prima.
   Lo script `sppbezierinterp_curve2d_circle.m` vuol far questo per la curva circle (vedi function `cp2_circle.m`), utilizzando la function del toolbox `anmglib_4.1`:

   ```
   ppP=curv2_ppbezierCC1_interp_der(Q,Q1,tpar)
   ```

Come argomenti sono previsti un array di punti 2D (`Q`), un array di vettori tangenti nei punti (`Q1`) e l'array dei valori parametrici (`tpar`); l'output consiste nella struttura della curva di Bézier a tratti di interpolazione. Si richiede di valutare l'errore fra la curva analitica e l'interpolante. Sperimentare poi lo script per interpolare altre curve in forma parametrica di cui sia nota l'espressione analitica.