Quadratura numerica

19 febbraio 2009

1 Introduzione

Un classico problema dell'analisi numerica è quello di calcolare l'*integrale definito* di una funzione f in un intervallo avente estremi di integrazione a, b (non necessariamente finiti) cioè

$$I_w(f) := I_w(f,a,b) = \int_a^b f(x)\,w(x)\,dx$$

dove w è una funzione peso in (a,b).

La nostra intenzione è di approssimare $I(f)$ come

$$I_w(f) \approx Q_N(f) := \sum_{i=1}^{N} w_i f(x_i) \tag{1}$$

I termini w_i e $x_i \in [\alpha,\beta]$ sono detti rispettivamente pesi e nodi.

2 Formule interpolatorie

Sia (a,b) l'intervallo di integrazione (non necessariamente limitato), $x_1,\ldots,x_N$ un insieme di N punti a due a due distinti ed $f \in C([a,b])$ una funzione w-integrabile cioè per cui esista finito $I_w(f)$. Se l'intervallo è limitato, per il teorema di Weierstrass e l'integrabilità della funzione peso, questo è vero per qualsiasi funzione continua in quanto

$$\left|\int_a^b f(x)\,w(x)\,dx\right| \le \int_a^b |f(x)|\,w(x)\,dx \le \|f\|_\infty \|w\|_1 < +\infty$$

Se

$$p_{N-1}(x) = \sum_{i=1}^{N} f(x_i) L_i(x)$$

è il polinomio che interpola le coppie $(x_i, f(x_i))$ con $i = 1,\ldots,N$, dove al solito L_i indica l'i-simo polinomio di Lagrange allora

$$
\begin{aligned}
\int_a^b f(x)\,w(x)\,dx &\approx \int_a^b p_{N-1}(x)\,w(x)\,dx \\
&= \int_a^b \sum_{i=1}^{N} f(x_i) L_i(x)\,w(x)\,dx \\
&= \sum_{i=1}^{N} \left(\int_a^b L_i(x)\,w(x)\,dx \right) f(x_i) \qquad (2)
\end{aligned}
$$

per cui, confrontando con la formula (1) abbiamo

$$w_i = \int_a^b L_i(x)\,w(x)\,dx,\ i = 1,\dots,N.$$

Se $f = p_{N-1}$ è un polinomio di grado $N-1$ ovviamente corrisponde col polinomio interpolante p_{N-1} nei nodi a due a due distinti $x_1,\dots,x_N$ e quindi la formula risulta esatta per polinomi di grado inferiore o uguale a $N-1$, cioè

$$I_w(p_{N-1}) = \sum_{i=1}^{N} w_i f(x_i),\ w_i = \int_a^b L_i(x)\,w(x)\,dx,\ i = 1,\dots,N.$$

Si dimostra che se una formula avente quali nodi un insieme $x_1,\dots,x_N$ di punti a due a due distinti è esatta per ogni polinomio di grado $N-1$ allora è interpolatoria.

Infatti se è esatta per ogni polinomio di grado $N-1$ allora lo è in particolare per i polinomi di Lagrange $L_i \in \mathscr{P}_{n-1}$, il che implica che $w_i = \int_a^b L_i(x)\,w(x)\,dx$ e quindi i pesi sono proprio quelli della formula interpolatoria corrispondente nei nodi $x_1,\dots,x_N$.

3 Formule di Newton-Cotes

Si supponga $[a,b]$ un intervallo chiuso e limitato di $\mathbb{R}$ e si ponga $w \equiv 1$. Per semplicità di notazione, in questo caso porremo $I := I_w = I_1$. Il primo esempio di formule interpolatorie che consideriamo sono le *regole* di tipo *Newton-Cotes* chiuse (cf. [13, p.336]) che si ottengono integrando l'interpolante di f in nodi equispaziati

$$x_i = a + \frac{(i-1)\,(b-a)}{N-1},\ i = 1,\dots,N.$$

Alcune classiche regole sono:

1. *regola del trapezio*

   $$I(f) \approx S_1(f) := S_1(f,a,b) := \frac{(b-a)\,(f(a)+f(b))}{2}$$

   avente *grado di precisione* 1, cioè esatta per polinomi di grado inferiore o uguale a 1; si può dimostrare (con un po' di fatica) dal teorema del resto per l'interpolazione polinomiale (cf. [3, p.132]) che l'errore della regola del trapezio [23] è

   $$E_1(f) := I(f) - S_1(f) = \frac{-h^3}{12} f^{(2)}(\xi)$$

   per qualche $\xi \in (a,b)$;

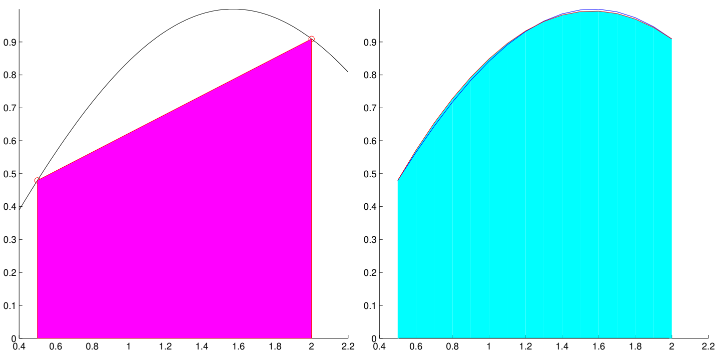

Figura 1: Regola del trapezio e di Cavalieri-Simpson per il calcolo di $\int_{0.5}^{2} \sin(x)\, dx$ (rispettivamente area in magenta e in azzurro).

2. *regola di Cavalieri-Simpson*

$$I(f) \approx S_3(f) := S_3(f,a,b) := \frac{b-a}{6}\left[f(a) + 4f(\frac{a+b}{2}) + f(b)\right]$$

avente *grado di precisione* 3, cioè esatta per polinomi di grado inferiore o uguale a 3; si può dimostrare (non facile!) che l'errore della regola di Cavalieri-Simpson [24] è

$$E_3(f) := I(f) - S_3(f) = \frac{-h^5}{90} f^{(4)}(\xi),\ h = \frac{b-a}{2}$$

per qualche $\xi \in (a,b)$;

Vediamo calcolando i pesi, che in effetti le due formule sono interpolatorie. Partiamo dalla regola del trapezio. Posti $x_1 = a$, $x_2 = b$ abbiamo che

$$L_1(x) = \frac{x-b}{a-b},\ L_2(x) = \frac{x-a}{b-a}$$

e quindi visto che $w \equiv 1$ abbiamo

$$\begin{aligned} w_1 &= \int_a^b L_1(x)\, dx = \int_a^b \frac{x-b}{a-b}\, dx = \frac{1}{a-b}\int_a^b (x-b)\, dx \\ &= \frac{1}{a-b}\frac{(x-b)^2}{2}\Big|_a^b = \frac{1}{a-b}\frac{(x-b)^2}{2}\Big|_a^b = \frac{1}{a-b}\frac{-(a-b)^2}{2} = \frac{b-a}{2} \end{aligned} \tag{3}$$

e

$$\begin{aligned} w_2 &= \int_a^b L_2(x)\, dx = \int_a^b \frac{x-a}{b-a}\, dx = \frac{1}{b-a}\int_a^b (x-a)\, dx \\ &= \frac{1}{b-a}\frac{(x-a)^2}{2}\Big|_a^b = \frac{1}{b-a}\frac{(x-a)^2}{2}\Big|_a^b = \frac{1}{b-a}\frac{(b-a)^2}{2} = \frac{b-a}{2} \end{aligned} \tag{4}$$

Figura 2: Cavalieri e Simpson.

Per quanto riguarda la formula di Cavalieri i ragionamenti sono analoghi. D'altra parte essendo quelle dei trapezi e Simpson regole rispettivamente aventi 2 e 3 punti con grado 2 e 4, allora sono entrambe interpolatorie.

Per ulteriori dettagli si confronti [1, p.252-258], [13, p.333-336]. Qualora le funzioni da integrare non siano sufficientemente derivabili, una stima dell'errore viene fornita dalle formule dell'errore via nucleo di Peano ([1, p.259]). Ricordiamo che per $N \geq 8$ le formule di Newton-Cotes chiuse hanno pesi di segno diverso e sono instabili dal punto di vista della propagazione degli errori (cf. [3, p.196]).

4 Formule di Newton-Cotes composte

Si suddivida l'intervallo (chiuso e limitato) $[a,b]$ in N subintervalli $T_j = [x_j, x_{j+1}]$ tali che $x_j = a + jh$ con $h = (b-a)/N$. Dalle proprietà dell'integrale

$$\int_a^b f(x)\,dx = \sum_{j=0}^{N-1} \int_{x_j}^{x_{j+1}} f(x)\,dx \approx \sum_{j=0}^{N-1} S(f, x_j, x_{j+1}) \tag{5}$$

dove S è una delle regole di quadratura finora esposte (ad esempio $S_3(f)$). Le formule descritte in (5) sono dette *composte*. Due casi particolari sono

1. *formula composta dei trapezi*

$$S_1^{(c)} := h\left[\frac{f(x_0)}{2} + f(x_1) + \ldots + f(x_{N-1}) + \frac{f(x_N)}{2}\right] \tag{6}$$

il cui errore è

$$E_1^{(c)}(f) := I(f) - S_1^{(c)}(f) = \frac{-(b-a)}{12} h^2 f^{(2)}(\xi),\ h = \frac{(b-a)}{N}$$

per qualche $\xi \in (a,b)$;

2. *formula composta di Cavalieri-Simpson* fissati il numero N di subintervalli e i punti $x_k = a + kH/2$ dove

$$H = \frac{b-a}{N}$$

sia

$$I(f) \approx S_3^{(c)}(f) := \frac{H}{6}\left[f(x_0) + 2\sum_{r=1}^{N-1} f(x_{2r}) + 4\sum_{s=0}^{N-1} f(x_{2s+1}) + f(x_{2N})\right]; \qquad (7)$$

il cui errore è

$$E_3^{(c)}(f) := I(f) - S_3^{(c)}(f) = \frac{-(b-a)}{180}\left(\frac{H}{2}\right)^4 f^{(4)}(\xi)$$

per qualche $\xi \in (a,b)$.

4.1 Implementazione Matlab di alcune formule composte

Mostreremo di seguito un'implementazione in Matlab/Octave della formula composta dei trapezi e di Cavalieri-Simpson.

```
function [x,w]=trapezi_composta(N,a,b)

% FORMULA DEI TRAPEZI COMPOSTA.

% INPUT:
% N: NUMERO SUBINTERVALLI.
% a, b: ESTREMI DI INTEGRAZIONE.

% OUTPUT:
% x: NODI INTEGRAZIONE.
% w: PESI INTEGRAZIONE (INCLUDE IL PASSO!).

h=(b-a)/N;              % PASSO INTEGRAZIONE.
x=a:h:b; x=x';          % NODI INTEGRAZIONE.
w=ones(N+1,1);          % PESI INTEGRAZIONE.
w(1)=0.5; w(N+1)=0.5;
w=w*h;
```

La funzione `trapezi_composta` appena esposta calcola i nodi e i pesi della omonima formula composta.
L'unica difficoltà del codice consiste nel calcolo dei pesi `w`. Essendo per loro definizione

$$I(f) \approx S_1^c(f) := \sum_{i=0}^{N} w_i f(x_i) \qquad (8)$$

come pure per (6)

$$S_1^{(c)} := h\left[\frac{f(x_0)}{2} + f(x_1) + \ldots + f(x_{N-1}) + \frac{f(x_N)}{2}\right] \qquad (9)$$

deduciamo che $w_0 = w_N = h/2$ mentre $w_1 = \ldots = w_{N-1} = h$, cosa che giustifica le ultime linee della function `trapezi_composta`.

Si potrebbe usare il comando Matlab `trapz` nella sua implementazione

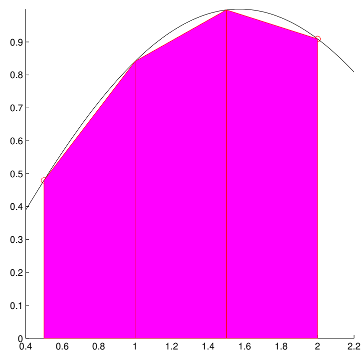

Figura 3: Formula dei trapezi composta per il calcolo di $\int_{0.5}^{2} \sin(x)\,dx$ (area in magenta).

```
>> help trapz

 TRAPZ  Trapezoidal numerical integration.
    Z = TRAPZ(Y) computes an approximation of the integral of Y via
    the trapezoidal method (with unit spacing).  To compute the integral
    for spacing different from one, multiply Z by the spacing increment.

    For vectors, TRAPZ(Y) is the integral of Y.
    ... ... ...
```

e sostituire la parte relativa al calcolo dei pesi con

```
I=h*trapz(fx);
```

Vediamone i dettagli in Matlab (versione 6.1) per il calcolo di

$$\int_0^1 \sin(x)dx = -\cos(1) - (-\cos(0)) = -\cos(1) + 1 \approx 0.45969769413186.$$

sia utilizzando la funzione `trapezi_composta` che `trapz`

```
>> format long;
>> [x,w]=trapezi_composta(10,0,1);
>> fx=sin(x);
>> I_trapezi_composta=w'*fx
I_trapezi_composta =
   0.45931454885798
>> h=(1-0)/10;
>> I_trapz=h*trapz(fx)
I_trapz =
   0.45931454885798
>>
```

Di conseguenza per implementare la regola è del tutto equivalente usare la function `trapezi_composta` o `trapz`. Si osserva che è sbagliato chiamare `trapz` senza il passo h (nell'esempio non si dividerebbe per 10, e invece di 0.45931454885798 avremmo 4.5931454885798).
Evitiamo il diretto utilizzo di `trapz` perchè non presente in alcune vecchie versioni di Octave (ma non nella più recente 2.1.73).

Per quanto riguarda la formula di Cavalieri-Simpson composta

```
function [x,w]=simpson_composta(N,a,b)

% FORMULA DI SIMPSON COMPOSTA.

% INPUT:
% N: NUMERO SUBINTERVALLI.
% a, b: ESTREMI DI INTEGRAZIONE.

% OUTPUT:
% x: INTEGRAZIONE.
% w: PESI INTEGRAZIONE (INCLUDE IL PASSO!).

h=(b-a)/N;              % AMPIEZZA INTERVALLO.
x=a:(h/2):b; x=x';      % NODI INTEGRAZIONE.

w=ones(2*N+1,1);            % PESI INTEGRAZIONE.
w(3:2:2*N-1,1)=2*ones(length(3:2:2*N-1),1);
w(2:2:2*N,1)=4*ones(length(2:2:2*N),1);
w=w*h/6;
```

Similmente alla routine per il calcolo dei nodi e i pesi di `trapezi_composta`, le ultime righe sono le più difficili da capire, ma un confronto con (8) e (7) ne spiega il significato.

Una volta noti il vettore (colonna) x dei nodi e w dei pesi di integrazione, se la funzione f è richiamata da un m-file *f.m*, basta

```
fx=f(x);                    % VALUT. FUNZIONE.
I=w'*fx;                    % VALORE INTEGRALE.
```

per calcolare il risultato fornito dalla formula di quadratura composta.
Ricordiamo che se $\mathbf{w} = (\mathbf{w}_k)_{k=0,\dots,N} \in \mathbb{R}^{N+1}$, $\mathbf{fx} = (f(x_k))_{k=0,\dots,N} \in \mathbb{R}^{N+1}$ sono due vettori colonna allora il prodotto scalare

$$\mathbf{w} * \mathbf{fx} := \sum_{k=0}^{N} \mathbf{w}_k \cdot \mathbf{fx}_k := \sum_{k=0}^{N} \mathbf{w}_k \cdot f(x_k)$$

si scrive in Matlab/Octave come `w'*fx`. Osserviamo che dimensionalmente il prodotto di un vettore $1 \times (N+1)$ con un vettore $(N+1) \times 1$ dà uno scalare (cioè un vettore 1×1).

Applichiamo ora la formula composta di Cavalieri-Simpson all'esempio precedente:

```
>> format long;
>> [x,w]=simpson_composta(5,0,1);
>> fx=sin(x);
>> I_simpson=w'*fx
I_simpson =
   0.45969794982382
>> length(x)
ans =
    11
>>
```

Facciamo ora un altro esempio. Calcoliamo numericamente utilizzando le formule composte sopra citate

$$\int_{-1}^{1} x^{20}\,dx = (1^{21}/21) - (-1)^{21}/21 = 2/21 \approx 0.09523809523810.$$

A tal proposito scriviamo il seguente codice `demo_composte`.

```
N=11;                                %SCEGLIERE DISPARI.
a=-1; b=1;

N_trap=N-1;
[x_trap,w_trap]=trapezi_composta(N_trap,a,b);
fx_trap=x_trap.^20;                  % VALUT. FUNZIONE.
I_trap=w_trap'*fx_trap;              % TRAPEZI COMPOSTA.

N_simpson=(N-1)/2;
[x_simp,w_simp]=simpson_composta(N_simpson,a,b)
fx_simp=x_simp.^20;                  % VALUT. FUNZIONE.
I_simp=w_simp'*fx_simp;              % SIMPSON COMPOSTA.

fprintf('\n \t [TRAPEZI COMPOSTA] [PTS]: %4.0f', length(x_trap));
fprintf('\n \t [TRAPEZI COMPOSTA] [RIS]: %14.14f', I_trap);

fprintf('\n \t [SIMPSON COMPOSTA] [PTS]: %4.0f', length(x_simp));
fprintf('\n \t [SIMPSON COMPOSTA] [RIS]: %14.14f', I_simp);
```

ottenendo

```
[TRAPEZI COMPOSTA] [PTS]:   11
[TRAPEZI COMPOSTA] [RIS]: 0.20462631505024
[SIMPSON COMPOSTA] [PTS]:   11
[SIMPSON COMPOSTA] [RIS]: 0.13949200364447
```

Si può vedere che usando formule di tipo gaussiano (cf. [21], [22]) o di tipo Clenshaw-Curtis (cf. [19], [15], [17]) a parità di valutazioni della funzione f avremmo ottenuto

```
[GAUSS-LEGENDRE  ]: 0.095238095238095649
[CLENSHAW-CURTIS ]: 0.094905176204004307
```

col costo aggiuntivo di dover calcolare tramite complicati algoritmi i pesi e i nodi di Gauss o i nodi di Clenshaw-Curtis via un IFFT [20].

4.2 Esercizio

Si calcolino con la formula composta dei trapezi e di Cavalieri-Simpson, con $N = 5, 7, 9, 11, 13, 15$ i seguenti integrali

$$
\begin{aligned}
\int_{-1}^{1} x^{20}\,dx &= 2/21 \approx 0.095238095238096801 && (10)\\
\int_{-1}^{1} e^{x}\,dx &= e - e^{-1} \approx 2.3504023872876032 && (11)\\
\int_{-1}^{1} e^{-x^2}\,dx &= \mathrm{erf}(1)\cdot\sqrt{(\pi)} \approx 1.4936482656248538 && (12)\\
\int_{-1}^{1} 1/(1+16\cdot x^2)\,dx &= 1/2\cdot\mathrm{atan}(4) \approx 0.66290883183401628 && (13)\\
\int_{-1}^{1} e^{-1/x^2}\,dx &= 2\cdot e^{-1} + 2\cdot e^{-1}\cdot\mathrm{erf}(1)\cdot\pi^{1/2}\cdot e^{-1} - 2\cdot\pi^{1/2} && (14)\\
&\approx 0.17814771178156086 && (15)\\
\int_{-1}^{1} |x|^3\,dx &= 1/2 && (16)\\
\int_{0}^{1} \sqrt{x}\,dx &= 2/3 && (17)\\
\int_{-1}^{1} e^{x}\cdot\sqrt{(1-x)}\,dx &= -2^{1/2}\cdot e^{-1} + 1/2\cdot e * \pi^{1/2} * \mathrm{erf}(2^{1/2}) && (18)\\
&\approx 1.7791436546919095 && (19)
\end{aligned}
$$

Quali delle due formule ha errori relativi inferiori? Quali funzioni risultano più difficili da integrare numericamente?

5 Formule gaussiane

Nelle formule interpolatorie di Newton-Cotes (come ad esempio la regola del Trapezio o di Cavalieri-Simpson) i nodi $x_1, \ldots, x_n$ sono equispaziati e il grado di precisione δ è generalmente uguale a $n-1$ ma in alcuni casi, come per la regola di Cavalieri-Simpson, uguale al numero di nodi n. Vediamo ora formule che a parità di nodi hanno grado di precisione maggiore di n.

Sia $w : (a, b) \to \mathbb{R}$ (non necessariamente limitato) è una funzione peso, cioè tale che (cf. [1, p.206, p.270])

1. w è nonnegativa in (a, b);
2. w è integrabile in $[a, b]$;
3. esista e sia finito
$$\int_a^b |x|^n\, w(x)\,dx$$
per ogni $n \in \mathbb{N}$;

4. se
$$\int_a^b g(x)\,w(x)\,dx$$
per una qualche funzione nonnegativa g allora $g \equiv 0$ in (a,b).

Tra gli esempi più noti ricordiamo

1. *Legendre*: $w(x) \equiv 1$ in $[a,b]$ limitato;
2. *Jacobi*: $w(x) = (1-x)^{\alpha}\,(1+x)^{\beta}$ in $(-1,1)$ per $\alpha,\ \beta \geq -1$;
3. *Chebyshev*: $w(x) = \frac{1}{\sqrt{1-x^2}}$ in $(-1,1)$;
4. *Laguerre*: $w(x) = \exp(-x)$ in $[0,\infty)$;
5. *Hermite*: $w(x) = \exp(-x^2)$ in $(-\infty,\infty)$;

Si supponga ora di dover calcolare per qualche funzione $f:(a,b) \to \mathbb{R}$
$$I(f,w) := \int_a^b f(x)\,w(x)\,dx.$$
Il problema è evidentemente più generale di quello di calcolare un integrale del tipo $\int_a^b f(x)dx$ con $f \in C([a,b])$, $[a,b]$ limitato, visto che l'integranda $f\,w$ non é necessariamente continua in $[a,b]$ (si consideri ad esempio il peso di Chebyshev che ha una singolarità in $a=-1$, $b=1$) oppure può succedere che l'intervallo sia illimitato come nel caso del peso di Laguerre o Hermite.

Esistono nuovamente $x_1,\ldots,x_n$ e pesi $w_1,\ldots,w_n$ (detti di *Gauss-nome funzione peso*) per cui le relative formule di quadratura di tipo interpolatorio abbiano grado di precisione $\delta > n$, cioè calcolino esattamente
$$\int_a^b p_m(x)\,w(x)\,dx$$
per $m > n$?

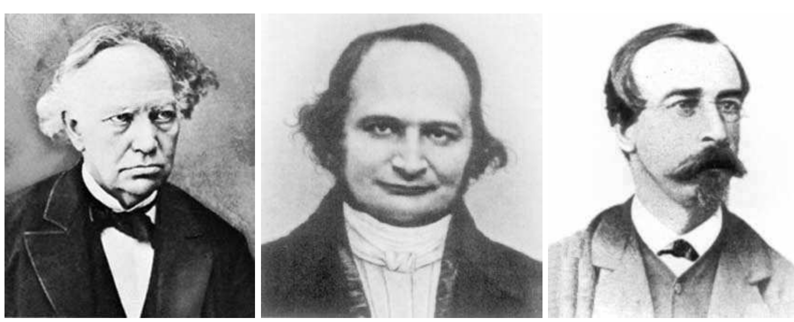

Figura 4: Hermite, Jacobi e Laguerre.

La risposta è affermativa, come si può vedere in [1, p.272].

Teorema 5.1 *Per ogni $n \leq 1$ esistono e sono unici dei nodi $x_1, \dots, x_n$ e pesi $w_1, \dots, w_n$ per cui il grado di precisione sia $2n-1$. I nodi sono gli zeri del polinomio ortogonale di grado n,*

$$\phi_n(x) = A_n \cdot (x - x_1) \cdot \ldots \cdot (x - x_n).$$

Dimostrazione [3, p.209]. Per prima cosa mostriamo che in effetti con tale scelta dei nodi la formula interpolatoria ha grado di precisione $2n-1$, che i pesi sono univocamente determinati e positivi.

Sia $p_{2n-1} \in \mathscr{P}_{2n-1}$ e $q_{n-1}, r_{n-1} \in \mathscr{P}_{n-1}$ tali che

$$p_{2n-1} = q_{n-1}\phi_n + r_{n-1}.$$

Allora poichè

- $\int_a^b q_{n-1}(x)\phi_n(x)w(x)dx = 0$, poichè ϕ_n è il polinomio ortogonale rispetto w di grado n;
- la formula è interpolatoria, per cui esatta per ogni polinomio di grado $n-1$ in quanto basata su n punti a due a due distinti;
- se x_k è uno zero di ϕ_n allora

$$p_{2n-1}(x_k) = q_{n-1}(x_k)\phi_n(x_k) + r_{n-1}(x_k) = r_{n-1}(x_k).$$

abbiamo

$$\begin{aligned} \int_a^b p_{2n-1}(x)w(x)dx &= \int_a^b q_{n-1}(x)\phi_n(x)w(x)dx + \int_a^b r_{n-1}(x)w(x)dx \\ &= 0 + \int_a^b r_{n-1}(x)w(x)dx \\ &= \sum_{k=1}^{n} w_k r_{n-1}(x_k) \\ &= \sum_{k=1}^{n} w_k p_{2n-1}(x_k) \end{aligned} \tag{20}$$

per cui la formula di Gauss ha grado di precisione $2n-1$. Inoltre, come dimostrato da Stieltjes nel 1884, i pesi sono positivi, poichè in particolare la formula è esatta per ognuno dei quadrati dei polinomi di Lagrange relativo ai punti $x_1, \dots, x_n$ per cui

$$0 < \int_a^b L_j^2(x)w(x)dx = \sum_{k=1}^{n} w_k L_j^2(x_k) = w_j.$$

Se esistesse un'altra formula interpolatoria con grado di precisione $2n-1$ e avesse nodi $\{\tilde{x}_j\}_{j=1,\dots,n}$ e pesi $\{\tilde{w}_j\}_{j=1,\dots,n}$ per prima cosa i pesi sarebbero positivi poichè il grado di precisione è $2n-1$ e quindi sarebbe esatta per il j-simo polinomio di Lagrange $\tilde{L}_j$ da cui

$$0 < \int_a^b \tilde{L}_j^2(x)\,w(x)\,dx = \sum_{k=1}^{n} \tilde{w}_k \tilde{L}_j^2(\tilde{x}_k) = \tilde{w}_j.$$

D'altra parte se $\tilde{L}_j$ è il j-simo polinomio di Lagrange, poichè ϕ_n è il polinomio ortogonale di grado n rispetto al peso w, e $\tilde{w}_j > 0$ abbiamo che da

$$0 = (\phi_n, \tilde{L}_j)_w = \sum_{k=1}^{n} \tilde{w}_k \tilde{L}_j(\tilde{x}_k)\phi_n(\tilde{x}_k) = \tilde{w}_j \cdot \phi_n(\tilde{x}_j)$$

necessariamente $x_j = \tilde{x}_j$ e visto che quindi $L_j = \tilde{L}_j$ ricaviamo anche

$$w_j = \int_a^b L_j^2(x)\,w(x)\,dx = \int_a^b \tilde{L}_j^2(x)\,w(x)\,dx = \tilde{w}_j$$

per cui la formula gaussiana cercata è unica. ∎

5.1 Formule gaussiane in Matlab

Eccetto che in pochi casi (come ad esempio per la funzione peso di Chebyshev), non esistono formule esplicite per l'individuazione di tali nodi e pesi. Una volta venivano tabulati, oggi si consiglia di applicare del software che si può trovare ad esempio nella pagina di Walter Gautschi

http://www.cs.purdue.edu/people/wxg

Fissato un peso, ad esempio quello di Jacobi, si cercano per prima cosa i *coefficienti di ricorrenza*, che possono essere calcolati con l' m-file *r_jacobi.m* nel sito di W. Gautschi. La sintassi è la seguente

```
ab=r_jacobi(N,a,b)
```

Come si vede dai commenti al file

```
% R_JACOBI Recurrence coefficients for monic Jacobi polynomials.
%
%    ab=R_JACOBI(n,a,b) generates the first n recurrence
%    coefficients for monic Jacobi polynomials with parameters
%    a and b. These are orthogonal on [-1,1] relative to the
%    weight function w(t)=(1-t)^a(1+t)^b. The n alpha-coefficients
%    are stored in the first column, the n beta-coefficients in
%    the second column, of the nx2 array ab. The call ab=
%    R_JACOBI(n,a) is the same as ab=R_JACOBI(n,a,a) and
%    ab=R_JACOBI(n) the same as ab=R_JACOBI(n,0,0).
%
%    Supplied by Dirk Laurie, 6-22-1998; edited by Walter
%    Gautschi, 4-4-2002.
```

a, b corrispondono rispettivamente all'α e β delle formule di Jacobi (e non agli estremi di integrazione!). I coefficienti di ricorrenza sono immagazzinati nella variabile ab. Ricordiamo che se $\{\phi_n(x)\}$ è una famiglia di polinomi ortogonali su $[a,b]$, rispetto ad una funzione peso $w(x)$, allora per $n \geq 1$

$$\phi_{n+1}(x) = \alpha_n (x - \beta_n)\phi_n(x) - \gamma_n \phi_{n-1}(x)$$

ove, detto A_n il coefficiente di grado n in ϕ_n, si ha $\alpha_n = A_{n+1}/A_n$,

$$\beta_n = \frac{(x\phi_n, \phi_n)}{\|\phi_n\|^2}, \gamma_n = \frac{(\phi_n, x\phi_{n-1})}{\|\phi_n\|^2},$$

con $(f,g) = \int_a^b f(x)g(x)\,dx$. Per ulteriori dettagli sui polinomi ortogonali si confronti [1, p.270-283], [3, p.978-985], [13, p.375-401]. A questo punto si chiama la funzione *gauss.m* (reperibile nuovamente presso lo stesso sito di W. Gautschi)

```
% GAUSS Gauss quadrature rule.
%
%    Given a weight function w encoded by the nx2 array ab of the
%    first n recurrence coefficients for the associated orthogonal
%    polynomials, the first column of ab containing the n alpha-
%    coefficients and the second column the n beta-coefficients,
%    the call xw=GAUSS(n,ab) generates the nodes and weights xw of
%    the n-point Gauss quadrature rule for the weight function w.
%    The nodes, in increasing order, are stored in the first
%    column, the n corresponding weights in the second column, of
%    the nx2 array xw.
%
function xw=gauss(N,ab)
N0=size(ab,1); if N0<N, error('input array ab too short'), end
J=zeros(N);
for n=1:N, J(n,n)=ab(n,1); end
for n=2:N
  J(n,n-1)=sqrt(ab(n,2));
  J(n-1,n)=J(n,n-1);
end
[V,D]=eig(J);
[D,I]=sort(diag(D));
V=V(:,I);
xw=[D ab(1,2)*V(1,:)'.^2];
```

che per un certo N, uguale al massimo al numero di righe della matrice di coefficienti di ricorrenza ab, fornisce nodi x e pesi w immagazzinati in una matrice che ha quale prima colonna x e quale seconda colonna w. Osserviamo che dall'help si evince che i polinomi ortogonali sono monici e quindi $\alpha_n = 1$. Inoltre $(ab)_k = (\beta_k, \gamma_k)$.

La descrizione di perchè tale software fornisca il risultato desiderato è complicata ma può essere trovata nella monografia di W. Gautschi sui polinomi ortogonali, per Acta Numerica.

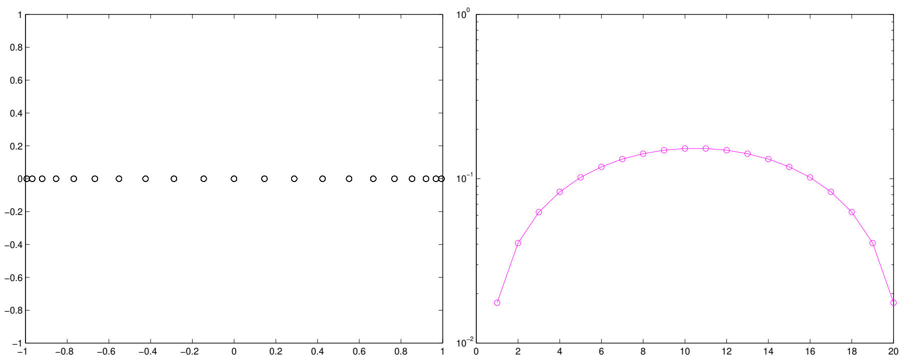

Figura 5: Grafico che illustra la distribuzione dei 20 nodi e i 20 pesi di Gauss-Legendre nell'intervallo $[1,1]$

.

Conseguentemente per trovare i nodi e pesi relativi alla formula di Gauss-Legendre in (a,b), che è una formula di tipo Gauss-Jacobi per $\alpha = 0$ e $\beta = 0$, nonchè calcolare con essi degli integrali di una funzione f, si procede come segue.

5.2 Una demo di Gauss-Jacobi

Salviamo nel file `integrazione_gauss_jacobi.m`

```
function [I_jac,x_jac,w_jac]=integrazione_gauss_jacobi(N,ajac,bjac,a,b,f)

% INPUT:
%
% N: GRADO DI GAUSS-JACOBI. SE SI VUOLE CALCOLARE I(f) CON
%                I(f)=int_a^b f(x) dx
%    PORRE ajac=0, bjac=0 (FUNZIONE PESO DI LEGENDRE w(x)=1).
% ajac, bjac: PARAMETRI DI GAUSS-JACOBI, CIOE' SI INTEGRA:
%         I(fw)=int_a^b f(x) w(x) dx
%    CON w(x)=(1-x)^ajac (1+x)^bjac.
% a,b: ESTREMI INTERVALLO INTEGRAZIONE.
% f: FUNZIONE DA INTEGRARE.
%
% OUTPUT:
%
% I_jac: VALORE DI I(fw) APPROSSIMATO DA gauss_jacobi. PER ajac=0,
%        bjac=0 CORRISPONDE A I(f).
% x_jac: NODI GAUSS-JACOBI.
% w_jac: PESI GAUSS-JACOBI.

% ROUTINES ESTERNE: r_jacobi, gauss.
```

```
ab_jac=r_jacobi(N,ajac,bjac);        % TERM. RICORSIVI.
xw_jac=gauss(N,ab_jac);              % NODI E PESI IN MATRICE.
x_jac=xw_jac(:,1);                   % NODI GAUSS-LEGENDRE [-1,1].
x_jac_ab=((a+b)/2)+((b-a)/2)*x_jac;  % NODI GAUSS-LEGENDRE [a,b].
w_jac=xw_jac(:,2);                   % PESI GAUSS-LEGENDRE [-1,1].
w_jac_ab=((b-a)/2)*w_jac;            % PESI GAUSS-LEGENDRE [a,b].
fx_jac_ab=feval(f,x_jac_ab);         % VALUTAZIONE FUNZIONE.
I_jac=w_jac_ab'*fx_jac_ab;           % VALORE INTEGRALE.
```

e nel file *f.m* delle funzioni su cui effettueremo dei test. Un esempio è

```
function fx=f(x)

fx=x.^20;

% ALCUNE FUNZIONI CHE FANNO PARTE DEL SET STUDIATO NELL'ARTICOLO:
% "IS GAUSS QUADRATURE BETTER THAN CLENSHAW-CURTIS?"
% DI L.N. TREFETHEN.
% fx=exp(x);

% fx=exp(-x.^2);
% fx=1./(1+16*(x.^2));
% fx=exp(-x.^(-2));
% fx=abs(x); fx=fx.^3;
% fx=x.^(0.5);
% fx=exp(x).*(sqrt(1-x));
```

Altre funzioni test [15] sono

```
fx = x.^20
fx = exp(x)
fx = exp(-x.^2)
fx = 1./(1+16*(x.^2))
fx = exp(-x.^(-2))
fx = (abs(x)).^3;
fx=x.^(0.5);
fx=exp(x).*(sqrt(1-x));
```

con $a = -1$, $b = 1$ e per cambiare il tipo di funzione, basta modificare la posizione dei caratteri %.

Scaricando dalla pagina web del corso i files `integrazione_gauss_jacobi.m`, `f.m` e dalla pagina di W. Gautschi `r_jacobi.m` e `gauss.m` possiamo fare alcuni esperimenti.
Una lista di possibili tests è la seguente:

$$\int_{-1}^{1} x^{20}\,dx = 2/21 \approx 0.095238095238096801 \tag{21}$$

$$\int_{-1}^{1} e^{x}\,dx = e - e^{-1} \approx 2.3504023872876032 \tag{22}$$

$$\int_{-1}^{1} e^{-x^2}\,dx = \mathrm{erf}(1)\cdot\sqrt{(}\pi) \approx 1.4936482656248538 \tag{23}$$

$$\int_{-1}^{1} 1/(1+16\cdot x^2)\,dx = 1/2\cdot \mathrm{atan}(4) \approx 0.66290883183401628 \tag{24}$$

$$\int_{-1}^{1} e^{-1/x^2}\,dx = 2\cdot e^{-1} + 2\cdot e^{-1}\cdot \mathrm{erf}(1)\cdot \pi^{1/2}\cdot e^{-1} - 2\cdot\pi^{1/2} \tag{25}$$

$$\approx 0.17814771178156086 \tag{26}$$

$$\int_{-1}^{1} |x|^3\,dx = 1/2 \tag{27}$$

$$\int_{-1}^{1} \sqrt{x}\,dx = 2/3 \tag{28}$$

$$\int_{-1}^{1} e^{x}\cdot\sqrt{(}1-x)\,dx = -2^{1/2}\cdot e^{-1} + 1/2\cdot e * \pi^{1/2} * \mathrm{erf}(2^{1/2}) \tag{29}$$

$$\approx 1.7791436546919095 \tag{30}$$

Alcune note prima di cominciare.

1. Si osservi che per ottenere i nodi e pesi di Gauss-Legendre in (a,b) da quelli di Gauss-Jacobi in $(-1,1)$ abbiamo effettuato uno *scaling*: se $x_{i,[-1,1]}$ sono i nodi di Gauss-Jacobi in $[-1,1]$ allora i nodi di Gauss-Jacobi $x_{i,[a,b]}$ in $[a,b]$ sono

$$x_{i,[a,b]} = ((a+b)/2) + ((b-a)/2)\cdot x_{i,[-1,-1]};$$

2. se $w_{i,[-1,1]}$ sono i pesi di Gauss-Jacobi in $[-1,1]$ allora i pesi di Gauss-Jacobi $w_{i,[a,b]}$ in $[a,b]$ sono

$$w_{i,[a,b]} = ((b-a)/2)\cdot w_{i,[-1,1]}.$$

   L'idea è la seguente. Dovendo le formule essere esatte per le costanti come la funzione $f \equiv 1$, nel caso della funzione peso di Legendre $w \equiv 1$

$$\sum_i w_{i,[a,b]} = \int_a^b w(x)\,dx = b-a$$

   mentre nel caso di $a=-1$, $b=1$ derivante dalla funzione peso di Jacobi abbiamo

$$\sum_i w_{i,[-1,1]} = \int_{-1}^{1} w(x)\,dx = 2;$$

   si ha quindi l'intuizione che i pesi in (a,b) siano quelli in $(-1,1)$ moltiplicati per $\frac{b-a}{2}$.

3. Nonostante l'introduzione riguardante nodi e pesi in $[a, b]$ non necessariamente uguale a $[-1, 1]$, alla fine eseguiremo test esclusivamente in quest'ultimo intervallo. Qualora necessario, basta aggiungere nuove funzioni matematiche al file `f.m`, modificare adeguatamente a, b, fornire il relativo risultato esatto in `exact_results.m` e testare la formula gaussiana. Per avere il risultato con alta precisione si usi la funzione `quadl.m` o `quad8.m` di Matlab oppure `integrazione_gauss_jacobi.m` con $N = 500$;

4. l'intervallo tipico di Gauss-Legendre è $[a, b]$ (chiuso), mentre per Gauss-Jacobi l'intervallo è tipicamente (a, b) (aperto),poichè in generale gli esponenti della funzione peso di Jacobi possono essere negativi; per Gauss-Legendre il problema non sussiste, visto che la funzione peso è continua, e i punti a, b sono un insieme *trascurabile*.

Se ora testiamo il primo esempio, cioè il calcolo di

$$\int_{-1}^{1} x^{20}\, dx \approx 0.09523809523809523300$$

otteniamo

```
>> format long;
>> N=11; ajac=0; bjac=0; a=-1; b=1;
>> [I_jac,x_jac,w_jac]=integrazione_gauss_jacobi(N,ajac,bjac,a,b,@f);
>> I_jac
I_jac =
   0.09523809523810
>> length(x_jac)
ans =
    11
>> fprintf('\n \t [GAUSS-LEGENDRE]: \%15.20f',I_jac);

          [GAUSS-LEGENDRE]: 0.09523809523809564900
>> 0.09523809523809523300-0.09523809523809564900
ans =
   -4.163336342344337e-016
>>
```

5.3 Facoltativo. Una implementazione di Clenshaw-Curtis

La quadrature di tipo Clenshaw-Curtis si basa su un'espansione dell'integranda in termini di polinomi ortogonali di Chebyshev (cf. [19], [15], [17]).
Usiamo una implementazione di Waldvogel [17]

```
function [x,w]=clenshaw_curtis(n,a,b)

N=[1:2:n-1]'; l=length(N); m=n-l; K=[0:m-1]';
```

```
g0=-ones(n,1); g0(1+l)=g0(1+l)+n; g0(1+m)=g0(1+m)+n;
g=g0/(n^2-1+mod(n,2));
end_N=length(N);
v0=[2./N./(N-2); 1/N(end_N); zeros(m,1)];
end_v0=length(v0);
v2=-v0(1:end_v0-1)-v0(end_v0:-1:2);
wcc=ifft(v2+g); weights=[wcc;wcc(1,1)];
k=0:n; nodes=(cos(k*pi/n))';

x=(a+b)/2+((b-a)/2)*nodes;
w=weights*((b-a)/2);
```

Noti i nodi e i pesi di Clenshaw-Curtis, definiamo la function `integrazione_clenshaw_curtis`

```
function [I_cc,x_cc,w_cc]=integrazione_clenshaw_curtis(N_cc,a,b,f)

% INTEGRAZIONE DI CLENSHAW-CURTIS.

% INPUT:
% N_cc: GRADO DI CLENSHAW-CURTIS, CIOE' NUMERO DI NODI MENO UNO.
% a, b: ESTREMI DI INTEGRAZIONE.
% f   : FUNZIONE DA INTEGRARE.

% OUTPUT:
% I_cc: APPROSSIMAZIONE DELL'INTEGRALE CON FORMULA DI CLENSHAW-CURTIS.
% x_cc: NODI DI CLENSHAW-CURTIS.
% w_cc: PESI DI CLENSHAW-CURTIS.

% ROUTINES ESTERNE: clenshaw_curtis.

[x_cc,w_cc]=clenshaw_curtis(N_cc,a,b);  % NODI E PESI DI CL.-CURTIS.
fx_cc=feval(f,x_cc);                    % VALUTAZIONE FUNZIONE.
I_cc=w_cc'*fx_cc;                       % VALORE INTEGRALE.
```

Applichiamola ora al calcolo di

$$\int_{-1}^{1} x^{20}\, dx = 2/21 \approx 0.095238095238095233 00.$$

```
>> N_cc=10; a=-1; b=1;
>> [I_cc,x_cc,w_cc]=integrazione_clenshaw_curtis(N_cc,a,b,@f);
>> fprintf('\n \t [CLENSHAW-CURTIS]: %15.20f',I_cc);

          [CLENSHAW-CURTIS]: 0.09490517620400430700
>> 0.09490517620400430700-2/21
ans =
   -3.329190340909255e-004
>> length(x_cc)
```

```
ans =
    11
>>
```

La formula ha 11 nodi e pesi, ed ha un'errore assoluto di circa $3.32 \cdot 10^{-4}$. Osserviamo il passaggio della funzione f attraverso `@f`.

Dalla figura con i 6 grafici di errore, si vedono le performances delle formule sopracitate quando applicate a calcolare

$$\int_{-1}^{1} x^{20}\,dx \qquad \text{(a sinistra, prima riga)} \tag{31}$$

$$\int_{-1}^{1} e^{x}\,dx \qquad \text{(a destra, prima riga)} \tag{32}$$

$$\int_{-1}^{1} e^{-x^2}\,dx \qquad \text{(a sinistra, seconda riga)} \tag{33}$$

$$\int_{-1}^{1} 1/(1+16\cdot x^2)\,dx \qquad \text{(a destra, seconda riga)} \tag{34}$$

$$\int_{-1}^{1} e^{-1/x^2}\,dx \qquad \text{(a sinistra, terza riga)} \tag{35}$$

$$\int_{-1}^{1} |x|^3\,dx \qquad \text{(a destra, terza riga)} \tag{36}$$

Da osservare che fissato N, le formule composte hanno N nodi mentre le gaussiane e Clenshaw-Curtis ne hanno $N-1$.

5.4 Un'integrale con funzione peso

Consideriamo ora l'integrale

$$\int_{-1}^{1} \exp(x)\sqrt{1-x}\,dx = 1.7791436546919097925911790299941. \tag{37}$$

Tale risultato è stato ottenuto usando il comando simbolico di Matlab 6.5 (non funziona in Octave, vedere in alternativa il programma Maxima!!)

```
>> syms x
>> int('(exp(x)) * ( (1-x)^(0.5) )',-1,1)
ans =
1.7791436546919097925911790299941
```

Si capisce che

1. `syms x` rende la variabile `x` di tipo simbolico (e non numerico!);
2. il termine

   ```
   int('(exp(x)) * ( (1-x)^(0.5) )',-1,1)
   ```

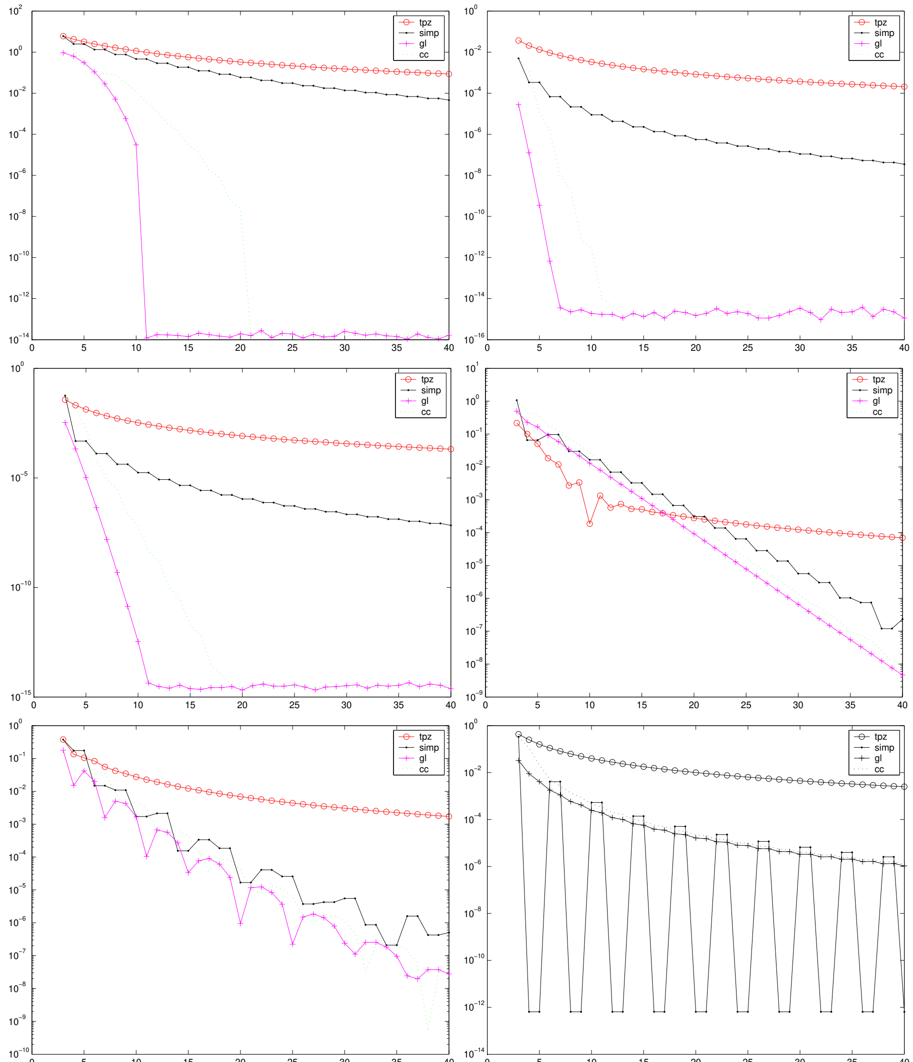

Figura 6: Grafico che illustra l'errore delle formule di quadratura dei trapezi composta, Cavalieri-Simpson composta, Gauss-Legendre e Clenshaw-Curtis su 6 funzioni test.

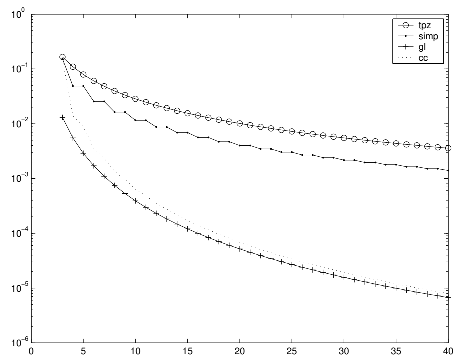

Figura 7: Grafico che illustra l'errore delle formule di quadratura dei trapezi composta, Cavalieri-Simpson composta, Gauss-Legendre e Clenshaw-Curtis sulla funzione test (37).

calcola simbolicamente l'integrale

$$\int_{-1}^{1} \exp(x)\sqrt{1-x}\,dx.$$

Dal grafico in figura 37, si vede come pure la formula di Gauss-Legendre non abbia una grande performance.

E' immediato osservare che $w(x) = \sqrt{1-x}$ è un peso di Gauss-Jacobi

$$w(t) = (1-t)^{\alpha}(1+t)^{\beta}$$

per $\alpha = 1/2$ e $\beta = 0$.

Infatti se $w(x) = \sqrt{1-x}$, allora $g(x) = \exp(x)\,w(x)$ il che corrisponde a usare Gauss-Jacobi con $f(x) = \exp(x)$. Quindi paragoniamo le formule gaussiane (il codice funziona in Matlab 6.1 come pure nella release 2.1.73 di Octave

```
>> a=-1; b=1;
>> [I_jac,x_jac,w_jac]=integrazione_gauss_jacobi(10,1/2,0,a,b,@exp);
>> fprintf('\n \t [GAUSS-JACOBI]: %15.20f',I_jac);

          [GAUSS-JACOBI]: 1.77914365469190930000
>> 1.7791436546919097925911790299941-1.77914365469190930000
ans =
    4.440892098500626e-016
>> length(x_jac)
ans =
    10
>> [I_jac,x_jac,w_jac]=integrazione_gauss_jacobi(10,0,0,-1,1,inline('exp(x).*sqrt(1-x)'));
>> fprintf('\n \t [GAUSS-LEGENDRE]: %15.20f',I_jac)

          [GAUSS-LEGENDRE]: 1.77984112101478020000
```

```
>> 1.77914365469190979259117902999941-1.77984112101478020000
ans =
 -6.9747e-004
>> length(x_jac)
ans =
    10
>>
```

Entrambe le formule hanno lo stesso numero di nodi (e pesi), come si vede dalla riga di comando

```
>> length(x_jac)
ans =
    10
```

ma offrono risultati diversi, con un errore assoluto di circa $4.44 \cdot 10^{-16}$ per Gauss-Jacobi con $a = 1/2$, $b = 0$ e di $2.52 \cdot 10^{-3}$ per Gauss-Legendre (cioè Gauss-Jacobi con $a = 0$, $b = 0$).

5.5 Esperimenti in Octave

Per via della compatibilità Matlab/Octave, ricapitoliamo quanto visto in questa lezione in Octave (versione 2.1.73). Osserviamo che per Octave, `octave:8>`, `octave:9>`, ... corrisponde in Matlab a >>.

```
octave:8> format long;
octave:9> [x,w]=trapezi_composta(10,0,1);
octave:10> fx=sin(x);
octave:11> I_trapezi_composta=w'*fx
I_trapezi_composta = 0.459314548857976
octave:12> h=(1-0)/10;
octave:13> I_trapz=h*trapz(fx)
I_trapz = 0.459314548857976
octave:14> fx=f(x);
octave:15> I=w'*fx
I = 0.0633941318588326
octave:16> demo_composte
x_simp =

   -1.00000000000000e+00
   -8.00000000000000e-01
   -6.00000000000000e-01
   -4.00000000000000e-01
   -2.00000000000000e-01
    5.55111512312578e-17
    2.00000000000000e-01
    4.00000000000000e-01
    6.00000000000000e-01
    8.00000000000000e-01
    1.00000000000000e+00
```

```
w_simp =

  0.0666666666666667
  0.2666666666666667
  0.1333333333333333
  0.2666666666666667
  0.1333333333333333
  0.2666666666666667
  0.1333333333333333
  0.2666666666666667
  0.1333333333333333
  0.2666666666666667
  0.0666666666666667


         [TRAPEZI COMPOSTA] [PTS]:   11
         [TRAPEZI COMPOSTA] [RIS]: 0.20462631505024
         [SIMPSON COMPOSTA] [PTS]:   11
         [SIMPSON COMPOSTA] [RIS]: 0.13949200364447
octave:17> % ESEMPIO GAUSS JACOBI.
octave:18> format long;
octave:19> [I_jac_x,x_jac,w_jac]=integrazione_gauss_jacobi(N,ajac,bjac,a,b,@f)
I_jac_x = 0.0952380952380953
x_jac =

   -9.78228658146057e-01
   -8.87062599768095e-01
   -7.30152005574049e-01
   -5.19096129206812e-01
   -2.69543155952345e-01
    1.25162426693849e-17
    2.69543155952345e-01
    5.19096129206812e-01
    7.30152005574049e-01
    8.87062599768095e-01
    9.78228658146057e-01

w_jac =

  0.0556685671161738
  0.1255803694649046
  0.1862902109277342
  0.2331937645919905
  0.2628045445102464
  0.2729250867779003
  0.2628045445102470
  0.2331937645919902
  0.1862902109277343
  0.1255803694649046
```

```
  0.0556685671161738

octave:20> % ESEMPIO CLENSHAW-CURTIS.
octave:21> N_cc=10; a=-1; b=1;
octave:22> [I_cc,x_cc,w_cc]=integrazione_clenshaw_curtis(N_cc,a,b,@f);
octave:23> fprintf('\n \t [CLENSHAW-CURTIS]: %15.20f',I_cc)

         [CLENSHAW-CURTIS]: 0.09490517620400433507
octave:24>  0.09490517620400433507-2/21
ans =  -3.32919034090898e-04
octave:25> length(x_cc)
ans = 11
octave:26> % ESEMPIO GAUSS-JACOBI (-1/2,0)
octave:27> % PER exp(x).*sqrt(1-x.^2)
octave:28> a=-1; b=1;
octave:29> [I_jac,x_jac,w_jac]=integrazione_gauss_jacobi(10,1/2,0,a,b,@exp);
octave:30> fprintf('\n\t [GAUSS-JACOBI]: %15.20f',I_jac)

         [GAUSS-JACOBI]: 1.77914365469190971503
octave:31> 1.7791436546919097925911790299941-1.77914365469190971503
ans = 0
octave:32> format long e
octave:33> 1.7791436546919097925911790299941-1.77914365469190971503
ans =  0.00000000000000e+00
octave:34> format short
octave:35> length(x_jac)
ans = 10
octave:36> % ESEMPIO GAUSS-LEGENDRE (CIOE' GAUSS-JACOBI (0,0)).
octave:37> % PER exp(x).*sqrt(1-x.^2)
octave:38> [I_jac,x_jac,w_jac]=integrazione_gauss_jacobi(10,0,0,...
            a,b,inline('exp(x).*sqrt(1-x)'));
octave:39> [I_jac,x_jac, w_jac]=integrazione_gauss_jacobi(10,0,0,...
        -1,1,inline('exp(x).*sqrt(1-x)'));
octave:40> fprintf('\n \t [GAUSS-LEGENDRE: %15.20f]',I_jac)

         [GAUSS-LEGENDRE: 1.77984112101478153534
octave:41> 1.7791436546919097925911790299941-I_jac
ans =  -6.9747e-04
octave:42> length(x_jac)
ans = 10
```

6 Formule dell'errore

Per quanto riguarda gli errori compiuti da alcune delle formula di quadratura discusse si ha (cf.[1, p.264])

Teorema 6.1 *Sia*

$$I(f) \approx L_n(f) = \sum_{i=0}^{n} w_{i,n} f(x_{i,n})$$

una regola di Newton-Cotes.

1. *se n è pari e $f \in C^{(n+2)}([a,b])$ allora*

$$I(f) - L_n(f) = C_n\, h^{n+3} f^{(n+2)}(\eta),\ \eta \in (a,b)$$

   con

$$C_n = \frac{1}{(n+2)!}\int_0^n \mu^2(\mu-1)\dots(\mu-n)\,d\mu;$$

2. *se n è dispari e $f \in C^{(n+1)}([a,b])$ allora*

$$I(f) - L_n(f) = C_n\, h^{n+2} f^{(n+1)}(\eta),\ \eta \in (a,b)$$

   con

$$C_n = \frac{1}{(n+1)!}\int_0^n \mu(\mu-1)\dots(\mu-n)\,d\mu;$$

Si osserva facilmente che quanto visto in precedenza per la regola del trapezio e la regola di Cavalieri-Simpson, è consistente con questi due teoremi. Inoltre, da questi ultimi, si ottengono gli errori di regole composte da un numero di punti minore o uguale a 7 (per motivi di stabilità non si suggeriscono regole con più punti).
Per quanto concerne le formule gaussiane (cf.[1, p.272], cf.[3, p.264], cf.[5, p.344]) ricordiamo il seguente teorema di Markov

Teorema 6.2 *Sia $f \in C^{(2n)}(a,b)$ con (a,b) compatto e supponiamo*

$$I_w(f) = \int_a^b f(x)w(x)dx \approx L_n(f) = \sum_{i=1}^n w_{i,n} f(x_{i,n})$$

sia una formula gaussiana rispetto alla funzione peso w. Allora

$$E_n(f) := I_w(f) - L_n(f) = \frac{\gamma_n}{A_n^2 (2n)!} f^{(2n)}(\eta),\ \eta i n(a,b)$$

dove A_n è il coefficiente di grado massimo del polinomio ortogonale ϕ_n di grado n, $\gamma_n = \int_a^b \phi_n^2(x)w(x)dx$.

In particolare, se $w \equiv 1$, $[a,b] \equiv [-1,1]$ allora

$$E_n(f) = \frac{2^{2n+1}(n!)^4}{(2n+1)[(2n)!]^3} f^{(2n)}(\eta),\ \eta \in (-1,1).$$

7 Stabilità di una formula di quadratura

Sia $[a,b]$ un intervallo compatto e w una funzione peso in (a,b). Inoltre supponiamo

$$I_n(f) = \sum_{j=1}^n w_j f_j,\ \text{ con } f_j = f(x_j) \tag{38}$$

e che invece di $\{f_j\}_j$ si disponga di una loro approssimazione $\{\tilde{f}_j\}_j$. Di conseguenza al posto di $I_n(f)$ si calcola

$$\tilde{I}_n(f) = \sum_{j=1}^{n} w_j \tilde{f}_j,$$

ed è

$$\begin{aligned} |I_n(f) - \tilde{I}_n(f)| &= |\sum_{j=1}^{n} w_j (f_j - \tilde{f}_j)| \le \sum_{j=1}^{n} |w_j||f_j - \tilde{f}_j| \\ &\le \left(\sum_{j=1}^{n} |w_j|\right) \cdot \max_j |f_j - \tilde{f}_j| \end{aligned} \tag{39}$$

Quindi la quantità

$$\sum_{j=1}^{n} |w_j|$$

è un indice di stabilità della formula di quadratura.

Si dimostra che in effetti l'operatore lineare $I_n : C([a,b]) \to \mathbb{R}$ è continuo in quanto

$$\begin{aligned} |I_n(f)| &= \left|\sum_{j=1}^{n} w_j f_j\right|, \le \sum_{j=1}^{n} |w_j||f_j| \\ &\le \left(\sum_{j=1}^{n} |w_j|\right) \cdot \max_j |f_j| \le \left(\sum_{j=1}^{n} |w_j|\right) \cdot \|f\|_\infty \end{aligned} \tag{40}$$

e che la norma dell'operatore di quadratura

$$\|I_n\|_\infty = \max_{f \in C([a,b]), f \neq 0} \frac{|I_n(f)|}{\|f\|_\infty}$$

coincide con $\sum_{j=1}^{n} |w_j|$.

Mostriamo ora il seguente teorema attribuito a Stieltjes, che lega l'errore delle formule di quadratura a quello di miglior approssimazione

Teorema 7.1 *Sia (a,b) un intervallo limitato e si supponga $w : (a,b) \to \mathbb{R}$ sia una funzione peso e*

$$I_n(f) = \sum_{j=1}^{n} w_j f_j, \ \text{con } f_j = f(x_j)$$

una formula di quadratura avente grado di precisione n. Allora

$$|I(f) - I_n(f)| \le (\|w\|_1 + \|I_n\|_\infty) \cdot \min_{q_n \in \mathscr{P}_n} \|f - q_n\|_\infty.$$

Dimostrazione. Se $q_n \in \mathscr{P}_n$ è un polinomio arbitrario di grado n ed $I(f) = I_n(f)$ avendo la formula di quadratura grado di precisione n, essendo

$$|I(f)| = \left|\int_a^b f(x) w(x) dx\right| \le \|f\|_\infty \|w\|_1$$

poichè per definizione

$$|I_n(f)| \leq \|I_n\|_\infty \|f\|_\infty$$

$$\begin{aligned}
|I(f) - I_n(f)| &= |I(f) - I_n(q) + I_n(q) - I_n(f)| \\
&\leq |I(f) - I_n(q)| + |I_n(q) - I_n(f)| \\
&\leq |I(f) - I(q)| + |I_n(q - f)| \\
&\leq |I(f - q)| + |I_n(f - q)| \\
&\leq \|w\|_1 \|f - q\|_\infty + \|I_n\|_\infty \|f - q\|_\infty \\
&= (\|w\|_1 + \|I_n\|_\infty) \cdot \|f - q\|_\infty \quad \blacksquare \qquad (41)
\end{aligned}$$

L'interesse di questo teorema è il legame col polinomio di miglior approssimazione. Se i pesi sono positivi, allora $\|I_n\|_\infty = \|w\|_1$ e si ha

$$|I(f) - I_n(f)| \leq 2\|w\|_1 \cdot \min_{q_n \in \mathscr{P}_n} \|f - q_n\|_\infty.$$

Per stime delle quantità

$$\min_{q_n \in \mathscr{P}_n} \|f - q_n\|_\infty$$

si devono usare i teoremi di Jackson per l'errore del polinomio di miglior approssimazione di una funzione $f \in C([a, b])$ (dotando $C([a, b])$ della norma infinito).

Così ad esempio, se usiamo la funzione peso di Legendre $w \equiv 1$ nell'intervallo $(-1, 1)$ si ha che

$$|I(f) - I_n(f)| \leq 4 \cdot \min_{q_n \in \mathscr{P}_n} \|f - q_n\|_\infty$$

Nota. Supponiamo che alcuni pesi siano negativi. Di conseguenza

$$\int_a^b w(x)dx = \sum_k w_k^+ + \sum_k w_k^- \qquad (42)$$

$$\|I_n\|_\infty = \sum_k |w_k|^+ + \sum_k |w_k|^- = \sum_k w_k^+ - \sum_k w_k^- \qquad (43)$$

da cui

$$\|I_n\|_\infty = \sum_k w_k^+ - \sum_k w_k^- = \left(\int_a^b w(x)dx - \sum_k w_k^+ \right) - \sum_k w_k^- = \int_a^b w(x)dx - 2\sum_k w_k^-,$$

mentre se i pesi fossero tutti positivi avremmo

$$\|I_n\|_\infty = \sum_k w_k^+ - \sum_k w_k^- = \sum_k w_k^+ = \int_a^b w(x)dx$$

per cui $-2\sum_k w_k^- > 0$ peggiora la stabilità della formula di quadratura.

Figura 8: G. Polya (1887-1985) e V.A. Steklov (1864-1926).

8 Teoremi di convergenza

Consideriamo una formula di quadratura del tipo

$$\int_a^b f(x)w(x)dx \approx L_k(f) := \sum_{i=0}^{n} w_{i,n} f(x_{i,n}) \tag{44}$$

con al solito w una funzione peso definita nell'intervallo (a,b). Se tale intervallo è limitato ed f continua in $[a,b]$, allora $fw \in L^1(a,b)$. Sia

$$E_n(f) := \int_a^b f(x)w(x)dx - \sum_{i=0}^{n} w_{i,n} f(x_{i,n}).$$

Dimostriamo ora il teorema di Polya-Steklov [3, p.202]

Teorema 8.1 *Siano $x_{i,k\,i,k}$ dei punti di un intervallo fissato e chiuso $[a,b]$. Condizione necessaria e sufficiente affinchè per ogni funzione continua $f \in C(a,b)$ si abbia*

$$\lim_{n\to+\infty} E_n(f) = 0$$

è che sia

1. *esiste $M \in \mathbb{R}$ tale che per ogni n si abbia*

$$\sum_{i=0}^{n} |w_{i,n}| \le M;$$

2. *per ogni $k \in \mathbb{N}$ si abbia*

$$\lim_{k\to+\infty} E_n(x^k) = 0.$$

Dimostrazione. Supponiamo che

1. esiste $M \in \mathbb{R}$ tale che per ogni n si abbia

$$\sum_{i=0}^{n} |w_{i,n}| \le M;$$

2. per ogni $k \in \mathbb{N}$ si abbia
$$\lim_{k \to +\infty} E_n(x^k) = 0.$$

Per un teorema di densità dovuto a Weierstrass, per ogni $\epsilon > 0$ esiste un polinomio p tale che $\|f - p_n\|_\infty$ e quindi

$$\begin{aligned}
|E_n(f - p_n)| &\leq \left| \int_a^b (f(x) - p_n(x))\, w(x)\, dx \right| + \left| \sum_{i=0}^n w_{i,n} (f(x_{i,n}) - p_n(x_{i,n})) \right| \\
&\leq \|f - p_n\|_\infty \|w\|_1 + \sum_{i=0}^n |w_{i,n}| |f(x_{i,n}) - p_n(x_{i,n})| \\
&\leq \|f - p_n\|_\infty \cdot \|w\|_1 + \left(\sum_{i=0}^n |w_{i,n}| \right) \cdot \|f - p_n\|_\infty \\
&= \left(\|w\|_1 + \sum_{i=0}^n |w_{i,n}| \right) \cdot \epsilon. \qquad (45)
\end{aligned}$$

Di conseguenza

$$\begin{aligned}
|E_n(f)| &\leq |E_n(f) - E_n(p)| + |E_n(p)| \\
&\leq (\|w\|_1 + \sum_{i=0}^n |w_{i,n}|) \cdot \epsilon + |E_n(p)| \\
&\leq (\|w\|_1 + M) \cdot \epsilon + |E_n(p)|
\end{aligned} \qquad (46)$$

e siccome $|E_n(p)| \to 0$ al crescere di n poichè se $p(x) = \sum_{k=0}^m a_k x^k$ allora

$$E_n(p) = \sum_{k=0}^m a_k E_n(x^k) \to 0$$

abbiamo

$$\lim_n |E_n(f)| \leq (\|w\|_1 + M) \cdot \epsilon + 0$$

da cui per l'arbitrarietà di ϵ abbiamo $\lim_n |E_n(f)| = 0$.

Viceversa supponiamo $\lim_{n \to +\infty} E_n(f) = 0$ per ogni $f \in C([a, b])$. Si vede che

$$\begin{aligned}
|E_n(f)| &= |\int_a^b f(x) w(x) dx - \sum_{i=0}^n w_{i,n} f(x_{i,n})| \\
&\leq \int_a^b |f(x)| w(x) dx + |\sum_{i=0}^n w_{i,n} f(x_{i,n})| \\
&\leq \|w\|_1 \|f\|_\infty + \sum_{i=0}^n |w_{i,n}| |f(x_{i,n})| \\
&\leq \|w\|_1 \|f\|_\infty + (\sum_{i=0}^n |w_{i,n}|) \|f\|_\infty \\
&= \left(\|w\|_1 + \sum_{i=0}^n |w_{i,n}| \right) \cdot \|f\|_\infty \qquad (47)
\end{aligned}$$

In realtà si prova che

$$\|E_n\|_\infty = \max_{f\in C([a,b]), f\neq 0} \frac{|E_n(f)|}{\|f\|} = \|w\|_1 + \sum_{i=0}^{n} |w_{i,n}|.$$

Il teorema di uniforme limitatezza (talvolta citato come di Banach-Steinhaus) [2, p.58] stabilisce che se L_n è una sequenza di operatori lineari limitati da uno spazio di Banach V a uno spazio di Banach W e per ogni $v \in V$ la sequenza $\{L_n(v)\}_n$ è limitata allora

$$\sup_n \|L_n\| < +\infty.$$

Nel nostro caso

- $V \equiv (C([a,b]), \|\cdot\|_\infty)$, $W \equiv \mathbb{R}$ sono spazi di Banach,
- posto $L_n \equiv E_n$, se $f \in C([a,b])$ abbiamo che $E_n(f)$ è limitata in quanto per ipotesi è infinitesima,

e di conseguenza

$$\sup_n \sum_{i=0}^{n} |w_{i,n}| = \sup_n \left(\|w\|_1 + \sum_{i=0}^{n} |w_{i,n}| \right) = \sup_n \|E_n\|_\infty < +\infty$$

per cui esiste M finito ed indipendente da n tale che $\sum_{i=0}^{n} |w_{i,n}| \leq M$.

Il secondo punto da dimostrare è ovvio in quanto per ogni k, si ha $x^k \in C([a,b])$ $\blacksquare$.

Notiamo che

1. L'intervallo $[a,b]$ è limitato per cui il teorema di Polya non è applicabile per funzioni peso quali Gauss-Laguerre e Gauss-Hermite.
2. Si consideri una suddivisione $\Delta_m = \{\tau_i\}_{i=0,\ldots,m}$ dell'intervallo (a,b) con $\tau_i < \tau_{i+1}$, $\tau_0 = a$, $\tau_m = b$; si supponga che $h_i = \tau_{i+1} - \tau_i$ e che sia

   $$h_{\Delta_m} = \max_{k=0,\ldots,m-1} |h_k|.$$

   Se l'l-simo intervallo (con $l = 1,\ldots,m$) della suddivisione contiene i punti

   $$\tau_{l-1} = x_{(l-1)\cdot n} < x_{(l-1)\cdot n+1} < \ldots < x_{l\cdot n} = \tau_l$$

   sia $s_{\Delta_m,n}(f)$ la funzione polinomiale a tratti di grado n ottenuta interpolando nell'i-simo subintervallo con un polinomio di grado n le coppie

   $$(x_{(l-1)\cdot n}, f(x_{(l-1)\cdot n})), \ldots, (x_{l\cdot n}, f(x_{l\cdot n})),\ i = 1,\ldots,m.$$

   Come è noto, se $f \in C^{(n+1)}(a,b)$, per qualche $\xi \in (a,b)$

   $$\|f - s_{\Delta_m,n}\|_\infty \leq \frac{h^{(n+1)} f^{(n+1)}(\xi)}{(n+1)!} \leq \frac{h^{(n+1)} \|f^{(n+1)}\|_\infty}{(n+1)!}.$$

Se consideriamo una formula composta basata sull'applicare una formula di Newton-Cotes su $n+1$ pesi positivi in ognuno degli m subintervalli, si vede facilmente che tutti i suoi pesi sono positivi (perchè ?) e siccome in ogni sottointervallo la formula interpolatoria ha grado di precisione almeno 0, abbiamo

$$\sum_{i=0}^{n} |w_{i,n}| = \sum_{i=0}^{n} w_{i,n} = \sum_{i=0}^{n} w_{i,n} \cdot 1 = \int_a^b 1 dx = b-a.$$

Quindi, relativamente alla prima ipotesi del teorema di Polya-Steklov, essa è verificata per $M = b-a$. Per quanto riguarda la verifica della seconda ipotesi, tale formula composta integra esattamente ogni funzione polinomiale a tratti di grado n su Δ e quindi dalla formula dell'errore dell'interpolante polinomiale a tratti si ha

$$\begin{aligned}\int_a^b f(x)dx &= \int_a^b (f(x)-s_{\Delta_m,n})dx + \int_a^b s_{\Delta_m,n}dx \\ &= \int_a^b (f(x)-s_{\Delta_m,n})dx + \sum_{i=0}^{n} w_{i,n} f(x_{i,n})dx \qquad (48)\end{aligned}$$

da cui

$$\begin{aligned}|E_n(f)| &= \left|\int_a^b f(x)dx - \sum_{i=0}^{n} w_{i,n} f(x_{i,n})\right| \\ &= \left|\int_a^b (f(x)-s_{\Delta_m,n})dx\right| \\ &\le \left|\int_a^b 1\cdot dx\right| \cdot \|f-s_{\Delta_m,n}\|_\infty \\ &\le (b-a)\cdot \|f-s_{\Delta_m,n}\|_\infty \\ &\le (b-a)\cdot \frac{h^{(n+1)}\|f^{(n+1)}\|_\infty}{(n+1)!}\end{aligned} \qquad (49)$$

Siccome un polinomio $f(x) = x^k$ è infinitamente derivabile, $f^{(n+1)}$ è continua in $[a,b]$ e quindi per il teorema di Weierstrass $\|f^{(n+1)}\|_\infty$ è finito. Di conseguenza, se la successione di formule composte è tale che la massima suddivisione h tende a 0 allora $E_n(x^k) \to 0$. Per il teorema di Polya-Steklov si ha così che la formula composta

$$\int_a^b f(x)w(x)dx \approx \sum_i w_{i,n} f(x_{i,n}) \qquad (50)$$

è tale che qualsiasi sia la funzione continua f, $E_m(f) \to 0$ quando la massima ampiezza dei subintervalli tende a 0.

3. Se consideriamo una formula di Gauss, su un dominio limitato, i nodi $w_{i,n}$ sono positivi ed avendo grado di precisione almeno 1

$$\sum_{i=0}^{n} |w_{i,n}| = \sum_{i=0}^{n} w_{i,n} = \sum_{i=0}^{n} w_{i,n} \cdot 1 = \int_a^b 1 \cdot w(x)dx = \|w\|_1.$$

Quindi posto $M = \|w\|_1$, la prima ipotesi del teorema di Polya è verificata. Per quanto riguarda la seconda ipotesi, essendo il grado di precisione $2n+1$, fissato k, per $n \geq \text{ceil}((k-1)/2)$ si ha $E_n(x^k) = 0$. Di conseguenza, essendo tutti gli zeri contenuti in (a,b), possiamo applicare il teorema di Polya-Steklov e dedurre che al crescere del numero di punti $n+1$ della formula gaussiana si ha che

$$\lim_{n\to+\infty} E_n(f) = 0$$

qualsiasi sia la funzione continua $f \in C(a,b)$.

9 Equazioni di Fredholm di seconda specie

Si consideri l'equazione integrale di Fredholm di seconda specie

$$\lambda u(x) = \int_a^b k(x,y)u(y)dy + f(x), \; x \in [a,b] \tag{51}$$

dove $k \in C([a,b]\times[a,b])$ e $f \in C([a,b])$, $\lambda \in \mathbb{R}$.

Esistono vari teoremi di esistenza ed unicità della soluzione u^*. Un primo esempio è che sia (cf.[2, p.139])

$$\max_{a\leq x\leq b} \int |k(x,y)|dy < |\lambda|.$$

Per un'analisi più accurata si consulti [16].

Introduciamo ora il metodo di Nyström (cf.[2, p.373]). Sia una sequenza di formule di quadratura

$$\int_a^b g(y)dy \approx \sum_{j=1}^{n} w_{j,n} g(x_{j,n}), \; g \in C([a,b]) \tag{52}$$

e supponiamo che tale sequenza converga all'integrale esatto qualsiasi sia $g \in C([a,b])$. Possiamo così considerare quale appossimazione del problema originario, la determinazione della funzione u_n soluzione del problema

$$\lambda u_n(x) = \sum_{j=1}^{n} w_{j,n} k(x, x_{j,n}) u_n(x_{j,n}) + f(x), \; x \in [a,b]. \tag{53}$$

dove $\{u_n(x_{j,n})\}_{j=1,\ldots,q_n}$ verifica

$$\lambda u(x_{i,n}) = \sum_{j=1}^{n} w_{j,n} k(x_{i,n}, x_{j,n}) u_n(x_{j,n}) + f(x), \; x \in [a,b]. \tag{54}$$

Posti $A^{(n)}_{i,j} = w_{j,n} k(x_{i,n}, x_{j,n})$, $\mathbf{u}_n = \{u_n(x_{j,n})\}_j$, $\mathbf{f}_n = \{f(x_{j,n})\}_j$ si ha così che

$$\mathbf{u}_n = A^{(n)} \mathbf{u}_n + \mathbf{f}_n$$

per cui se $B^{(n)} = I - A^{(n)}$ è non singolare, allora

$$B^{(n)}\mathbf{u}_n = \mathbf{f}_n,$$

sistema lineare che può essere risolto con il comando \ di Matlab.

Una volta ottenuto $u_n(x_{j,n})$, tramite (53), la funzione soluzione u_n può essere valutata in qualsiasi punto $x \in [a,b]$. Per le proprietà di convergenza di questo metodo si veda [2, p.376]

9.1 Esercizio

Si consideri il problema

$$2 \cdot u(x) = \int_0^1 \exp(xy)u(y)dy + (2\exp(x) - \frac{1}{x+1} \cdot (\exp(x+1)-1)),\ x \in [a,b]. \quad (55)$$

avente unica soluzione $\exp(x)$ (perchè ?). Usando una formula di Gauss-Legendre, scalata nell'intervallo $[0,1]$, si valutino per $n = 2,4,8,16,32$ le soluzioni approssimate u_n nei nodi test $x_k = kh$ dove $h = 1/100$, $k = 0,\ldots,100$ e si calcoli

$$E_n = \max_{k=0,\ldots,100} |u_n(x_k) - \exp(x_k)|.$$

Come diminuisce l'errore (farne un plot in scala semilogaritmica)?
Suggerimento: in una versione di base, si può descrivere la valutazione dell'operatore integrale tramite due cicli for innestati

```
for i_index=1:length(x_pts)
    for j_index=1:length(x_gl_ab)
        kern_eval=feval(kern,x_pts(i_index),x_gl_ab(j_index));
        B_gl_grid(i_index,j_index)=kern_eval*w_gl_ab(j_index);
    end
end
```

10 Online

Alcuni link interessanti sono i seguenti:

http://www.cs.purdue.edu/people/wxg
http://en.wikipedia.org/wiki/Clenshaw-Curtis_quadrature
http://en.wikipedia.org/wiki/Fast_Fourier_transform
http://en.wikipedia.org/wiki/Gaussian_quadrature
http://it.wikipedia.org/wiki/Quadratura_di_Gauss
http://it.wikipedia.org/wiki/Regola_del_trapezio
http://it.wikipedia.org/wiki/Regola_di_Cavalieri-Simpson

Per le biografie di alcuni matematici menzionati in questa lezioni si considerino

http://it.wikipedia.org/wiki/Bonaventura_Cavalieri
http://it.wikipedia.org/wiki/Roger_Cotes
http://www-gap.dcs.st-and.ac.uk/~history/Biographies/Cavalieri.html
http://www-history.mcs.st-andrews.ac.uk/Biographies/Cotes.html
http://www-history.mcs.st-andrews.ac.uk/Biographies/Hermite.html
http://www-history.mcs.st-andrews.ac.uk/Biographies/Jacobi.html
http://www-history.mcs.st-andrews.ac.uk/Biographies/Laguerre.html
http://www-history.mcs.st-andrews.ac.uk/Biographies/Simpson.html

Quale nota storica osserviamo che da [11]

> Simpson is best remembered for his work on interpolation and numerical methods of integration. However the numerical method known today as "Simpson's rule", although it did appear in his work, was in fact due to Newton as Simpson himself acknowledged. By way of compensation, however, the Newton-Raphson method for solving the equation f(x) = 0 is, in its present form, due to Simpson. Newton described an algebraic process for solving polynomial equations which Raphson later improved. The method of approximating the roots did not use the differential calculus. The modern iterative form $x_{n+1} = x_n - f(x_n)/f'(x_n)$ is due to Simpson, who published it in 1740.

11 Frasi celebri

1. Cayley, Hermite and I constitute the Invariantive Trinity. (Sylvester)
2. The traditional mathematics professor of the popular legend is absentminded. He usually appears in public with a lost umbrella in each hand. He prefers to face a blackboard and to turn his back on the class. He writes a, he says b, he means c, but it should be d. Some of his sayings are handed down from generation to generation:
   (a) *In order to solve this differential equation you look at it till a solution occurs to you.*
   (b) *This principle is so perfectly general that no particular application of it is possible.*
   (c) *Geometry is the science of correct reasoning on incorrect figures.*
   (d) *My method to overcome a difficulty is to go round it.*
   (e) *What is the difference between method and device? A method is a device which you used twice.*
3. Even fairly good students, when they have obtained the solution of the problem and written down neatly the argument, shut their books and look for something else. Doing so, they miss an important and instructive phase of the work. ... A good teacher should understand and impress on his students the view that no problem whatever is completely exhausted. (Polya)

4. One of the first and foremost duties of the teacher is not to give his students the impression that mathematical problems have little connection with each other, and no connection at all with anything else. We have a natural opportunity to investigate the connections of a problem when looking back at its solution. (Polya)

5. If there is a problem you can't solve, then there is an easier problem you can solve: find it. (Polya)

6. A great discovery solves a great problem, but there is a grain of discovery in the solution of any problem. Your problem may be modest, but if it challenges your curiosity and brings into play your inventive faculties, and if you solve it by your own means, you may experience the tension and enjoy the triumph of discovery. (Polya)

7. The first rule of discovery is to have brains and good luck. The second rule of discovery is to sit tight and wait till you get a bright idea. (Polya)

8. If you have to prove a theorem, do not rush. First of all, understand fully what the theorem says, try to see clearly what it means. Then check the theorem; it could be false. Examine the consequences, verify as many particular instances as are needed to convince yourself of the truth. When you have satisfied yourself that the theorem is true, you can start proving it. (Polya)

9. Mathematics consists of proving the most obvious thing in the least obvious way. (Polya)

10. Mathematics is the cheapest science. Unlike physics or chemistry, it does not require any expensive equipment. All one needs for mathematics is a pencil and paper. (Polya)

11. There are many questions which fools can ask that wise men cannot answer. (Polya)

12. A mathematician who can only generalise is like a monkey who can only climb up a tree, and a mathematician who can only specialise is like a monkey who can only climb down a tree. In fact neither the up monkey nor the down monkey is a viable creature. A real monkey must find food and escape his enemies and so must be able to incessantly climb up and down. A real mathematician must be able to generalise and specialise. (Polya)

13. Look around when you have got your first mushroom or made your first discovery: they grow in clusters. (Polya)

14. John von Neumann was the only student I was ever afraid of. (Polya)

15. The apex and culmination of modern mathematics is a theorem so perfectly general that no particular application of it is feasible. (Polya)

16. I am too good for philosophy and not good enough for physics. Mathematics is in between. (Polya)

17. The elegance of a mathematical theorem is directly proportional to the number of independent ideas one can see in the theorem and inversely proportional to the effort it takes to see them. (Polya)

References

[1] K. Atkinson, *Introduction to Numerical Analysis*, Wiley, 1989.

[2] K. Atkinson e W. Han, *Theoretical Numerical Analysis*, Springer, 2001.

[3] V. Comincioli, *Analisi Numerica, metodi modelli applicazioni*, Mc Graw-Hill, 1990.

[4] S.D. Conte e C. de Boor, *Elementary Numerical Analysis, 3rd Edition*, Mc Graw-Hill, 1980.

[5] P.J. Davis, *Interpolation and Approximation*, Dover, 1975.

[6] W. Gautschi, personal homepage, http://www.cs.purdue.edu/people/wxg.

[7] Mac Tutor, Cavalieri, http://www-gap.dcs.st-and.ac.uk/ history/Biographies/Cavalieri.html.

[8] Mac Tutor, Cotes, http://www-history.mcs.st-andrews.ac.uk/Biographies/Cotes.html.

[9] Mac Tutor, Hermite, http://www-history.mcs.st-andrews.ac.uk/Biographies/Hermite.html.

[10] Mac Tutor, Jacobi, http://www-history.mcs.st-andrews.ac.uk/Biographies/Jacobi.html.

[11] Mac Tutor, Simpson, http://www-history.mcs.st-andrews.ac.uk/Biographies/Simpson.html.

[12] The MathWorks Inc., *Numerical Computing with Matlab*, http://www.mathworks.com/moler.

[13] A. Quarteroni e F. Saleri, *Introduzione al calcolo scientifico*, Springer Verlag, 2006.

[14] A. Suli e D. Mayers, *An Introduction to Numerical Analysis*, Cambridge University Press, 2003.

[15] L.N. Trefethen, *Is Gauss quadrature better than Clenshaw-Curtis?*, SIAM Reviews, to appear (2007), http://web.comlab.ox.ac.uk/oucl/work/nick.trefethen/CC_trefethen_revised2.pdf.

[16] F.G. Tricomi, *Integral Euations*, Dover, 1985.

[17] J. Waldvogel, *"Fast Construction of the Fejer and Clenshaw-Curtis Quadrature Rules"*, BIT 46 (2006), no. 1, 195–202, http://www.math.ethz.ch/~waldvoge/Papers/fejer.pdf.

[18] Wikipedia, Cavalieri, http://it.wikipedia.org/wiki/Bonaventura_Cavalieri.

[19] Wikipedia, Clenshaw-Curtis quadrature, http://en.wikipedia.org/wiki/Clenshaw-Curtis_quadrature.

[20] Wikipedia, Fast Fourier Transform, http://en.wikipedia.org/wiki/Fast_Fourier_transform.

[21] Wikipedia, Gaussian quadrature, http://en.wikipedia.org/wiki/Gaussian_quadrature.

[22] Wikipedia, Quadratura gaussiana, http://it.wikipedia.org/wiki/Quadratura_di_Gauss.

[23] Wikipedia, Regola del trapezio, http://it.wikipedia.org/wiki/Regola_del_trapezio.

[24] Wikipedia, Regola di Cavalieri-Simpson, http://it.wikipedia.org/wiki/Regola_di_Cavalieri-Simpson.