ESERCITAZIONE MATLAB 6: Interpolazione polinomiale

1. Utilizzando la base di Newton, il polinomio $p_n \in \Pi_n$ che interpola

$$(x_i, f_i), \qquad i = 0, 1, \ldots, n, \tag{1}$$

è dato da

$$\begin{aligned} p(z) &= f[x_0] + f[x_0, x_1](z - x_0) + f[x_0, x_1, x_2](z - x_0)(z - x_1) \\ &+ \ldots + f[x_0, x_1, x_2, \ldots, x_n](z - x_0)(z - x_1)\cdots(z - x_{n-1}). \end{aligned} \tag{2}$$

Implementare la function Matlab `diffdiv.m` per il calcolo delle differenze divise

$$f[x_0, \ldots, x_r], \qquad r = 0, 1, \ldots, n$$

Si ricorda che le differenze divise di ordine zero ed uno sono così definite

$$\begin{aligned} f[x_j] &= f_j & 0 \le j \le n, \\ f[x_0, x_j] &= \frac{f[x_j] - f[x_0]}{x_j - x_0} & 1 \le j \le n, \end{aligned}$$

mentre le differenze divise di ordine $r \ge 2$ verificano la seguente relazione

$$f[x_0, \ldots, x_{r-2}, x_{r-1}, x_j] = \frac{f[x_0, \ldots, x_{r-2}, x_j] - f[x_0, \ldots, x_{r-2}, x_{r-1}]}{x_j - x_{r-1}},$$

per ogni $j = r, \ldots, n$.

2. Si implementi l'algoritmo di Horner generalizzato, per la valutazione del polinomio $p(z)$ in (2) una volta che le differenze divise sono state calcolate. In notazione Matlab, indicando con `dd` e con `x` i vettori contenenti le differenze divise e le ascisse dei nodi di interpolazione, rispettivamente, tale algoritmo è dato da

```
np1 = length(x);
p   = dd(np1);
p   = dd(np1-1) + (z-x(np1-1))*p;
p   = dd(np1-2) + (z-x(np1-2))*p;
    .
    .
p   = dd(1) + (z-x(1))*p;
```

Si osserva che la precedente versione presuppone che `z` sia una variabile scalare. Si riformuli il codice in modo che possa essere applicato anche se `z` è una matrice fornendo in uscita una matrice `p` contenente il valore del polinomio interpolante su ciascun elemento di `z`.
Si salvi l'M-file con il nome `hornerg.m`.

3. Al fine di verificare che i codici `diffdiv.m` e `hornerg.m` sono corretti si digitino un certo numero di volte i seguenti comandi

```
>> x=rand(5,1);
>> f=rand(5,1);
>> dd=diffdiv(x,f);
>> p=hornerg(x,x,dd);
>> p-f
```

   Se la differenza tra `p` e `f` è dell'ordine della precisione di macchina allora i codici sono corretti (con elevata probabilità!!!).

4. Determinare il polinomio di grado massimo 5 interpolante la funzione $f(x) = \sin(x)$ sul numero opportuno di ascisse equidistanti definite sull'intervallo $[0, \pi]$. Tracciare il grafico della funzione e del polinomio interpolante su tale intervallo.

5. Determinare i polinomi di grado massimo $10, 15, 20$ interpolanti la funzione di Runge

$$f(x) = \frac{1}{1 + 25\,x^2}$$

   su ascisse equidistanti definite sull'intervallo $[-1, 1]$. Rappresentare i grafici della funzione e dei tre polinomi.
   Ripetere l'esercizio utilizzando le ascisse di Chebyshev

$$x_i = \cos\left(\frac{2i+1}{2n+2}\,\pi\right), \qquad i = 0, 1, \ldots, n$$

   dove $n + 1$ è l'opportuno numero di ascisse da usare per ottenere i polinomi interpolanti del suddetto ordine.

SOLUZIONE:

1. Il seguente codice implementa il calcolo delle differenze divise.

```
function dd = diffdiv(x,f)
%
% dd = diffdiv(x,f)
%
% Calcola le differenze divise del polinomio interpolante
% i nodi
%
% (x(i+1),f(i+1)),   i=0,1,...,n
%
% dove n=length(x)-1=length(f)-1.
%
% Input:
%    x: ascisse dei nodi di interpolazione
%    f: ordinate dei nodi di interpolazione
%
% Output:
%   dd: differenze divise ovvero
%      dd(r+1) = f[x(1),...,x(r+1)],   r=0,1,...,n.

np1  = length(x);

if (length(f)~=np1),   error('input non corretto'),  end

T = zeros(np1);
T(:,1) = f(:);

for r=2:np1
   for j=r:np1
      if x(j)==x(r-1)
          error('le ascisse non sono a due a due distinte')
      end
      T(j,r)=(T(j,r-1)-T(r-1,r-1))/(x(j)-x(r-1));
   end
end

dd = diag(T);
```

2. Il seguente codice implementa l'algoritmo di Horner generalizzato.

```
 function p = hornerg(z,x,dd)

% p = hornerg(z,x,dd)
%
% Valuta il polinomio interpolante espresso nella
% base di Newton.
%
% Dati di input:
%     z: vettore contenente i punti nei
%         quali si vuole valutare il polinomio
%
%     x: vettore contenente le ascisse
%         dei nodi di interpolazione
%
%    dd: vettore contenente le differenze divise
%
% Dato di output:
%     p: vettore con i valori del polinomio
%        interpolante richiesti.

np1  = length(x);

if (length(dd)~=np1),
    error('input non corretto'),
end

p=dd(np1)*ones(size(z));

for i=np1-1:-1:1
   p=dd(i) + (z-x(i)).*p;
end
```

3. A titolo di esempio, il risultato che si ottiene digitando i comandi indicati è il seguente.

```
>> x=rand(5,1);
>> f=rand(5,1);
>> dd=diffdiv(x,f);
>> p=hornerg(x,x,dd);
```

```
>> p-f
ans =
   1.0e-15 *
         0
         0
   -0.1110
         0
   -0.3886
```

4. Al fine di calcolare il polinomio di grado massimo 5 che interpola la funzione $f(x) = \sin(x)$ sono necessari 6 nodi di interpolazione. In particolare, volendo usare ascisse equidistanti sull'intervallo $[0, \pi]$ i comandi da digitare sono i seguenti:

   ```
   >> x=linspace(0,pi,6);
   >> f=sin(x);
   >> dd=diffdiv(x,f);
   ```

   Per tracciare poi il grafico della funzione e del suo polinomio interpolante si possono usare i seguenti comandi

   ```
   >> z=linspace(0,pi,1000);
   >> p=hornerg(z,x,dd);
   >> plot(z,sin(z),'b-',z,p,'r-')
   ```

   In questo caso si osserva che il polinomio interpolante approssima bene la funzione. Le due curve sono infatti pressocchè sovrapposte come mostrato nel grafico a sinistra in Figura 1. Nel grafico a destra della medesima figura è stato riportato il corrispondente errore. Per tracciare tale grafico sono stati digitati i seguenti comandi

   ```
   >> semilogy(z,abs(sin(z)-p))
   >> axis([0 pi 1e-7 1e-2])
   ```

5. Le istruzioni da eseguire sono simili a quelle del precedente esercizio. Ad esempio, per il polinomio di grado massimo 20 interpolante $f(x)$ su ascisse equidistanti

   ```
   >> x=linspace(-1,1,21);
   >> f=1./(1+25*x.^2);
   >> dd=diffdiv(x,f);
   >> z=linspace(-1,1,1000);
   >> p=hornerg(z,x,dd);
   >> plot(z,1./(1+25*z.^2),'b-',z,p,'r-')
   ```

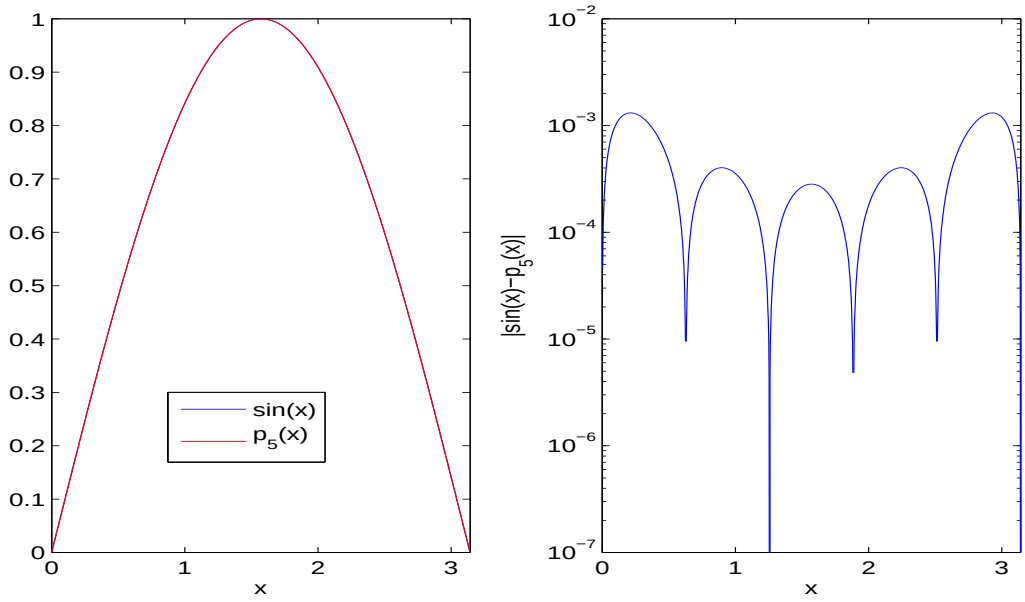

Figura 1: Polinomio interpolante la funzione $\sin(x)$ su ascisse equidistanti in $[0, \pi]$.

mentre per il polinomio del medesimo grado massimo ma su ascisse di Chebyschev

```
>> xc=cos(((2*[0:20]+1)/42)*pi);
>> fc=1./(1+25*xc.^2);
>> ddc=diffdiv(xc,fc);
>> z=linspace(-1,1,1000);
>> pc=hornerg(z,xc,ddc);
>> plot(z,1./(1+25*z.^2),'b-',z,pc,'r-')
```

I risultati ottenuti sono mostrati in Figura 2.

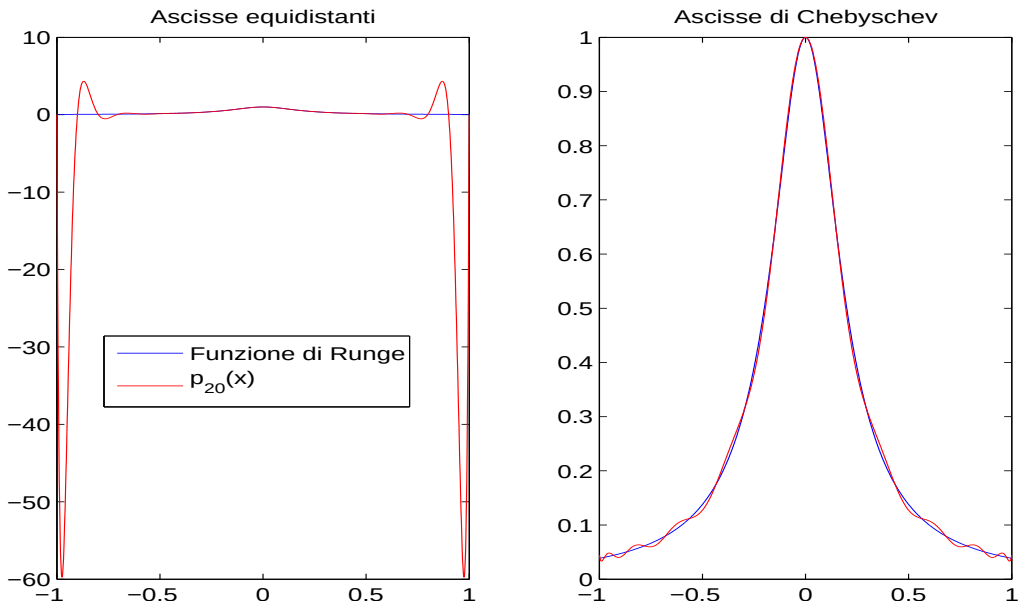

Figura 2: Polinomio interpolante la funzione di Runge su ascisse equidistanti e di Chebyschev.