ESERCITAZIONE MATLAB 6: Interpolazione polinomiale

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

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

è dato da

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

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

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

di tale polinomio. La seguente è l'interfaccia della function.

```
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.
```

Si osservi che in (1) i nodi sono numerati a partire da zero mentre gli indici degli elementi di un vettore in Matlab partono da uno. Pertanto, come descritto nel precedente help, nel codice `diffdiv.m` si avrà

$$\begin{aligned} \texttt{x(1)} &= x_0, \quad \texttt{x(2)} = x_1, \quad \ldots, \quad \texttt{x(n+1)} = x_n, \\ \texttt{f(1)} &= f_0, \quad \texttt{f(2)} = f_1, \quad \ldots, \quad \texttt{f(n+1)} = f_n, \end{aligned}$$

dove `n=length(x)-1` e `n=length(f)-1`. In output il codice deve fornire il vettore `dd` i cui elementi contengono i seguenti valori

$$\begin{aligned} \texttt{dd(1)} &= f[x_0], \\ \texttt{dd(2)} &= f[x_0, x_1], \\ \texttt{dd(3)} &= f[x_0, x_1, x_2], \\ &\vdots \\ \texttt{dd(n+1)} &= f[x_0, x_1, x_2, \ldots, x_n]. \end{aligned}$$

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. La sua interfaccia deve essere la seguente

```
% 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.
```

3. Al fine di verificare che i codici `diffdiv.m` e `hornerg.m` che avete scritto 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 $[0, \pi]$. Rappresentare il grafico della funzione e del polinomio interpolante su tale intervallo.

5. Determinare i polinomi di grado massimo $6, 10, 14$ 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(i+1) = f[x(1),...,x(i+1)],   i=0,1,...,n.

n = length(x);
n1 = length(f);

if (n1~=n),
   error('Dati di input non corretti'),
end

dd=f;
n=n-1;
for j=1:n
   for k=n:-1:j
      dd(k+1)=(dd(k+1)-dd(k))/(x(k+1)-x(k-j+1));
   end
end
```

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.

n  = length(x);
n1 = length(dd);

if (n1<n), error('dati non corretti'), end

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

for i=n-1:-1:1
   p=p.*(z-x(i))+dd(i);
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,1001);
   >> 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 quelli del precedente esercizio. Ad esempio, per il polinomio di grado massimo 14 interpolante $f(x)$ su ascisse equidistanti

   ```
   >> x=linspace(-1,1,15);
   >> f=1./(1+25*x.^2);
   >> dd=diffdiv(x,f);
   >> z=linspace(-1,1,1001);
   >> p=hornerg(z,x,dd);
   >> plot(z,1./(1+25*z.^2),'b-',z,p,'r-')
   ```

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

```
>> xc=cos(((2*[0:14]+1)/30)*pi);
>> fc=1./(1+25*xc.^2);
>> ddc=diffdiv(xc,fc);
>> z=linspace(-1,1,1001);
>> pc=hornerg(z,xc,ddc);
>> plot(z,1./(1+25*z.^2),'b-',z,pc,'r-')
```

I risultati ottenuti sono mostrati in Figura 2.

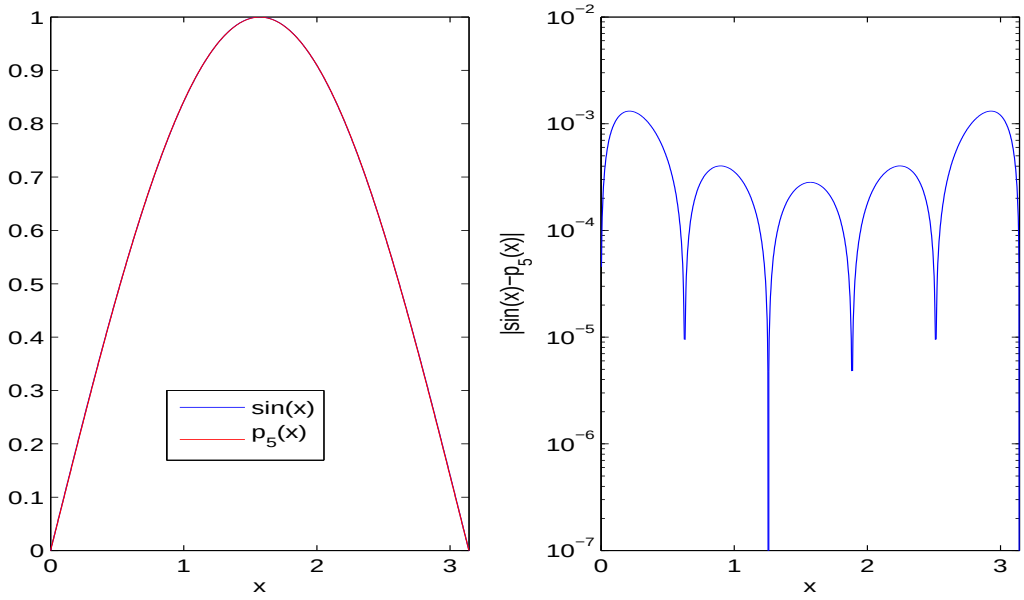

Figura 1: Polinomio interpolante la funzione $\sin(x)$ su ascisse equidistanti in $[0, \pi]$.

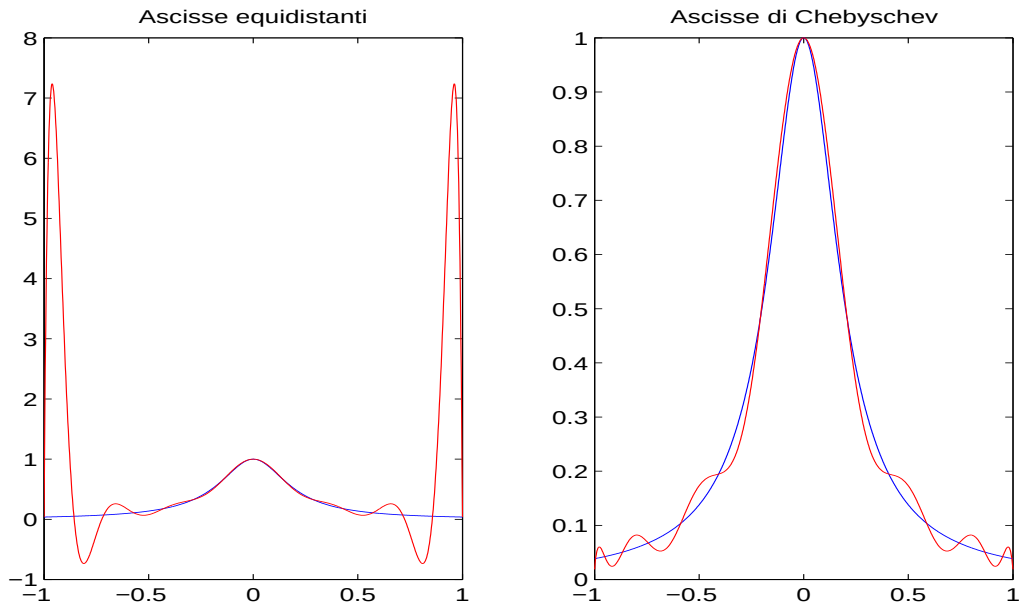

Figura 2: Polinomio interpolante la funzione di Runge su ascisse equidistanti e di Chebyschev in $[-1, 1]$.