I polinomi

- Un polinomio in Matlab è dato da un vettore che contiene i sui coefficienti ordinati da a_n fino ad a_0.
- Per valutare un polinomio in uno o più punti si usa il comando `polyval`:

```
>> y = polyval(p,x)
```

  - `x` è un vettore dove si specificano le ascisse nelle quale si vuole valutare il polinomio `p`.
  - `y` è un vettore che contine i valori di `p` in `x`.

I polinomi

- I comandi `polyint(p)` e `polyder(p)` calcolano rispetivamente i coefficienti di una primitiva (quella che si annulla in $x = 0$) e della derivata di `p`.
- Se `x` e `y` sono due vettori di $n + 1$ componenti, il comando `p=polyfit(x,y,n)` calcola il polinomio interpolatore dei dati $\{(x_i, y_i)\}_{i=0}^{n}$.
- Se `x` e `y` sono due vettori di $n + 1$ componenti, il comando `p=polyfit(x,y,m)` calcola i coefficienti del polinomio di grado `m` che approssima le $n + 1$ coppie di valori $\{(x_i, y_i)\}_{i=0}^{n}$ nel senso dei minimi quadrati.

Esempio di Runge

Il seguente script calcola i coefficienti `p` del polinomio di grado n che interpola la funzione

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

in $n + 1$ nodi equispaziati dell'intervallo [-5,5] (estremi compressi).

Disegna il grafico della funzione f e del polinomio interpolatore e disegna anche i punti dove coincidono.

Esempio di Runge

```
n=input('Grado del polinomio: ');
a=-5;b=5;
h=(b-a)/n;
x=[a:h:b];
fx=1./(x.^2+1);
p=polyfit(x,fx,n);
xx=linspace(a,b);
pxx=polyval(p,xx);
fxx=1./(xx.^2+1);
plot(xx,fxx,xx,pxx,x,fx,'*');
legend('1/(x^2+1)','Polinomio intepolatore','Dati');
```

Si può osservare come

$$\max_{-5\leq x\leq 5} |f(x) - \Pi_n f(x)| \longrightarrow \infty \quad \text{se } N \to \infty\,,$$

dove $\Pi_n f(x)$ denota il polinomio interpolatore in nodi equispaziati di grado n. (I suoi coefficienti sono nel vettore p.)

Nodi di Chebishev

Nell'intervallo $[-1, 1]$ sono

$$\hat{x}_i = -\cos\left(\frac{i\pi}{n}\right) \qquad i = 0, \ldots, n\,.$$

E in un intervallo generico $[a, b]$

$$x_i = \frac{a+b}{2} + \frac{b-a}{2}\hat{x}_i \qquad i = 0, \ldots, n\,.$$

Nodi di Chebishev

Il seguente script calcola i coefficienti q del polinomio di grado n che interpola di nuovo la funzione

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

ma negli $n + 1$ nodi di Chebyshev dell'intervallo [-5,5].

Come prima disegna il grafico della funzione f, il grafico del polinomio interpolatore e i punti dove coincidono.

Nodi di Chebishev

```
n=input('Grado del polinomio: ');
a=-5; b=5;
x=-cos([0:n]*pi/n);
x=(a+b)/2+(b-a)/2*x;
fx=1./(x.^2+1);
q=polyfit(x,fx,n);
xx=linspace(a,b);
qxx=polyval(q,xx);
fxx=1./(xx.^2+1);
plot(xx,fxx,xx,qxx,x,fx,'* ');
legend('1/(x^2+1)','Polinomio interpolatore','Dati');
```

Osserviamo che adesso

$$\max_{-5\le x\le 5} |f(x) - \widetilde{\Pi}_n f(x)| \longrightarrow 0 \quad \text{se } N \to \infty\,,$$

dove $\widetilde{\Pi}_n f(x)$ denota il polinomio interpolatore nei nodi di Chebyshev di grado n. (I suoi coefficienti sono nel vettore q.)

Differenze divise

$$f[x_i, x_{i+1}, \ldots, x_j] = \frac{f[x_{i+1}, \ldots, x_j] - f[x_i, x_{i+1}, \ldots, x_{j-1}]}{x_j - x_i}$$

$$\begin{array}{llllll}
x_0 & f[x_0] & & & & \\
x_1 & f[x_1] & f[x_0, x_1] = \frac{f[x_1]-f[x_0]}{x_1-x_0} & & & \\
x_2 & f[x_2] & f[x_1, x_2] = \frac{f[x_2]-f[x_1]}{x_2-x_1} & f[x_0, x_1, x_2] = \frac{f[x_1,x_2]-f[x_0,x_1]}{x_2-x_0} & & \\
x_3 & f[x_3] & f[x_2, x_3] = \frac{f[x_3]-f[x_2]}{x_3-x_2} & f[x_1, x_2, x_3] = \frac{f[x_2,x_3]-f[x_1,x_2]}{x_3-x_1} & f[x_0, x_1, x_2, x_3] &
\end{array}$$

$$f[x_0, x_1, x_2, x_3] = \frac{f[x_1, x_2, x_3] - f[x_0, x_1, x_2]}{x_3 - x_0}$$

Forma di Newton del polinomio interpolatore

$$P(x) = f[x_0] + (x - x_0)f[x_0, x_1] + (x - x_0)(x - x_1)f[x_0, x_1, x_2] + (x - x_0)(x - x_1)(x - x_2)f[x_0, x_1, x_2, x_3]$$

$$= f[x_0] + (x - x_0)\big(f[x_0, x_1] + (x - x_1)\big[f[x_0, x_1, x_2] + (x - x_2)f[x_0, x_1, x_2, x_3]\big]\big)$$

Algoritmo di Horner

$$c_1 = f[x_0], \quad c_2 = f[x_0, x_1], \quad c_3 = f[x_0, x_1, x_2], \quad c_4 = f[x_0, x_1, x_2, x_3].$$

$$\begin{aligned} &Pz = c_4 \\ &\textit{for}\, i = 3 : -1 : 1 \\ &\qquad Pz = c_i + (z - x_{i-1})\, Pz \\ &\textit{end} \end{aligned}$$

Differenze divise

La seguente funzione calcola il valore del polinomio che interpola i dati $\{(x_i, y_i)\}_{i=1}^{n}$ nelle ascisse contenute nel vettore z. Restituisce anche la tabella delle differenza divise.

```
function [Pz,A]=interpol(x,y,z)
n=length(x);
A=zeros(n,n);
A(:,1)=y;
for j=2:n
    for i=j:n;
        A(i,j)=(A(i,j-1)-A(i-1,j-1))./(x(i)-x(i-j+1));
    end
end
Pz=A(n,n);
for k=n-1:-1:1
    Pz=A(k,k)+(z-x(k)).*Pz;
end
A=[x';A];
```

Differenze divise

La seguente funzione aggiunge un nuovo punto (x, y) a una tabella di differenze divise

```
function B=addpoint(x,y,A)
B=zeros(size(A)+[1 1]);
B(1:end-1,1:end-1)=A;
[M,N]=size(B);
B(M,1)=x; B(M,2)=y;
for j=3:N
    B(M,j)=(B(M,j-1)-B(M-1,j-1))/(B(M,1)-B(M-j+2,1));
end
```

Usando questa funzione possiamo costruire facilmente la tabella di differenze divise per i dati $\{(x_i, y_i)\}_{i=1}^{n}$.

```
function A=interpoladd(x,y)
A=[x(1) y(1)];
n=length(x);
for k=2:n
    A=addpoint(x(k),y(k),A);
end
```