Calcolo Numerico - Laurea in Matematica, a.a. 2021-2022
Esercizi di Laboratorio del 07/04/2022

Nella pagina seguente sono riportate alcuni importanti definizioni per questa esercitazione.

ESERCIZIO N.1. **Splines interpolanti di grado 3. Scelta delle condizioni agli estremi.**

1. Fai il grafico della funzione $f(x) = \sin(x)\cos(x)$, $x \in [0, 2\pi]$.
2. Per $n \in \{4, 8, 12\}$, determina la spline cubica interpolante "not-a-knot", e sovrapponine il grafico a quello di f. Ricorda che il comando Matlab `s=spline(x,f(x),t)` automaticamente usa la spline "not-a-knot", dove `x` è il vettore degli $n+1$ nodi usati nella procedura composita, `f` lo handle alla funzione, e `t` il vettore dei valori in ascissa in cui calcolare la spline per fare il grafico.
3. La chiamata a `spline` può essere modificata inserendo le condizioni di spline cubica interpolante "completa": se `d1, d2` sono i valori delle derivate risp in a ed in b, allora

   ```
   s=spline(x, [d1;f(x);d2],t)
   ```

   determina i valori della spline completa. Effettua questa modifica per

   i) $s'(a) = 1$, $s'(b) = 1$ ii) $s'(a) = -1$, $s'(b) = -1$

   Dopo aver fatto ognuno dei due grafici, commenta e giustifica i risultati ottenuti.

ESERCIZIO N.2. **Valutazione dell'errore con la spline e con le sue derivate.**
Considera la funzione di Runge, $f(x) = 1/(1 + x^2)$, $x \in [-5, 5]$.

1. Per $n = 7, 9, 11$, determina la spline interpolatoria "not-a-knot" approssimante f, e sovrapponi il suo grafico a quello di f.

   Verifica sperimentalmente, anche per n più grande, che l'ordine di convergenza è $\mathcal{O}(h^p)$ con $p = 4$, dove h è il massimo della lunghezza degli sottointervalli.
2. Per $n = 11$, il comando: `S = spline(x,f(x))` (nota il diverso uso di `spline` !) crea la struttura contenente le informazioni della spline. Usando `rho=S.coefs;` (vedi pag.2 per le spiega), e ricordando che

   $$S(z) = \rho_{i,4} + \rho_{i,3}(z - x_i) + \rho_{i,2}(z - x_i)^2 + \rho_{i,1}(z - x_i)^3, \quad z \in [x_i, x_{i+1}]$$

   (la prima colonna di ρ contiene il coeff. principale!), nota che i coefficienti delle funzioni derivate $S'(z)$, $S''(z)$, $S'''(z)$ si determinano come

   ```
   drho = [3*rho(:,1) 2*rho(:,2) rho(:,3)];     d2rho = [6*rho(:,1) 2*rho(:,2)];
   d3rho = [6*rho(:,1) ];
   ```

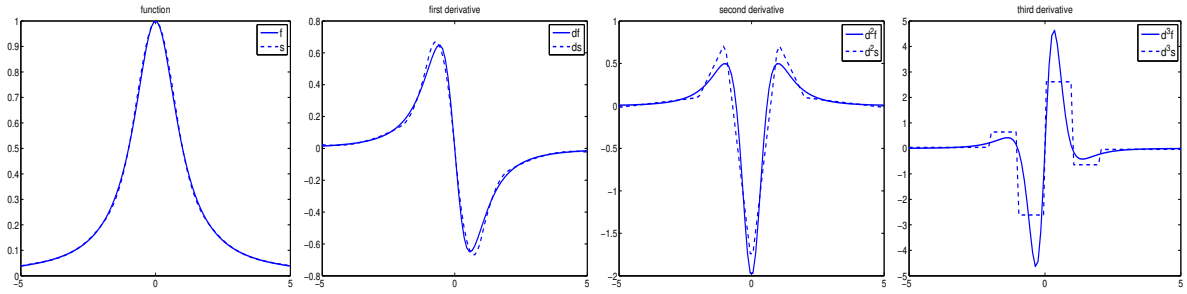
   I seguenti comandi Matlab determinano la struttura della derivata della spline, ed i suoi valori nei punti z :

   ```
   dS = mkpp(x,drho); % costruisce la struttura dai coeff.
   dSval = ppval(dS,z) % valuta la spline dS nei punti z
   ```

   Usa questi comandi per riprodurre i quattro grafici sottostanti, relativi all'approssimazione mediante splines di f, f', f'', f''' (calcola a mano le derivate di f).
3. Valuta come varia l'errore per $h \to 0$ per le derivate fino alla seconda.

ESERCIZIO N.3. (Facoltativo) **Un'applicazione.**
I dati in Tabella 1 nel sito si riferiscono ai valori di temperatura dell'aria in prossimità del suolo a 12 diverse latitudini. La prima colonna è la latitudine, la seconda le variazioni della temperatura media annua a quella latitudine (normalizzata rispetto alla concentrazione di acido carbonico).

Siamo interessati a costruire una funzione che, per i dati disponibili, fornisca una approssimazione dei valori della temperatura media per ogni possibile latitudine.



1. Riporta sullo stesso grafico i dati e l'approssimazione mediante la spline "not-a-knot".
2. Aggiungi quindi il grafico ottenuto con il polinomio di Lagrange di grado 11, mediante la funzione `get_polyn` vista in precedenti laboratori; aggiusta eventualmente gli assi con `axis([-70,80,2,5])` per una migliore visualizzazione. Confronta i risultati.
3. Stima il valore della variazione di temperatura y_{new} per la nuova latitudine $x_{new} = -45$, sia mediante spline che mediante polinomio di Lagrange. Commenta il risultato confrontandolo col valore misurato, $y_{new,mis} = 3.7$.

Alcuni comandi utili

FUNZIONE `spline`. La funzione Matlab `spline` genera la spline cubica interpolante una funzione data in nodi fissati. A seconda dei valori in input, l'output può variare.

- `ps=spline(x,f(x),t);`. La funzione determina la spline cubica interpolante $(x, f(x))$ con `x` vettore di nodi prefissati e `f` lo handle della funzione, e valuta la spline nei punti definiti in `t`. Il vettore `ps` in uscita contiene i valori della spline nei punti $t(i)$.

  Di default la funzione usa la condizione "not-a-knot". Nel caso di spline completa, ai valori nel vettore `f(x)` vengono aggiunti i vincoli corrispondenti (vedi testo della esercitazione).

- `S=spline(x,f(x));`. La funzione restituisce una struttura, contenente le informazioni locali della spline (vedi sotto per il concetto di struttura). Per esempio, per una certa scelta dei nodi `x` si può avere

```
>> S
S =
  struct with fields:

      form: 'pp'
    breaks: [0 3.1416e-01 6.2832e-01 9.4248e-01 1.2566e+00 1.5708e+00]
     coefs: [5x4 double]
    pieces: 5
     order: 4
       dim: 1
```

  dove `S.form` individua la tipologia (pp=piecewise polynomial), `S.breaks` contiene i nodi, `S.coefs` contiene i coeff (per riga) su ogni intervallo, `S.pieces` indica il numero di intervalli, `S.order` indica l'ordine dell'accuratezza, e `S.dim` indica la dimensionalità del pb (uno-dimensionale, qui).

- La funzione `unmkpp` usa la struttura `S` descritta sopra e la "spacchetta" nelle sue componenti, usando per esempio: `[forma,nodi,coefs,k,l1,dim]=unmkpp(S);`. In alternativa allo spacchettamento, si può direttamente definire, per es., nodi=S.breaks; (vedi sotto).

- La funzione `mkpp` invece crea la struttura, dati i nodi e la matrice dei coefficienti (fai `help mkpp` in Matlab per avere una spiegazione più dettagliata)
- La funzione `ppval(S,t)` valuta la spline nei punti in `t` (`ppval` riconosce la struttura spline di S).

STRUTTURE. Una struttura (`struct` data type) in matlab è un "contenitore" di oggetti di tipo diverso, chiamati *campi*, accessibili mediante una sintassi precisa.

Supponiamo di voler inserire in una struttura tutte le informazioni di un sistema lineare, di cui conosciamo A, b, n e la string `sim/nonsim` per identificare la simmetria di A. Definiamo quindi:

```
sistema=struct;
sistema.coef=A;
sistema.rhs=b;
sistema.dim=n;
sistema.strutt='sim';
```

Per esempio, se i dati corrispondo ad un sistema simmetrico 4×4, con queste definizioni a prompt di Matlab otteniamo:

```
>> sistema
sistema =
  struct with fields:
      coef: [4x4 double]
       rhs: [4x1 double]
       dim: 4
    strutt: 'sim'
```

Per accadere alla matrice è sufficiente scrivere il comando `A=sistema.coef;`, e A conterrà la matrice 4×4 memorizzata nella struttura. Analogamente per gli altri campi.