Esercitazione 10

Istruzioni trattate: `interp1`.

Nella prima parte di questa esercitazione discuteremo due *applicazioni* della ricostruzione con funzioni continue lineari a tratti: l'*approssimazione numerica di un integrale* e l'*approssimazione del grafico di una funzione.* Nella seconda parte si realizzano le procedure `integrale` per la prima applicazione e `grafico` per la seconda, e si utilizzano in due esempi.

Prima parte

Siano $[a,b]$ un intervallo non degenere e $f : [a,b] \to \mathbb{R}$ una funzione con derivata seconda continua.

- *Approssimazione numerica di un integrale*

  Si vuole approssimare il numero reale:

  $$I = \int_a^b f(x)\,\mathrm{d}x$$

  Scelto un numero intero $k \geqslant 1$ e posto, per $j = 0, \ldots, k$:

  $$x_j = a + \frac{b-a}{k}\,j$$

  siano $\tau = (x_0, x_1) \cup \cdots \cup (x_{k-1}, x_k)$ e σ_k l'elemento di $S(\tau)$ che interpola i dati $(x_0, f(x_0))$, $\ldots$, $(x_k, f(x_k))$. Per approssimare I si utilizza il numero reale:

  $$J_k = \int_a^b \sigma_k(x) dx$$

  La scelta è ragionevole. Infatti, utilizzando la base canonica di $S(\tau)$ si ha:

  $$\sigma_k = f(x_0) s_0 + \cdots + f(x_k) s_k$$

  e per il Teorema sull'errore di ricostruzione con funzioni continue lineari a tratti, posto $M_2 = \max_{x\in[a,b]} |\, f''(x)\,|$:

  $$e(f) = \max_{x\in[a,b]} |\, f(x) - \sigma_k(x)\,| \leqslant \frac{M_2}{8} \left(\frac{b-a}{k}\right)^2$$

  Allora:

  * Per l'errore assoluto commesso utilizzando J_k per approssimare I si ha:

    $$|\, I - J_k\,| \leqslant \frac{M_2}{8} \frac{(b-a)^3}{k^2}$$

    dunque: *l'errore può essere reso piccolo quanto si vuole scegliendo k sufficientemente grande.*

    Infatti:

    $$\begin{aligned} |\, I - J_k\,| &= \left| \int_a^b \big(f(x) - \sigma_k(x)\big)\,\mathrm{d}x \right| \leqslant \int_a^b |\, f(x) - \sigma_k(x)\,|\,\mathrm{d}x \leqslant \\ &\leqslant \int_a^b \max_{x\in[a,b]} |\, f(x) - \sigma_k(x)\,|\,\mathrm{d}x = (b-a) \max_{x\in[a,b]} |\, f(x) - \sigma_k(x)\,| \leqslant \\ &\leqslant \frac{M_2}{8} \frac{(b-a)^3}{k^2} \end{aligned}$$

∗ *Il calcolo di J_k è elementare*:

$$J_k = \frac{b-a}{k}\,\big(\tfrac{1}{2}f(x_0) + f(x_1) + \cdots + f(x_{k-1}) + \tfrac{1}{2}f(x_k)\big) \qquad (*)$$

Infatti:

$$J_k = \int_a^b \sigma_k(x)\,\mathrm{d}x = \int_a^b \big(f(x_0)s_0(x) + \cdots + f(x_k)s_k(x)\big)\,\mathrm{d}x =$$

$$= f(x_0)\int_a^b s_0(x)\,\mathrm{d}x + \cdots + f(x_k)\int_a^b s_k(x)\,\mathrm{d}x$$

Poiché:

$$\int_a^b s_j(x)\,\mathrm{d}x = \begin{cases} \dfrac{b-a}{2k} & \text{se } j = 0, k \\ \dfrac{b-a}{k} & \text{se } j = 1, \ldots, k-1 \end{cases}$$

si ottiene la formula cercata.

L'espressione $(*)$ si chiama, per un motivo evidente, *formula dei trapezi*. I valori x_j, $j = 0, \ldots, k$ vengono detti *nodi*.

L'approssimazione proposta consiste nell'approssimare l'integrale I di f con un'opportuna *somma pesata* J_k di campioni di f. Utilizzando il calcolatore, il valore di quest'ultima somma è *approssimato* con quello, Φ_k, determinato da un algoritmo. Per l'errore assoluto commesso approssimando I con Φ_k si ha:

$$|\, I - \Phi_k \,| = |\, (I - J_k) + (J_k - \Phi_k) \,| \leqslant |\, I - J_k \,| + |\, J_k - \Phi_k \,|$$

Il primo addendo, $|\, I - J_k \,|$, è indipendente dall'uso del calcolatore: tiene conto soltanto dell'aver approssimato l'integrale con la somma pesata. Come abbiamo visto, per $k \to \infty$ l'addendo tende a zero come $1/k^2$. Il secondo addendo, $|\, J_k - \Phi_k \,|$, tiene conto delle limitazioni imposte dall'uso del calcolatore nella somma pesata. Una paziente analisi degli errori introdotti dai vari arrotondamenti mostra che *nel caso peggiore* l'addendo *cresce*, per $k \to \infty$, come k. Complessivamente, detti C e C' numeri reali positivi opportuni e u la precisione di macchina, si ha:

$$|\, I - \Phi_k \,| \leqslant \frac{C}{k^2} + C'uk$$

La funzione che maggiora l'errore è *decrescente* fino a valori di k usualmente ben al di sopra di quelli utilizzati.

- *Approssimazione del grafico di una funzione*

  Si vuole disegnare una curva che approssima il grafico di f su $[a, b]$.

  Scelto un numero intero $k \geqslant 1$ e posto, per $j = 0, \ldots, k$:

$$x_j = a + \frac{b-a}{k}\,j$$

  siano $\tau = (x_0, x_1) \cup \cdots \cup (x_{k-1}, x_k)$ e σ_k l'elemento di $S(\tau)$ che interpola i dati $(x_0, f(x_0))$, $\ldots$, $(x_k, f(x_k))$. Per approssimare il grafico di f si utilizza il grafico di σ_k.

  La scelta è ragionevole. Infatti, per il Teorema sull'errore di ricostruzione con funzioni continue lineari a tratti si ha:

$$e(f) = \max_{x \in [a,b]} |\, f(x) - \sigma_k(x) \,| \leqslant \frac{M_2}{8}\left(\frac{b-a}{k}\right)^2$$

  Allora: *l'errore tra il grafico di f e quello di σ_k può essere reso piccolo quanto si vuole scegliendo k sufficientemente grande.* Inoltre il grafico di σ_k si ottiene in modo elementare: disegnando la spezzata di vertici $(x_0, f(x_0))$, $\ldots$, $(x_k, f(x_k))$. In *Scilab*, realizzata una procedura `f` da utilizzare per approssimare i valori di f ed assegnati valori alle variabili `a`, `b` e `k`, le istruzioni:

```
--> x = linspace(a,b,k+1)';

--> plot2d(x,f(x));
```

producono il grafico richiesto.

Seconda parte

- *La procedura* ***integrale***

  La definizione seguente realizza una procedura che, *dati* gli estremi dell'intervallo di integrazione a, b, una procedura che restituisce i valori della funzione integranda f e il numero di intervalli di τ, *restituisce* l'approssimazione J dell'integrale:

$$\int_a^b f(x)\,\mathrm{d}x$$

ottenuta con la formula dei trapezi su $k+1$ nodi:

```
function J = integrale(a,b,f,k)
    //
    // Approssima l'integrale di f su [a,b] con la formula dei trapezi
    // che utilizza k + 1 nodi.
    //
    pesi = ones(1,k+1);
    pesi(1) = 1/2; pesi(k+1) = 1/2;
    nodi = linspace(a,b,k+1)';
    J = (b-a) * pesi * f(nodi) / k;
endfunction
```

Si consideri l'integrale:

$$I = \int_0^\pi \operatorname{sen} x\,\mathrm{d}x = 2$$

Per approssimare i valori di $f(x) = \operatorname{sen} x$ e $f''(x) = -\operatorname{sen} x$ si utilizzano le funzioni:

```
function y = f(x)
    y = sin(x);
endfunction
```

e:

```
function y = d2f(x)
    y = - sin(x);
endfunction
```

Eseguiti gli assegnamenti `a = 0` e `b = %pi`, i comandi:

```
\\ Intestazione della tabella
printf('\n   k         J(k)       |2 - J(k)|     |2 - J(k)| < L(k)\n');
printf('  ----------------------------------------------------');
for k = 50:50:500,
    J = integrale(a,b,f,k);
    \\ Calcolo della stima di M2 = max |f''|
       nodi = linspace(a,b,k+1)';
       M = max(abs(d2f(nodi)));
    \\ Calcolo della stima della maggiorazione dell'errore |I - J|
```

```
        L = M * (b-a)^3 / (8 * k^2);
    \\ Scrittura della riga nella tabella
    printf('\n %4d  %9.8e    %3.2e            %s\n',...
                k,    J,   abs(2 - J),    string(abs(2 - J) < L));
end;
```

generano, nella *console*, la tabella:

```
  k         J(k)        |2 - J(k)|      |2 - J(k)| < L(k)
-------------------------------------------------------

  50  1.99934198e+00    6.58e-04             T

 100  1.99983550e+00    1.64e-04             T

 150  1.99992689e+00    7.31e-05             T

 200  1.99995888e+00    4.11e-05             T

 250  1.99997368e+00    2.63e-05             T

 300  1.99998172e+00    1.83e-05             T

 350  1.99998657e+00    1.34e-05             T

 400  1.99998972e+00    1.03e-05             T

 450  1.99999188e+00    8.12e-06             T

 500  1.99999342e+00    6.58e-06             T
```

Si osservi che, coerentemente con la teoria esposta nella Prima parte, l'errore $|\, I - J_k \,|$ *decresce* all'aumentare di k ed è minore della maggiorazione:

$$L_k = \frac{M_2}{8} \, \frac{(b-a)^3}{k^2}$$

Il valore L_k è approssimato utilizzando come *stima* di M_2 il massimo di `abs(d2f(nodi))`, vettore le cui componenti approssimano $|\, f''(x_0) \,|, \ldots, |\, f''(x_k) \,|$.

- *La procedura* ***grafico***

  La definizione seguente realizza una procedura che, *dati* gli estremi dell'intervallo a, b, una procedura che restituisce i valori della funzione f, il numero reale E e una maggiorazione M del massimo M_2 della funzione $|\, f''(x) \,|$ su $[a, b]$, *genera* il grafico di una funzione σ, continua e lineare a tratti su un opportuno insieme τ, tale che:

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

e *restituisce* il numero j dei campioni di f utilizzati per la determinazione di σ:

```
function j = grafico(a,b,f,E,M)
    //
    // Data una funzione f con derivata seconda continua
    // ed una maggiorazione M del massimo di | f''(x) | su [a,b],
    // genera il grafico di una funzione s, continua e lineare
    // a tratti, tale che:
    //
    //     per ogni x in [a,b] si ha | f(x) - s(x) | < E
    //
    // e restituisce il numero j di campioni di f utilizzati per
```

```
    // determinare s.
    //
    k = ceil((b - a) * sqrt(M / (8 * E)));
    j = k + 1;
    t = linspace(a,b,j)';
    plot2d(t,f(t),style = 2);
    xgrid(); xlabel('x');
    xtitle('Grafico di f(x) con errore < ' + ...
           msprintf('%3.2e',E) + '  (' + msprintf('%d',j) + ' punti)');
endfunction
```

Una volta scelto il valore di j, la procedura esegue un *campionamento uniforme* di f con j istanti e genera il grafico della funzione σ continua e lineare a tratti sugli intervalli aperti definiti dagli istanti di campionamento che ricostruisce i campioni ottenuti. Il valore di j scelto dalla procedura è ottenuto calcolando il più piccolo numero intero k che rende vera la relazione:

$$\frac{M_2}{8}\left(\frac{b-a}{k}\right)^2 < E$$

e ponendo $j = k + 1$.

Si consideri la funzione $f(x) = \operatorname{sen} x$. Per approssimare i valori di f si utilizza la funzione `f` definita nel punto precedente. Eseguiti gli assegnamenti `a = 0` e `b = %pi`, i comandi:

```
E = 1d-3; M = 1;
subplot(211); j = grafico(a,b,f,E,M);
// Istanti di campionamento
t = linspace(a,b,j)';
// 10 punti per intervallo + istanti di campionamento
x = linspace(a,b,11*j - 10)';
// Errore = | f(x) - s(x) |
errore = abs( f(x) - interp1(t,f(t),x) );
// Stampa nella console una stima di e(f)
printf('\nErrore nel grafico: %3.2e\n', max(errore));
// Grafico dell'errore
subplot(212); plot2d(x,errore,style = 5);
              xgrid(); xlabel('x'); ylabel('errore');
```

producono i grafici riportati in Figura 1 e, nella *console*:

```
Errore nel grafico:  9.43e-04
```

I valori della differenza $f - \sigma$ sono calcolati utilizzando la funzione `interp1`:

* `interp1`

  Questa *funzione predefinita* calcola i valori, in un insieme assegnato di punti, della funzione continua e lineare a tratti che interpola dati assegnati. Precisamente, dati vettori `x` e `y` di n componenti e `xp` di m componenti, l'assegnamento:

  ```
  yp = interp1(x,y,xp)
  ```

  determina la funzione σ continua e lineare a tratti sugli intervalli $(\mathtt{x}_1, \mathtt{x}_2), \ldots, (\mathtt{x}_{n-1}, \mathtt{x}_n)$ che interpola i dati $(\mathtt{x}_1, \mathtt{y}_1), \ldots, (\mathtt{x}_n, \mathtt{y}_n)$ e restituisce il vettore `yp` di componenti $\sigma(\mathtt{xp}_1)$, $\ldots$, $\sigma(\mathtt{xp}_m)$.

La procedura `grafico` richiede all'utilizzatore di fornire una maggiorazione M di M_2. Per eliminare, parzialmente, questa incombenza si osservi che una *stima* di M_2, quasi sempre *per difetto*, si ottiene, scelto un sottoinsieme *finito* A di $[a, b]$, utilizzando $\max_{x \in A} |f''(x)|$. La seguente modifica della definizione precedente realizza una procedura che, *data* una procedura che restituisce i valori di f'', calcola una stima di M_2 e la utilizza per determinare la funzione σ il cui grafico approssima quello di f:

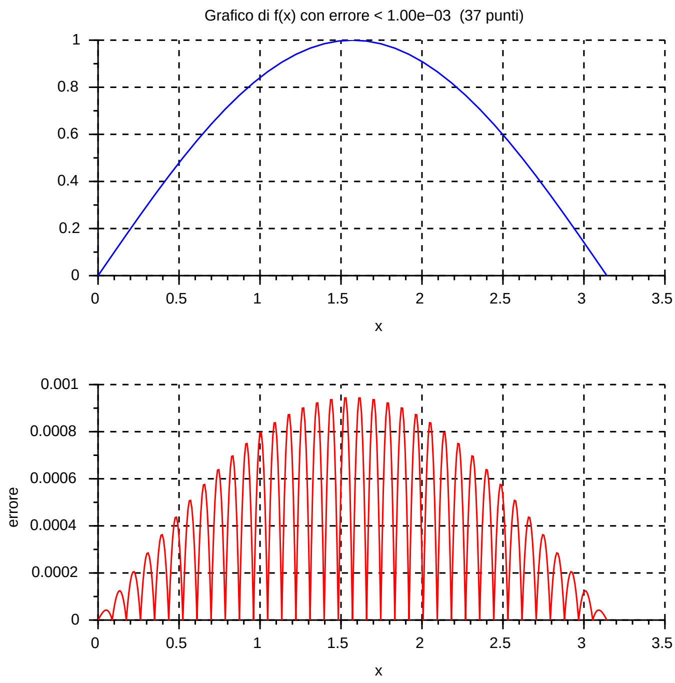

Figura 1: Grafici relativi alla procedura `grafico`.

```
function [j,M] = grafico(a,b,f,E,d2f)
    //
    // Data una funzione f con derivata seconda continua
    // ed una funzione d2f che restituisce i valori di f",
    // genera il grafico di una funzione s, continua e lineare
    // a tratti, tale che:
    //
    //     per ogni x in [a,b] si ha | f(x) - s(x) | < E
    //
    // e restituisce il numero j di campioni di f utilizzati per
    // determinare s e la riga M contenente le stime del massimo M2
    // di | f"(x) | su [a,b] determinate dalla procedura.
    //
    // Stima iniziale di M2
       n = 10;
       t = linspace(a,b,n)';
       M(1) = max(abs(d2f(t)));
    // Valore iniziale di j
       k = ceil((b - a) * sqrt(M(1) / (8 * E)));
       j = k + 1;
    // Nuova stima di M2
       t = linspace(a,b,j)';
       NuovoM = max(abs(d2f(t)));
    // Iterazione per migliorare la stima di M2
       while NuovoM > M($),
          M($+1) = NuovoM;
          k = ceil((b - a) * sqrt(M($) / (8 * E)));
          j = k + 1;
          // Nuova stima di M2
             t = linspace(a,b,j)';
             NuovoM = max(abs(d2f(t)));
       end;
```

```
    plot2d(t,f(t),style = 2);
    xgrid(); xlabel('x');
    xtitle('Grafico di f(x) con errore < ' + ...
           msprintf('%3.2e',E) + '  (' + msprintf('%d',j) + ' punti)');
endfunction
```

Per $f(x) = \operatorname{sen} x$, utilizzando la funzione `d2f` definita nel punto precedente per approssimare i valori di f'' ed eseguiti gli assegnamenti `a = 0` e `b = %pi`, i comandi:

```
E = 1d-3;
subplot(211); [j,M] = grafico(a,b,f,E,d2f);
// Istanti di campionamento
t = linspace(a,b,j)';
// 10 punti per intervallo + istanti di campionamento
x = linspace(a,b,11*j - 10)';
// Errore = | f(x) - s(x) |
errore = abs( f(x) - interp1(t,f(t),x) );
// Stampa nella console una stima di e(f)
printf('\nErrore nel grafico: %3.2e\n', max(errore));
// Grafico dell'errore
subplot(212); plot2d(x,errore,style = 5);
              xgrid(); xlabel('x'); ylabel('errore');
```

producono *gli stessi risultati* del caso precedente. Infatti la procedura determina *esattamente* M_2:

```
-->M
 M  =

    0.9848078
    0.9989931
    1.

-->M(3) == 1
 ans  =

  T
```

Esercizi

1. Sia $p(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$ un polinomio di grado $n \geqslant 2$. Dimostrare che per ogni $x \in \mathbb{R}$ si ha:

$$|\, p''(x) \,| \leqslant 2\,|\, a_2 \,| + 3 \cdot 2\,|\, a_3 \,|\,|\, x \,| + \cdots + n(n-1)\,|\, a_n \,|\,|\, x \,|^{n-2}$$

   e quindi, posto $m = \max\left(\,|\, a \,|, |\, b \,|\,\right)$:

$$M_2 = \max_{x \in [a,b]} |\, p''(x) \,| \leqslant \sum_{j=2}^{n} j(j-1)\,|\, a_j \,|\, m^{j-2}$$

2. Utilizzare il risultato dell'Esercizio precedente per definire una realizzazione della procedura che *dati* gli estremi a, b di un intervallo non degenere, la colonna c di $n \geqslant 1$ componenti $c_1, \ldots, c_n$ ed un numero reale E, *genera* una curva che approssima il grafico del polinomio $c_1 + c_2 x + \cdots + c_n x^{n-1}$ sull'intervallo $[a, b]$ con errore non superiore a E e *restituisce* il numero

j di campioni utilizzato per creare il grafico e stime Max del massimo e Min del minimo del polinomio sull'intervallo $[a, b]$. La procedura deve avere intestazione:

```
[j,Max, Min] = GraficoPoli(a,b,c,E)
```

Utilizzare la procedura per ottenere approssimazioni dei grafici di $p(x) = 3$, $q(x) = 1 - x$ e $r(x) = (x + 1)(x - 1)$ su $[-1, 5]$.