Interpolazione Polinomiale

Esercitazione guidata in Matlab

Realizzazione in Matlab dell'interpolazione polinomiale mediante

1) il polinomio interpolatore di Lagrange

2) il polinomio di Newton mediante le differenze divise.

3) l'interpolazione mediante spline

Specifiche per la realizzazione del codice

1) Per realizzare il polinomio interpolatore di Lagrange costruire una funzione matlab con la seguente sintassi:

dati in input i vettori x e y dei punti di interpolazione, restituisca in output il polinomio interpolatore valutato nel punto xval (scalare)

y_Lagrange = Lagrange(x,y,xval)

seguendo l'algoritmo presentato:

```
y_Lagrange=0;
Per i = 1,...,n
        product = y(i);
        Per j = 1,...,n
                se i ~= j
                        product = product*(xval-x(j))/(x(i)-x(j));
                 end
        end
        y_Lagrange = y_Lagrange +product;
end
```

2) Per realizzare il polinomio interpolatore di Newton costruire due funzioni matlab:

- una funzione che calcoli la tabella delle differenze divise con la seguente sintassi:
  dati in input i vettori x e y dei punti di interpolazione, restituisca in output la diagonale della matrice relativa alla tabella delle differenze divise

d=differenze_div(x,y)

seguendo l'algoritmo presentato:

per k = 0,...,n
d (k) = y(k)
end
per i = 2, 3..., n
per k = n, n-1 ,..., i
d (k) = (d (k) - d (k-1)) / (x(k)-x(k-i))
end
end

- una funzione che calcoli il polinomio di Newton con la seguente sintassi:

  dati in input il vettore x dei punti d'interpolazione, il vettore d delle differenze divise calcolate con la funzione differenze_div appena realizzata, restituisca in output il polinomio interpolatore valutato nei punti xval (xval può essere scalare o vettore).

  Pn=Newton_interp(xval, x, d)

  seguendo l'algoritmo presentato:

Pn= d(n);
for i = n,n-1,...,1
Pn = d(i) + (xval-x(i)) Pn*
end

2) Per realizzare l'interpolazione mediante funzioni spline cubiche con condizione di raccordo ai bordi si faccia uso della funzione matlab **spline** che presenta la seguente sintassi:

yval = spline (x, y, xval)

calcola la forma polinomiale a tratti della spline cubica interpolante, valutata nei punti xval, fornendo così, in yval, i valori del interpolante in xval.

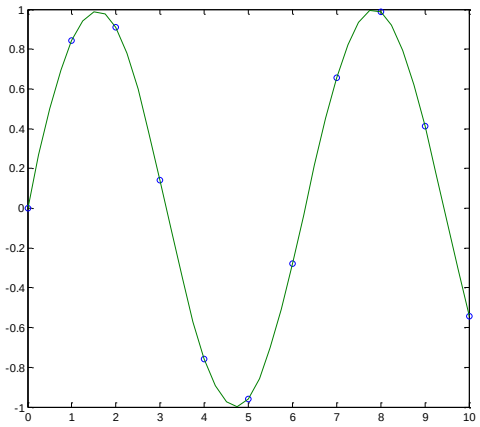
xval può essere uno scalare o un vettore e rappresenta i punti in cui valutare il polinomio.

x e y sono i punti di interpolazione. x e y devono avere la stessa lunghezza.

yval ha le stesse dimensioni xval.

ES:

```
x = 0:10;
y = sin(x);
xval = 0:.25:10;
yval = spline(x,y,xval);
plot(x,y,'o',xval,yval)
```



Acquisizione punti da mouse:

In Matlab la funzione **ginput** permette di acquisire punti graficamentedal mouse o cursore.

Sintassi:

[x,y] = ginput(n)

[x,y] = ginput

ginput consente di selezionare punti dalla figura utilizzando il mouse per posizionare il cursore. La figura deve essere in exposure prima che ginput riceva l'input.

In particolare:

[x, y] = ginput (n) consente di selezionare n punti dagli assi attuali e restituisce le coordinate x e y nei vettori colonna x ed y, rispettivamente. Premere il tasto Invio per terminare l'inserimento prima degli punti n.

[x, y] = ginput raccoglie un numero illimitato di punti fino a quando si preme il tasto Invio.