INTERPOLAZIONE POLINOMIALE *

A. SOMMARIVA † E M.VENTURIN ‡

1. Interpolazione polinomiale. Siano dati $N+1$ punti $x_0, \ldots, x_N$ a due a due distinti e in ordine crescente (cioé $x_i < x_{i+1}$), e i valori $y_0, \ldots, y_N$ ivi assunti da una funzione $y = f(x)$.

Il problema dell'*interpolazione polinomiale* (cf. [1, p.131], [8, p.289]) consiste nel calcolare il polinomio p_N di grado N tale che

$$p_N(x_i) = y_i, \;\; i = 0, \ldots, N. \tag{1.1}$$

Perché questo problema abbia senso, necessita garantire l'esistenza ed unicitá di tale polinomio p_N, questione che puó essere provata in vari modi (ad esempio in termini della non singolaritá della matrice di Vandermonde [18]).

Nel caso $N = 1$, il problema diventa relativamente semplice. Non è altro che il calcolo della retta che passa per due punti assegnati $P_0 = (x_0, y_0)$ e $P_1 = (x_1, y_1)$ cioė da

$$\frac{y - y_0}{y_1 - y_0} = \frac{x - x_0}{x_1 - x_0} \tag{1.2}$$

abbiamo facilmente

$$y = y_0 \frac{x - x_1}{x_0 - x_1} + y_1 \frac{x - x_0}{x_1 - x_0}. \tag{1.3}$$

Notiamo subito che i polinomi

$$L_0(x) := \frac{x - x_1}{x_0 - x_1} \tag{1.4}$$

$$L_1(x) := \frac{x - x_0}{x_1 - x_0} \tag{1.5}$$

hanno la particolarità che

$$L_i(x_j) = \delta_{ij}$$

dove al solito $\delta_{i,j}$ è il *delta di Kronecker* , cioè

$$\delta_{i,i} = 1, \; \delta_{i,j} = 0 \text{ per } i \neq j.$$

Da (1.3), (1.4), (1.5) si ha ovviamente

$$p_N(x) = y_0 L_0(x) + y_1 L_1(x). \tag{1.6}$$

Vediamo il grafico dei polinomi di Lagrange nel caso $x_0 = 1$, $x_1 = 3$. Digitiamo il codice

*Ultima revisione: 10 novembre 2011

†DIPARTIMENTO DI MATEMATICA PURA ED APPLICATA, UNIVERSITÀ DEGLI STUDI DI PADOVA, VIA TRIESTE 63, 35121 PADOVA, ITALIA (ALVISE@MATH.UNIPD.IT)

‡DIPARTIMENTO DI INFORMATICA, UNIVERSITÀ DEGLI STUDI DI VERONA, STRADA LE GRAZIE 15, 37134 VERONA, ITALIA (MANOLO.VENTURIN@GMAIL.COM)

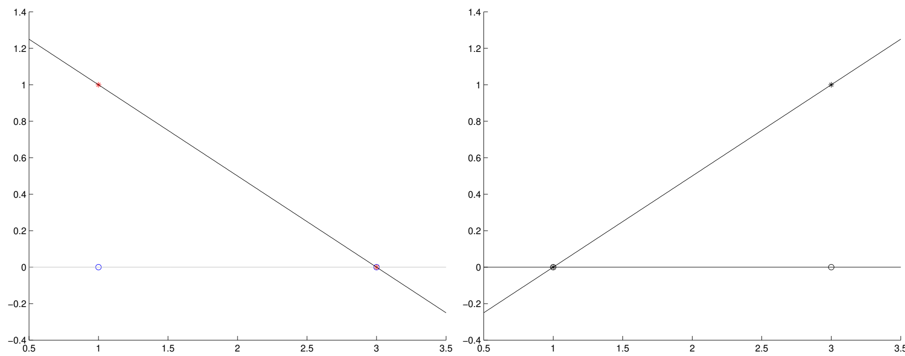

FIGURA 1.1. *I polinomi di Lagrange L_0, L_1 relativi ai punti $x_0 = 1$, $x_1 = 3$.*

```
clf;

x0=1;
x1=3;
s=0.5:0.01:3.5;

caso=2; % DICE QUALE POLINOMIO DI LAGRANGE VOGLIAMO STUDIARE.

if caso == 1
    lx=(s-x1)/(x0-x1);
    l0=(x0-x1)/(x0-x1);
    l1=(x1-x1)/(x0-x1);
else
    lx=(s-x0)/(x1-x0);
    l0=(x0-x0)/(x1-x0);
    l1=(x1-x0)/(x1-x0);
end

hold on;

% DISEGNO PUNTI x0=1 e x1=3.
plot(x0,0,'bo');
plot(x1,0,'bo');

% DISEGNO VALORI x0=1 e x1=3 POLINOMI DI LAGRANGE.
plot(x0,l0,'r*');
plot(x1,l1,'r*');

% DISEGNO ASSE x.
plot(s,zeros(size(s)));

% DISEGNO POLINOMI DI LAGRANGE
plot(s,lx,'k-');

hold off
```

ottenendo, rispettivamente per `caso=1` e `caso=2`, i grafici in figura.

L'idea dei polinomi di Lagrange è di estendere questa proprietà al caso $N > 1$. Lagrange (cf. [12], [5]) osservò che il polinomio di grado N

$$L_k(x) := \prod_{j=0, j\neq k}^{N} \frac{x - x_j}{x_k - x_j} = \frac{(x - x_0)\ldots(x - x_{k-1})(x - x_{k+1})\ldots(x - x_N)}{(x_k - x_0)\ldots(x_k - x_{k-1})(x_k - x_{k+1})\ldots(x_k - x_N)} \tag{1.7}$$

è tale che $L_k(x_j) = \delta_{ij}$. Quindi si vede facilmente che

$$p_N(x) = \sum_{k=0}^{N} y_k L_k(x). \tag{1.8}$$

Infatti

$$p_N(x_j) = \sum_{k=0}^{N} y_k L_k(x_j) = \sum_{k=0}^{N} y_k \delta_{k,j} = y_j. \tag{1.9}$$

Si osservi come ogni termine della sommatoria mostra separa (in qualche senso) il contributo delle ordinate y_k da quello dei polinomi di Lagrange L_k essenzialmente dipendenti dai punti $\{x_j\}_{j=0,\ldots,N}$ (si rifletta bene su questa affermazione).

1.0.1. Un esempio. Calcoliamo il polinomio di grado 2 che assume nei nodi $x_0 = -2$, $x_1 = 1$, $x_2 = 3$ rispettivamente i valori $f_0 = -2$, $f_1 = 11$, $f_2 = 17$. Come si vede dalla definizione, i polinomi di Lagrange sono

$$\begin{aligned}
L_0(x) &= \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} = \frac{(x - 1)(x - 3)}{(-2 - 1)(-2 - 3)} = \frac{(x - 1)(x - 3)}{15} \\
L_1(x) &= \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} = \frac{(x - (-2))(x - 3)}{(1 - (-2))(1 - 3)} = \frac{(x + 2)(x - 3)}{-6} \\
L_2(x) &= \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} = \frac{(x - (-2))(x - 1)}{(3 - (-2))(3 - 1)} = \frac{(x + 2)(x - 1)}{10}
\end{aligned}$$

Si vede subito che i polinomi L_0, L_1, L_2 sono di secondo grado (e quindi tali le loro combinazioni lineari), che

$$L_0(x_0) = L_0(-2) = 1, L_0(x_1) = L_0(1) = 0, L_0(x_2) = L_0(3) = 0$$

$$L_1(x_1) = L_1(1) = 1, L_1(x_0) = L_1(-2) = 0, L_1(x_2) = L_1(3) = 0$$

$$L_2(x_2) = L_2(3) = 1, L_2(x_0) = L_2(-2) = 0, L_2(x_1) = L_2(1) = 0$$

e che il polinomio definito in (1.8)

$$p_2(x) = -2\, L_0(x) + 11\, L_1(x) + 17\, L_2(x)$$

è tale che

$$p(x_0) = -2\, L_0(x_0) + 11\, L_1(x_0) + 17\, L_2(x_0) = -2 \cdot 1 + 11 \cdot 0 + 17 \cdot 0 = -2,$$

$$p(x_1) = -2\, L_0(x_1) + 11\, L_1(x_1) + 17\, L_2(x_1) = -2 \cdot 0 + 11 \cdot 1 + 17 \cdot 0 = 11,$$

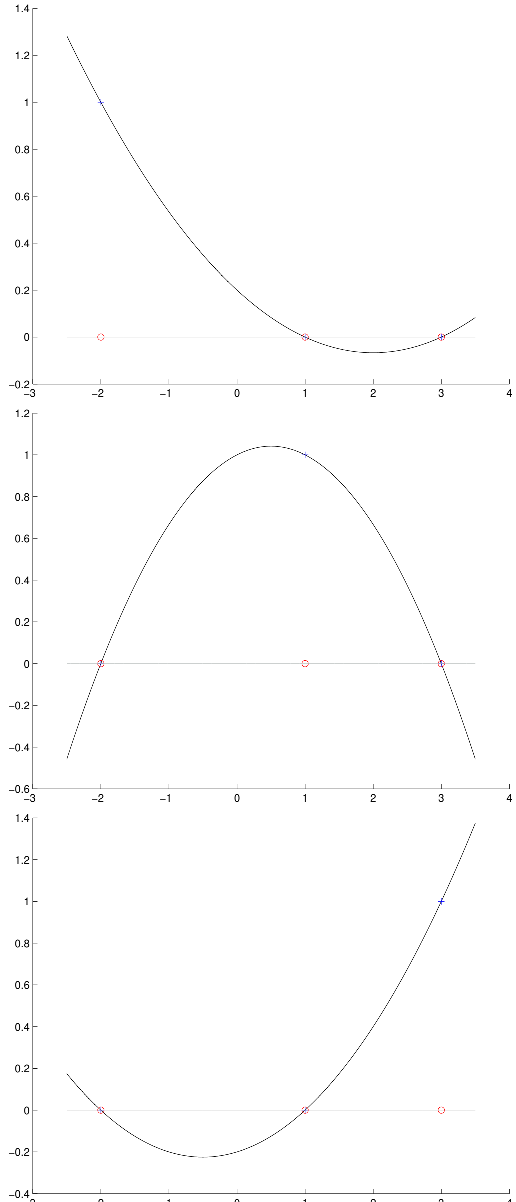

FIGURA 1.2. *I polinomi di Lagrange L_0, L_1, L_2 relativi ai punti $x_0 = -2$, $x_1 = 1$, $x_2 = 3$.*

$$p(x_2) = -2\,L_0(x_2) + 11\,L_1(x_2) + 17\,L_2(x_2) = -2 \cdot 0 + 11 \cdot 0 + 17 \cdot 1 = 17,$$

e quindi è proprio il polinomio interpolante cercato.

Dal punto di vista pratico, (1.8) ha alcuni problemi. Se dopo aver calcolato il polinomio p_N interpolante in $N+1$ punti (x_0, y_0), ..., (x_N, y_N), desideriamo ottenere il polinomio p_{N+1} interpolante in $N+2$ punti (x_0, y_0), ..., (x_N, y_N), la formula (1.8) è inefficiente poichè bisogna ricalcolare tutti i polinomi di Lagrange. Fortunatamente, esistono altre maniere di esprimere il polinomio interpolatore, come quella di Newton (cf. [1, p.138], [8, p.294], [13],

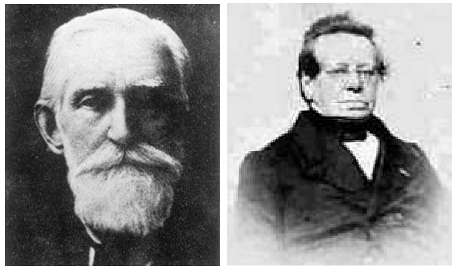

FIGURA 1.3. *Pafnuty Lvovich Chebyshev (1821-1894) e Rehuel Lobatto (1797-1866).*

[11]), che non soffrono di questo problema.

Per quanto concerne l'errore, se la funzione è di classe C^{N+1}, si puó vedere dal *teorema del resto* che

$$f(x) - p_N(x) = f^{(N+1)}(\xi) \frac{\prod_{i=0}^{N}(x - x_i)}{(N+1)!} \tag{1.10}$$

dove $\xi \in I$ con I il più piccolo intervallo aperto contenente $x_0, \ldots, x_N$.

Ma come scegliere (avendone la possibilità) i punti in cui interpolare la funzione f : $[a, b] \to \mathbb{R}$? Mostriamo due casi notevoli:

1. *nodi equispaziati*: fissato N, i punti sono

$$x_k = a + k\,\frac{(b-a)}{N},\ \ k = 0, \ldots, N; \tag{1.11}$$

2. *nodi di Chebyshev-Gauss (scalati)* [8, p.294]: fissato N, i punti sono

$$x_k = \frac{(a+b)}{2} + \frac{(b-a)}{2}\,t_k,\ \ k = 0, \ldots, N \tag{1.12}$$

con

$$t_k = \cos\left(\frac{2k+1}{2N+2}\,\pi\right),\ k = 0, \ldots, N; \tag{1.13}$$

3. *nodi di Chebyshev-Gauss-Lobatto (scalati)* [8, p.295]: fissato N, i punti sono

$$x_k = \frac{(a+b)}{2} + \frac{(b-a)}{2}\,t_k,\ \ k = 0, \ldots, N \tag{1.14}$$

con

$$t_k = -\cos\left(\frac{k\pi}{N}\right),\ k = 0, \ldots, N; \tag{1.15}$$

In seguito useremo i nodi di Chebyshev-Gauss-Lobatto, che a differenza dei nodi di Chebyshev (cf. [14], [4]) includono gli estremi dell'intervallo. Si consiglia di effettuare gli esercizi successivi sostituendo ai nodi di Chebyshev-Gauss-Lobatto (scalati), quelli di Chebyshev-Gauss. Si osservi che l'interpolante polinomiale in un set di nodi prefissati non

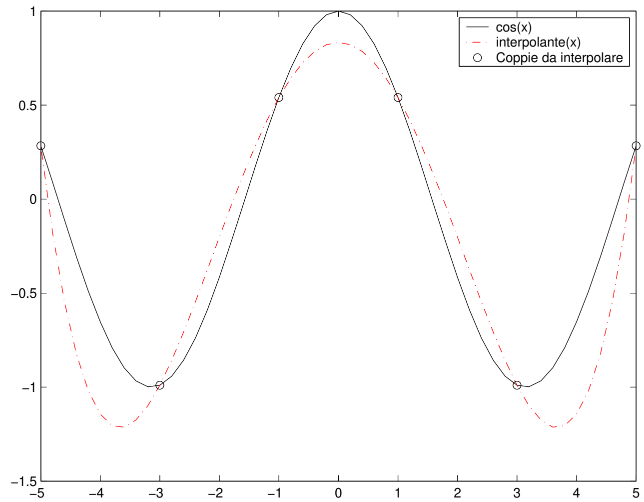

FIGURA 1.4. *Grafico che illustra l'interpolazione su nodi equispaziati (la funzione ha la linea continua, il polinomio interpolatore è tratteggiato, i nodi di interpolazioni coi cerchietti).*

converge sempre puntualmente alla funzione da approssimare. Infatti, per la funzione di Runge

$$f(x) = \frac{1}{1+x^2}, \ \ x \in [-5, 5] \tag{1.16}$$

si ha che il polinomio interpolatore p_N in nodi equispaziati non converge (puntualmente) a f. Fortunatamente ciò non succede per i nodi di Chebyshev-Gauss, per cui comunque (per un teorema dovuto a Faber) esistono funzioni continue f ma non C^1 tali che l'interpolante p_N non converge puntualmente a f.

2. Interpolazione polinomiale in Matlab/Octave. Una volta nota le coppie da interpolare

$$(x_k, y_k), \ \ k = 0, \ldots, N$$

e immagazzinate le componenti nei vettori $x = (x_{k+1})$, $y = (y_{k+1})$ ove $k = 0, \ldots, N$ (la scelta degli indici dei vettori x e y è dovuta al fatto che un vettore in Matlab/Octave non può avere indice 0), siamo interessati a valutare il polinomio interpolatore p_N nei punti s_j con $j = 1, \ldots, M$. A tale scopo, Matlab/Octave propongono le funzioni `polyfit` e `polyval`. Per capirne il loro utilizzo ci aiutiamo con l'help di Matlab.

2.1. Il comando `polyfit`. Partiamo con il descrivere il comando `polyfit`.

```
>> help polyfit

 POLYFIT Fit polynomial to data.
    POLYFIT(X,Y,N) finds the coefficients of a polynomial P(X) of
    degree N that fits the data, P(X(I))~=Y(I), in a least-squares sense.

    [P,S] = POLYFIT(X,Y,N) returns the polynomial coefficients P and a
    structure S for use with POLYVAL to obtain error estimates on
    predictions.  If the errors in the data, Y, are independent normal
```

```
    with constant variance, POLYVAL will produce error bounds which
    contain at least 50% of the predictions.

    The structure S contains the Cholesky factor of the Vandermonde
    matrix (R), the degrees of freedom (df), and the norm of the
    residuals (normr) as fields.

    [P,S,MU] = POLYFIT(X,Y,N) finds the coefficients of a polynomial
    in XHAT = (X-MU(1))/MU(2) where MU(1) = mean(X) and MU(2) = std(X).
    This centering and scaling transformation improves the numerical
    properties of both the polynomial and the fitting algorithm.

    Warning messages result if N is >= length(X), if X has repeated, or
    nearly repeated, points, or if X might need centering and scaling.

    See also POLY, POLYVAL, ROOTS.

>>
```

Quello che non dice la routine è come venga descritto il polinomio interpolatore. Aiutiamoci con il calcolo simbolico eseguito da Matlab, per calcolare il polinomio interpolatore in forma usuale. Consideriamo di nuovo il polinomio di grado 2 che interpola le coppie $(-2,-2)$, $(1,11)$, $(3,17)$.

```
>> syms x
>> simplify((-2*(x-1)*(x-3)/15)+(11*(x+2)*(x-3)/(-6))+(17*(x+2)*(x-1)/10))
ans =
-4/15*x^2+61/15*x+36/5
>> format long
>> -4/15
ans =
  -0.26666666666667
>> 61/15
ans =
   4.06666666666667
>> 36/5
ans =
   7.20000000000000
>>
```

Testiamo l'uso di polyfit per quest'esempio

```
>> x=[-2 1 3];
>> y=[-2 11 17];
>> a=polyfit(x,y,2);
>> a
a =
  -0.26666666666667   4.06666666666667   7.19999999999999
>>
>>
```

Quindi, se $a = (a_k)_{k=1,\ldots,3}$, abbiamo

$$p_2(x) = a_1\, x^2 + a_2\, x + a_3.$$

Più in generale, se p_N è il polinomio interpolatore di grado N, e $a = (a_k)$ è il vettore ottenuto utilizzando `polyfit`, allora

$$p_N(x) = a_1\, x^N + a_2\, x^{N-1} + \ldots + a_{N+1}.$$

2.2. Il comando `polyval`. Descriviamo il comando `polyval`. Aiutandosi con l'help di Matlab

```
>> help polyval

  To get started, select "MATLAB Help" from the Help menu.


 POLYVAL Evaluate polynomial.
    Y = POLYVAL(P,X), when P is a vector of length N+1 whose elements
    are the coefficients of a polynomial, is the value of the
    polynomial evaluated at X.

        Y = P(1)*X^N + P(2)*X\^(N-1) + ... + P(N)*X + P(N+1)

    If X is a matrix or vector, the polynomial is evaluated at all
    points in X.  See also POLYVALM for evaluation in a matrix sense.

    Y = POLYVAL(P,X,[],MU) uses XHAT = (X-MU(1))/MU(2) in place of X.
    The centering and scaling parameters MU are optional output
    computed by POLYFIT.

    [Y,DELTA] = POLYVAL(P,X,S) or [Y,DELTA] = POLYVAL(P,X,S,MU) uses
    the optional output structure S provided by POLYFIT to generate
    error estimates, Y +/- delta.  If the errors in the data input to
    POLYFIT are independent normal with constant variance, Y +/- DELTA
    contains at least 50% of the predictions.

    See also POLYFIT, POLYVALM.

>>
```

L'help ci suggerisce che quanto fornito da `polyfit` è effettivamente un polinomio scritto nella forma

$$y = P(1) \cdot x^N + P(2) \cdot x^{(N-1)} + \cdots + P(N) \cdot x + P(N+1),$$

come notato precedentemente.
Se $a = (a_k)_{k=1,\ldots,N+1}$ e $s = (s_j)_{j=1,\ldots,M}$, la chiamata

```
t=polyval(a,s);
```

valuta il polinomio

$$p_N(x) = a_1\, x^N + \ldots + a_{N+1}$$

nei nodi $s = (s_j)$ e pone $t = (t_j)$ con $t_j = p_N(s_j)$.

2.3. Una funzione per il calcolo e la valutazione del polinomio interpolatore. Avendo a disposizione i comandi `polyfit` e `polyval`, posti

$$x = \{x_k\}_{k=1,\ldots,N+1},\ y = \{y_k\}_{k=1,\ldots,N+1},\ s = \{s_k\}_{k=1,\ldots,M}$$

vediamo di valutare nei punti s_j il polinomio p_N tale che

$$p_N(x_k) = y_k.$$

A tal proposito definiamo la funzione `interpol`

```
function t=interpol(x,y,s)

%----------------------------------------------
% Interpolazione
%
% In input:
% x: nodi.
% y: valori nei nodi.
% s: nodi su cui calcolare l'interpolante.
%
% In output:
% t: valori dell'interpolante o approssimante.
%----------------------------------------------

m=length(x)-1;
coeff=polyfit(x,y,m);
t=polyval(coeff,s);
```

Quale esempio torniamo di nuovo al polinomio p_2 di grado 2 che interpola le coppie $(-2,-2)$, $(1,11)$, $(3,17)$. Usando Matlab e il calcolo simbolico abbiamo visto direttamente che

$$p_2(x) = (-4/15) \cdot x^2 + (61/15) \cdot x + 36/5.$$

In effetti si verifica numericamente che questo è il polinomio interpolante $(-2,-2)$, $(1,11)$, $(3,17)$

```
>> format long
>> p2=inline('-4/15*x.^2+61/15*x+36/5');
>> feval(p2,-2)

ans =

  -1.999999999999999

>> feval(p2,1)

ans =

    11

>> feval(p2,3)

ans =
```

```
    17

>>
```

Vediamo per prima cosa quanto valgono $p_2(-1)$, $p_2(0)$, $p_2(2)$.

```
>> format long
>> p2=inline('-4/15*x.^2+61/15*x+36/5');
>> feval(p2,-1)

ans =

   2.866666666666667

>> feval(p2,0)

ans =

   7.200000000000000

>> feval(p2,2)

ans =

  14.266666666666666

>>
```

Verifichiamo di seguito tali risultati con la funzione `interpol`.

```
>> % ASCISSE "x" E ORDINATE "y" CHE DEFINISCONO
>> % IL POLINOMIO INTERPOLATORE.
>> x=[-2 1 3];
>> y=[-2 11 17];
>> % ASCISSE IN CUI VALUTARE IL POLINOMIO INTERPOLATORE.
>> s=[-1 0 2];
>> % VALUTO IL POLINOMIO INTERPOLATORE.
>> t=interpol(x,y,s);
>> t=t'

t =

   2.866666666666664
   7.199999999999995
  14.266666666666662

>>
```

Di conseguenza, a meno di decimali, `interpol` fa quanto richiesto.

3. Un esempio sull'interpolazione polinomiale in Matlab/Octave.

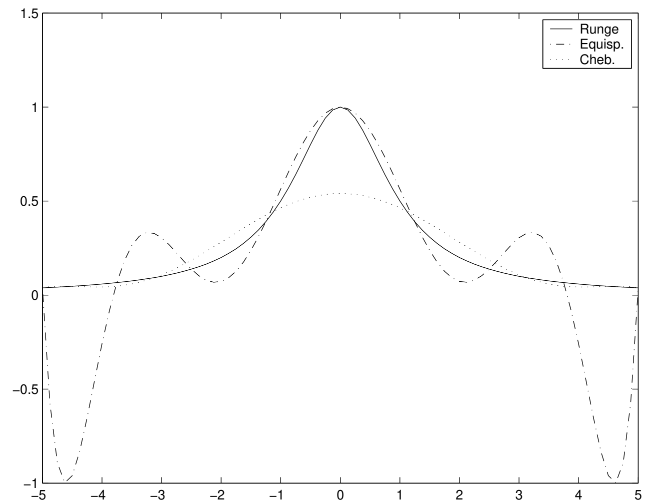

FIGURA 3.1. *Grafico della funzione di Runge* $1/(1+x^2)$ *nell'intervallo* $[-5,5]$ *e delle sue interpolanti di grado* 8 *nei nodi equispaziati e di Chebyshev-Gauss-Lobatto (rispettivamente linea continua, tratteggiata e a puntini, come da legenda).*

1. Si implementi una function che calcola i nodi di Chebyshev-Gauss per un intervallo qualsiasi.
2. Si implementi poi una function che interpoli polinomialmente una funzione che assume valori y su un vettore di nodi x che vengono forniti in input. Tale funzione Matlab deve inoltre calcolare i valori t che l'interpolante polinomiale p_N assume nei nodi test s. Si testi il codice, producendo dei grafici, sulle funzioni
   - $f(x) = \frac{1}{1+x^2}$ in $[-5,5]$.

   usando n nodi equispaziati (con alcuni valori di n tra 6 e 11) e n nodi di Chebyshev-Gauss-Lobatto (con i medesimi valori di n), calcolando gli errori in norma infinito tra la funzione e le interpolanti. In particolare, si mostri (mediante grafici e/o valori dell'errore) che per il controesempio di Runge (cf. [9], [16]) l'aumento del numero di nodi equispaziati non migliora la ricostruzione della funzione da parte dell'interpolante.

3.1. Implementazione. Si scrivano le functions `cheb.m`, `interpol.m`, `runge.m` definite rispettivamente come

```
function xc=cheb(a,b,n)
for m=1:1:n
    xc(m)=(a+b)/2-((b-a)/2)*cos(pi*(m-1)/(n-1));
end
```

```
function [fx]=runge(x)
fx=1./(x.^2+1);
```

Eseguire poi da Matlab/Octave il programma principale `esperimento.m`

```
n=11;               % GRADO.
```

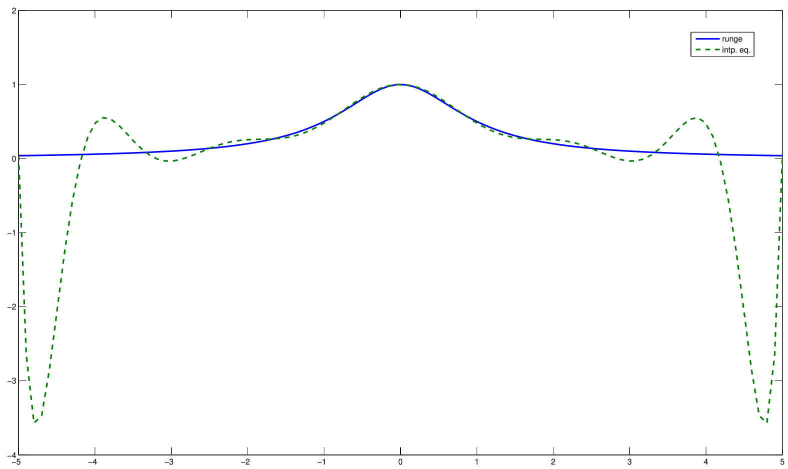

FIGURA 3.2. *Grafico della funzione di Runge* $1/(1+x^2)$ *nell'intervallo* $[-5,5]$ *e della sua interpolante di grado* 12 *nei nodi equispaziati (rispettivamente linea continua, tratteggiata, come da legenda.*

```
x=-5:10/n:5;          % NODI INTERP. (EQUISPAZ.).
y=runge(x);           % FUNZIONE NEI NODI EQUISP.
s=-5:10/(10*n):5;      % NODI TEST.
t=interpol(x,y,s);    % INTERPOLANTE NEI NODI TEST.

xcheb=cheb(-5,5,n); % NODI CHEB.
ycheb=runge(xcheb); % FUNZIONE NEI NODI CHEB.
tcheb=interpol(xcheb,ycheb,s);   % INTP. CHEB.
plot(s,runge(s),s,t,s,tcheb);    % PLOT INTP. VS RUNGE.
err_eqs=norm(runge(s)-t,inf);      % ABS. ERR. EQUISPAZ.
err_cheb=norm(runge(s)-tcheb,inf); % ABS. ERR. CHEB.
fprintf('\n \t [ABS.ERR.][EQS]: 2.2e [CHEB]: %2.2e',err_eqs,err_cheb);
```

Provando per diversi valori di n, e tralasciando i warnings di Matlab relativi a polyfit, otteniamo:

```
[N]:  2 [ABS.ERR.][EQS]: 6.46e-001 [CHEB]: 9.62e-001
[N]:  3 [ABS.ERR.][EQS]: 7.07e-001 [CHEB]: 6.46e-001
[N]:  4 [ABS.ERR.][EQS]: 4.38e-001 [CHEB]: 8.29e-001
[N]:  5 [ABS.ERR.][EQS]: 4.33e-001 [CHEB]: 4.58e-001
[N]:  6 [ABS.ERR.][EQS]: 6.09e-001 [CHEB]: 6.39e-001
[N]:  7 [ABS.ERR.][EQS]: 2.47e-001 [CHEB]: 3.11e-001
[N]:  8 [ABS.ERR.][EQS]: 1.04e+000 [CHEB]: 4.60e-001
[N]:  9 [ABS.ERR.][EQS]: 2.99e-001 [CHEB]: 2.04e-001
[N]: 10 [ABS.ERR.][EQS]: 1.92e+000 [CHEB]: 3.19e-001
[N]: 11 [ABS.ERR.][EQS]: 5.57e-001 [CHEB]: 1.32e-001
[N]: 12 [ABS.ERR.][EQS]: 3.66e+000 [CHEB]: 2.18e-001
[N]: 13 [ABS.ERR.][EQS]: 1.07e+000 [CHEB]: 8.41e-002
[N]: 14 [ABS.ERR.][EQS]: 7.15e+000 [CHEB]: 1.47e-001
```

```
[N]: 15 [ABS.ERR.][EQS]: 2.10e+000 [CHEB]: 5.33e-002
```

Si nota subito che

- l'istruzione `m=length(x)-1;` calcola il grado dell'interpolante che è uguale al numero di punti meno uno;
- i nodi di `cheb`, sono quelli di Chebyshev-Gauss-Lobatto. Infatti per $m = 1$

$$\begin{aligned} xc(1) &= \frac{(a+b)}{2} - \frac{(b-a)}{2} \cdot \cos\left(\pi \cdot \frac{1-1}{n-1}\right) \\ &= \frac{(a+b)}{2} - \frac{(b-a)}{2} \cdot \cos(0) \\ &= \frac{(a+b)}{2} - \frac{(b-a)}{2} \\ &= a \end{aligned} \tag{3.1}$$

  mentre $m = n$ porge

$$\begin{aligned} xc(n) &= \frac{(a+b)}{2} - \frac{(b-a)}{2} \cdot \cos\left(\pi \cdot \frac{n-1}{n-1}\right) \\ &= \frac{(a+b)}{2} - \frac{(b-a)}{2} \cdot \cos(\pi) \\ &= \frac{(a+b)}{2} - \frac{(b-a)}{2}(-1) \\ &= b \end{aligned} \tag{3.2}$$

  Osserviamo che a prima vista la routine cheb sembra produrre un set di nodi diversi da quelli di Chebyshev-Gauss-Lobatto (scalati) introdotti in (1.15). In realtà sono gli stessi come si può facilmente verificare matematicamente o in Matlab

```
>> N=10;
>> k=0:N;
>> t=-cos(k*pi/N);
>> t=sort(t);
>> t
t =
  Columns 1 through 7
   -1.0000   -0.9511   -0.8090   -0.5878   -0.3090   -0.0000    0.3090
  Columns 8 through 11
    0.5878    0.8090    0.9511    1.0000
>> xc=cheb(-1,1,N+1)
xc =
  Columns 1 through 7
   -1.0000   -0.9511   -0.8090   -0.5878   -0.3090   -0.0000    0.3090
  Columns 8 through 11
    0.5878    0.8090    0.9511    1.0000
>>
```

- il caso dei nodi equispaziati è semplice da programmare. Infatti la chiamata

```
>> a=-5;
>> b=5;
>> N=10;
>> h=(b-a)/N;
>> x=a:h:b
x =

  -5  -4  -3  -2  -1   0   1   2   3   4   5

>> length(x)
ans = 11
>>
```

suggerisce come fissato N si possa generare un set di $N+1$ punti equispaziati in $[a, b]$ (comprendente gli estremi).

- l'errore dell'interpolante della funzione di Runge nei nodi equispaziati non converge a 0. Osserviamo la parziale descrescita degli errori dall'iterata 2 alla 4 per poi aumentare peggiorando da iterazione ad iterazione. Similmente osserviamo la parziale descrescita degli errori dall'iterata 3 alla 5, dall'iterata 5 alla 7, per poi peggiorare sempre più . Il grafico in figura fa capire come l'errore sia particolarmente significativo in prossimità degli estremi -5 e 5.

- l'errore dell'interpolante della funzione di Runge nei nodi di Chebyshev-Gauss-Lobatto (scalati) converge a 0 (si osservi la particolare decrescita dei gradi pari e dispari).

- Ci si può chiedere perchè i nodi di Chebyshev-Gauss-Lobatto (scalati) vadano meglio. Ricordiamo che la funzione di Runge è $C^{\infty}([-5, 5])$ e sussiste il seguente teorema dovuto a Bernstein (cf. [3, p.136])
TEOREMA 3.1. *Se $f \in C^1([a, b])$ con $[a, b]$ intervallo limitato e chiuso della retta reale, il polinomio P_n di grado n di interpolazione della funzione f nei nodi di Chebyshev-Gauss di grado $n+1$ converge uniformemente a f su $[a, b]$, per $n \to \infty$. Se inoltre $f \in C^2([a, b])$ si ha la seguente stima dell'errore*
$$\|f - P_n\|_{\infty} = O(n^{-1/2}).$$
Il comportamento dell'interpolante nei nodi di Chebyshev-Gauss-Lobatto è analogo.

4. Esercizio. Calcolare analiticamente il polinomio di terzo grado che interpola le coppie $(1, 0)$, $(2, 2)$, $(4, 12)$, $(5, 21)$. Quindi valutare tale polinomio nei punti di ascissa -1, 0, 10.

5. Facoltativo e tecnico: sul calcolo dell'interpolante polinomiale. Lanciamo in Matlab 6.1 il codice `esperimento`, che fa un confronto tra nodi equispaziati e di Chebyshev per $n = 12$ (quindi ci sono 13 punti di sampling). Il risultato è il seguente

```
>> esperimento
          [ABS.ERR.][EQS]: 3.66e+000 [CHEB]: 2.18e-001
>>
```

Variamo in `esperimento` la variabile n, ponendola uguale a 13. Matlab da' il seguente messaggio di warning:

```
>> esperimento
Warning: Polynomial is badly conditioned. Remove repeated data points
         or try centering and scaling as described in HELP POLYFIT.
> In C:\MATLAB6p1\toolbox\matlab\polyfun\polyfit.m at line 74
  In D:\DVD_27_OTTOBRE_2007\CS_2008\INTERPOLAZIONE_2008\MFILES\interpol.m
  at line 17
  In D:\DVD_27_OTTOBRE_2007\CS_2008\INTERPOLAZIONE_2008\MFILES\esperimento.m
  at line 5

          [ABS.ERR.][EQS]: 1.07e+000 [CHEB]: 8.41e-002
>>
```

Poniamo ora $n = 15$ è il risultato è anche peggio

```
>> esperimento
Warning: Polynomial is badly conditioned. Remove repeated data points
         or try centering and scaling as described in HELP POLYFIT.
> In C:\MATLAB6p1\toolbox\matlab\polyfun\polyfit.m at line 74
  In D:\DVD_27_OTTOBRE_2007\CS_2008\INTERPOLAZIONE_2008\MFILES\interpol.m
  at line 17
  In D:\DVD_27_OTTOBRE_2007\CS_2008\INTERPOLAZIONE_2008\MFILES\esperimento.m
  at line 5
Warning: Polynomial is badly conditioned. Remove repeated data points
         or try centering and scaling as described in HELP POLYFIT.
> In C:\MATLAB6p1\toolbox\matlab\polyfun\polyfit.m at line 74
  In D:\DVD_27_OTTOBRE_2007\CS_2008\INTERPOLAZIONE_2008\MFILES\interpol.m
   at line 17
  In D:\DVD_27_OTTOBRE_2007\CS_2008\INTERPOLAZIONE_2008\MFILES\esperimento.m
   at line 9

          [ABS.ERR.][EQS]: 2.10e+000 [CHEB]: 5.33e-002
>>
```

Diamo una spiegazione. Il codice di `polyfit` risolve il sistema lineare prodotto dalla matrice di Vandermonde V (cf. [18]), con una fattorizzazione QR (cf. [17]). In altre parole se $V = (V_{i,j})$ con $V_{i,j} = x_i^{j-1}$ (x_i sono le ascisse dei nodi di sampling prescelti), si scrive $A = Q * R$ con Q ortonormale e R triangolare superiore e quindi si risolve $Vc = y$ (c sono i coefficenti del polinomio interpolante $p(x) = \sum_k c_k x^k$) come soluzione del sistema $Rc = Q' * y$. Se R è malcondizionata in norma 1, più precisamente $\|R\|_1 \|R^{-1}\|_1 > 10^{10}$, Matlab/Octave segnala un warning. La cosa è più elaborata di così in quanto `polyfit` usa la cosidetta *economy size version* della fattorizzazione QR, ma tralasciamo i dettagli per chiarezza.

Per sopperire a questi problemi una delle possibili soluzioni consiste nell'usare la cosidetta *formulazione baricentrica* dei polinomi di Lagrange (cf. [2], [19]). Questa è implementata in Matlab nel codice `barylag` [19]. Una volta scaricato e scompattato il file zip, e digitato il codice `esperimento_bar`

```
n=15;                      % GRADO.
x=(-5:10/n:5)';         % NODI INTERP. (EQUISPAZ.).
y=runge(x);          % FUNZIONE NEI NODI EQUISP.
s=(-5:10/(10*n):5)';  % NODI TEST.
data_eqs=[x y];
```

```
t=barylag(data_eqs,s);

xcheb=(cheb(-5,5,n))';    % NODI CHEB.
ycheb=runge(xcheb);    % FUNZIONE NEI NODI CHEB.
data_cheb=[xcheb ycheb];
tcheb=barylag(data_cheb,s);
                      % INTP. CHEB.
plot(s,runge(s),s,t,s,tcheb);    % PLOT INTP. VS RUNGE.
err_eqs=norm(runge(s)-t,inf);      % ABS. ERR. EQUISPAZ.
err_cheb=norm(runge(s)-tcheb,inf); % ABS. ERR. CHEB.
fprintf('\n \t [ABS.ERR.][EQS]: %2.2e [CHEB]: %2.2e',err_eqs,err_cheb);
```

abbiamo

```
>> esperimento_bar

          [ABS.ERR.][EQS]: 2.10e+000 [CHEB]: 5.33e-002
```

Per curiosità paragoniamo i due codici `esperimento` ed `esperimento_bar` per $n = 100$, ottenendo rispettivamente

```
>> esperimento
Warning: Polynomial is badly conditioned. Remove repeated data points
         or try centering and scaling as described in HELP POLYFIT.
> In C:\MATLAB6p1\toolbox\matlab\polyfun\polyfit.m at line 74
  In D:\DVD_27_OTTOBRE_2007\CS_2008\INTERPOLAZIONE_2008\MFILES\interpol.m
   at line 17
  In D:\DVD_27_OTTOBRE_2007\CS_2008\INTERPOLAZIONE_2008\MFILES\esperimento.m
   at line 5
Warning: Polynomial is badly conditioned. Remove repeated data points
         or try centering and scaling as described in HELP POLYFIT.
> In C:\MATLAB6p1\toolbox\matlab\polyfun\polyfit.m at line 74
  In D:\DVD_27_OTTOBRE_2007\CS_2008\INTERPOLAZIONE_2008\MFILES\interpol.m
   at line 17
  In D:\DVD_27_OTTOBRE_2007\CS_2008\INTERPOLAZIONE_2008\MFILES\esperimento.m
   at line 9

          [ABS.ERR.][EQS]: 5.02e+019 [CHEB]: 2.14e+007
>>
```

e

```
>> esperimento_bar

          [ABS.ERR.][EQS]: 4.57e+005 [CHEB]: 5.62e-009
>>
```

segno che i risultati ottenuti con `polyfit` sono inaccettabili, mentre quelli di `barylag` sono perfettamente in parallelo con quanto detto nelle sezioni precedenti relativamente al fenomeno di Runge (si osservi che nel primo caso l'interpolazione nei nodi di Chebyshev sembra divergere, in disaccordo con la teoria!).

6. Sulla formula del resto. Si consideri la funzione

$$f(x) := \log(x)$$

e si supponga di conoscere i valori per $x_0 = 1$, $x_1 = 1.1$ ed $x_2 = 1.2$. Si calcoli il polinomio di secondo grado p_2 che interpola tali punti, e lo si valuti in $s = 1.09$. Quindi si utilizzi la formula dell'errore per funzioni di classe C^{N+1} (*teorema del resto*)

$$f(x) - p_N(x) = f^{(N+1)}(\xi)\frac{\prod_{i=0}^{N}(x - x_i)}{(N+1)!}, \xi \in I(x_0, \ldots, x_N) \tag{6.1}$$

per valutare

$$|\log(s) - p_2(s)|.$$

Risoluzione. Il polinomio interpolatore è per quanto visto

$$p(x) = \log(1)L_0(x) + \log(1.1)L_1(x) + \log(1.2)L_2(x),$$

dove L_j è il j-simo polinomio di Lagrange.

Dalla formula del resto (per $f(x) = \log(x)$, $x = 1.09$, $x_0 = 1$, $x_1 = 1.1$, $x_2 = 1.2$ ed $N = 2$) otteniamo:

$$|\log(1.09) - p_2(1.09)| = |\frac{2}{\xi^3}\frac{(1.09-1)(1.09-1.1)(1.09-1.2)}{6}|,$$

in quanto la derivata terza di $\log(x)$ è $\frac{2}{x^3}$. Siccome $\xi \in (1, 1.2)$, dalla decrescenza di $\frac{2}{x^3}$ deduciamo che

$$1.1574 \approx \frac{2}{1.2^3} < \frac{2}{\xi^3} < \frac{2}{1^3} = 2.$$

Essendo

$$(1.09-1)(1.09-1.1)(1.09-1.2) \approx 9.9 \cdot 10^{-5}$$

ricaviamo quindi che l'errore dell'interpolante è

$$1.9097 \cdot 10^{-5} \approx 1.1574 \cdot 9.9 \cdot 10^{-5}/6 \leq |\log(s) - p_2(s)| \leq 2 \cdot 9.9 \cdot 10^{-5}/6 = 3.3 \cdot 10^{-5}.$$

Utilizzando la funzione *interpol.m* precedentemente introdotta, si verifica che

$$p_2(1.09) \approx 0.08615260795055$$

mentre

$$\log(1.09) \approx 0.08617769624105$$

e quindi

$$|\log(1.09) - p_2(1.09)| \approx 2.51 \cdot 10^{-5}$$

perfettamente in linea con le stime fornite.

Esercizio facoltativo. Effettuare un programma Matlab che calcoli il valore assunto in 1.09 dal polinomio interpolatore della funzione $\log$ relativamente ai punti 1, 1.1, 1.2. E' buona l'approssimazione fornita dal polinomio interpolatore? (Suggerimento: si noti che

1. il programma è una variante di quanto visto in `esperimento.m`, per una adeguata scelta dei nodi `x`, dei valori `y`, del punto `s`;
2. eliminare la parte relativa ai nodi di Chebyshev-Gauss-Lobatto;
3. il plot non è molto indicativo;
4. siccome il comando `norm(v,inf)` calcola per $v = \{v_i\}$ la quantità

$$\|v\|_\infty = \max_i |v_i|$$

visto che si deve valutare l'errore su un solo punto, il comando `norm(...,inf)` in `esperimento.m` può essere sostituito da `abs(...)` (rifletterci sopra);
5. la funzione `log` è predefinita in Matlab/Octave e quindi non serve ridefinirla come funzione.

Esercizio facoltativo. Aiutandosi con Matlab/Octave eseguire un programma che calcoli l'interpolante della funzione

$$f(x) := \exp(x)$$

relativamente alle ascisse $x = 1$, $x = 1.1$ ed $x = 1.2$. Si calcoli il polinomio di secondo grado p_2 che interpola tali punti, e lo si valuti in $s = 1.09$. Fornire una stima dell'errore e verificarne la bontà rispetto al risultato esatto.

Esercizio facoltativo. Fissati $a = -1$, $b = 1$, $n = 15$, si plottino i nodi di Chebyshev-Gauss. Si suggerisce di utilizzare la function `cheb.m` e per il plottaggio un comando del tipo

```
plot(x,y,'r-o')
```

Ricordiamo che

- l'opzione `r-o`
  1. disegna un cerchietto rosso per ognuno dei punti (x_1, y_1), ..., (x_i, y_i), ..., (x_n, y_n);
  2. per $i = 1, \ldots, n-1$, unisce i punti (x_i, y_i), (x_{i+1}, y_{i+1}) con un segmento rosso.
- essendo importante disegnare solo le ascisse (e non le ordinate), quali ordinate si può porre

```
y=zeros(size(x));
```

Una volta completato l'esercizio si osservi la particolare disposizione dei nodi di Chebyshev-Gauss. Si accumulano verso gli estremi dell'intervallo?

7. Online. Si possono reperire online varie bibliografie dei matematici sopra citati:

1. http://en.wikipedia.org/wiki/Chebyshev
2. http://www-history.mcs.st-andrews.ac.uk/Biographies/Chebyshev.html
3. http://it.wikipedia.org/wiki/Lagrange
4. http://www-history.mcs.st-andrews.ac.uk/Biographies/Lagrange.html

5. http://en.wikipedia.org/wiki/Isaac_Newton

Per quanto riguarda l'interpolazione:

1. http://it.wikipedia.org/wiki/Interpolazione_%28matematica%29
2. http://it.wikipedia.org/wiki/Interpolazione_polinomiale
3. http://en.wikipedia.org/wiki/Interpolation
4. http://en.wikipedia.org/wiki/Polynomial_interpolation
5. http://it.wikipedia.org/wiki/Fenomeno_di_Runge
6. http://en.wikipedia.org/wiki/Runge%27s_phenomenon
7. http://en.wikipedia.org/wiki/Newton_polynomial
8. http://it.wikipedia.org/wiki/Nodi_di_Chebyshev
9. http://en.wikipedia.org/wiki/QR_decomposition
10. http://en.wikipedia.org/wiki/Vandermonde_matrix

8. Frasi celebri.

1. To isolate mathematics from the practical demands of the sciences is to invite the sterility of a cow shut away from the bulls. (Chebyshev)
2. God made the integers; all else is the work of man. (Kronecker).
3. Before we take to sea we walk on land. Before we create we must understand. (Lagrange)
4. When we ask advice, we are usually looking for an accomplice. (Lagrange)
5. It took the mob only a moment to remove his head; a century will not suffice to reproduce it. [said about the chemist Lavoisier] (Lagrange)
6. I do not know. [summarising his life's work] (Lagrange)
7. These astronomers are queer; they won't believe in a theory unless it agrees with their observations. (Lagrange) item How lucky was Newton! In his time the system of the world was still to be discovered! (Lagrange)

RIFERIMENTI BIBLIOGRAFICI

[1] K. Atkinson, *An Introduction to Numerical Analysis*, Wiley, (1989).
[2] J.P. Berrut and L.N.Trefethen, *Barycentric Lagrange interpolation*, (SIAM Review, 2004), http://web.comlab.ox.ac.uk/oucl/work/nick.trefethen/barycentric.pdf.
[3] V. Comincioli, *Analisi Numerica, metodi modelli applicazioni*, Mc Graw-Hill, 1990.
[4] Mac Tutor, Chebyshev biography, http://www-history.mcs.st-andrews.ac.uk/Biographies/Chebyshev.html.
[5] Mac Tutor, Lagrange biography, http://www-history.mcs.st-andrews.ac.uk/Biographies/Lagrange.html.
[6] The MathWorks Inc., *Numerical Computing with Matlab*, `http://www.mathworks.com/moler`.
[7] A. Quarteroni e F. Saleri, *Introduzione al calcolo scientifico*, Springer Verlag, 2006.
[8] A. Quarteroni, R. Sacco e F. Saleri, *Matematica numerica*, Springer Verlag, 1998.
[9] Wikipedia, Fenomeno di Runge, http://it.wikipedia.org/wiki/Fenomeno_di_Runge.
[10] Wikipedia, Interpolazione polinomiale, http://it.wikipedia.org/wiki/Interpolazione_polinomiale.

[11] Wikipedia, Isaac Newton,
http://en.wikipedia.org/wiki/Isaac_Newton.
[12] Wikipedia, Lagrange,
http://it.wikipedia.org/wiki/Lagrange.
[13] Wikipedia, Newton polynomial,
http://en.wikipedia.org/wiki/Newton_polynomial.
[14] Wikipedia, Nodi di Chebyshev,
http://it.wikipedia.org/wiki/Nodi_di_Chebyshev.
[15] Wikipedia, Polynomial interpolation,
http://en.wikipedia.org/wiki/Polynomial_interpolation.
[16] Wikipedia, Runge's phenomenon,
http://en.wikipedia.org/wiki/Runge%27s_phenomenon.
[17] Wikipedia, QR decomposition,
http://en.wikipedia.org/wiki/QR_decomposition.
[18] Wikipedia, Vandermonde matrix,
http://en.wikipedia.org/wiki/Vandermonde_matrix.
[19] Greg von Winckel, `barylag.m`,
http://www.mathworks.com/matlabcentral/fileexchange.