

# Quadratura numerica

Alvise Sommariva

Università degli Studi di Padova  
Dipartimento di Matematica Pura e Applicata

30 aprile 2012

Il problema della quadratura numerica consiste nell'approssimare l'*integrale definito* di una funzione  $f$  in un intervallo avente estremi di integrazione  $a, b$  (non necessariamente finiti) cioè

$$I^{(w)}(f) := I^{(w)}(f, a, b) = \int_a^b f(x) w(x) dx$$

con

$$I_N^{(w)}(f) := \sum_{k=1}^N w_i f(x_i)$$

dove  $w$  è una funzione peso in un intervallo avente estremi  $a, b$ . A seconda degli esempi, l'intervallo  $\mathcal{I}$  può essere aperto o chiuso.

I termini  $w_i$  e  $x_i \in \mathcal{I}$  sono detti rispettivamente **pesi** e **nodi**.

# Funzioni peso

Sia  $w : (a, b) \rightarrow \mathbb{R}$  una funzione non negativa, con  $(a, b)$  non necessariamente limitato, tale che

1.  $\int_a^b |x|^n w(x) dx < +\infty$  per tutti gli  $n \in \mathbb{N}$ ;
2.  $\int_a^b g(x) w(x) dx = 0$  per qualche funzione continua e non negativa  $g$  implica  $g \equiv 0$  in  $(a, b)$ .

La funzione  $w$  è detta *peso*. Le funzioni peso più comuni sono

1.  $w(x) = 1$  con  $x \in [-1, 1]$  (peso di Legendre);
2.  $w(x) = \frac{1}{\sqrt{1-x^2}}$  con  $x \in (-1, 1)$  (peso di Chebyshev);
3.  $w(x) = (1-x^2)^{\gamma-(1/2)}$  con  $x \in (-1, 1)$ ,  $\gamma > (-1/2)$  (peso di Gegenbauer);
4.  $w(x) = (1-x)^\alpha \cdot (1+x)^\beta$  con  $x \in (-1, 1)$ ,  $\alpha > -1$ ,  $\beta > -1$  (peso di Jacobi);
5.  $w(x) = \exp(-x)$  con  $x \in (0, +\infty)$  (peso di Laguerre);
6.  $w(x) = \exp(-x^2)$  con  $x \in (-\infty, +\infty)$  (peso di Hermite);

Si supponga  $[a, b]$  un intervallo chiuso e limitato di  $\mathbb{R}$  e sia  $w \equiv 1$ . 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 che si ottengono integrando l'interpolante di  $f$  in nodi equispaziati

$$x_k = a + \frac{(k-1)(b-a)}{N-1}, \quad k = 1, \dots, N.$$

## Esempi:

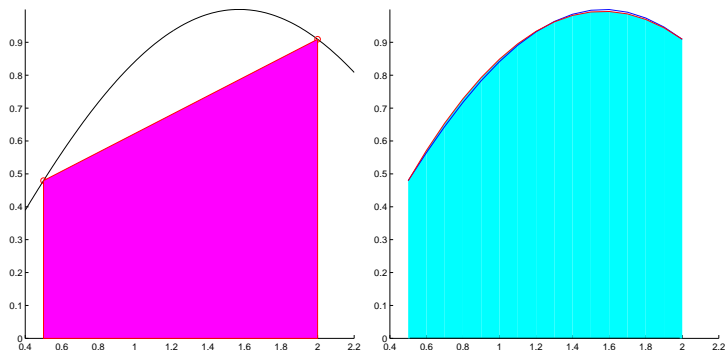
- ▶  $N=2$ , **regola del trapezio**, **g.d.p.=1**:

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

- ▶  $N=3$ , **regola di Cavalieri-Simpson**, **g.d.p.=3**:

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

# Formule di Newton-Cotes



**Figura:** 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).

# Formule di Newton-Cotes composte

Si suddivide 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

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

dove  $S$  è una delle regole di quadratura finora esposte (ad esempio  $S_3(f)$ ). Le formule descritte in (1) sono dette *composte*.

**Formula composta dei trapezi:** fissati il numero  $N$  di subintervalli e i punti  $x_k = a + kh$  dove  $h = \frac{b-a}{N}$  è definita da

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

il cui errore è

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

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

**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

$$I(f) \approx S_3^{(c)}(f) := \frac{h}{6} \left[ f(x_0) + 2 \sum_{r=1}^{N-1} f(x_{2r}) + 4 \sum_{s=0}^{N-1} f(x_{2s+1}) + f(x_{2N}) \right] \quad (3)$$

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



# Formule di Newton-Cotes

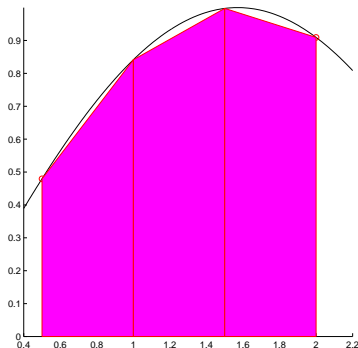


Figura: Formula dei trapezi composta  $\int_{0.5}^2 \sin(x) dx$ .

# Formula trapezi composta

La funzione `trapezi_composta` appena esposta calcola i nodi e i pesi della omonima formula composta.

```
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;
```

# Formula Cavalieri-Simpson composta

La funzione `simpson_composta` appena esposta calcola i nodi e i pesi della omonima formula 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;
```

# Esempio 1

Vogliamo approssimare l'integrale

$\int_{-1}^1 x^{20} dx = 2/21 \approx 0.09523809523810$  tramite le formule composte dei trapezi e Cavalieri-Simpson. Perchè il paragone sia veritiero, decidiamo che il numero di valutazioni della funzione integranda sia uguale. Scriviamo nel file `demo_composte.m`

```
N=11; %SCEGLIERE DISPARI.
a=-1; b=1; f=inline('x.^20');
% a=0; b=1; f=inline('exp(x)');
N_trap=N-1; % TRAPEZI COMPOSTA.
[x_trap,w_trap]=trapezi_composta(N_trap,a,b);
fx_trap=feval(f,x_trap); I_trap=w_trap'*fx_trap;
N_simpson=(N-1)/2; % CAV.SIMPSON COMPOSTA.
[x_simp,w_simp]=simpson_composta(N_simpson,a,b);
fx_simp=feval(f,x_simp); I_simp=w_simp'*fx_simp;
fprintf('\n\t [TPZ.COMP.][PTS]: %4.0f',length(x_trap));
fprintf('\n\t [TPZ.COMP.][RIS]: %14.14f',I_trap);
fprintf('\n\t [CS.COMP.][PTS]: %4.0f',length(x_simp));
fprintf('\n\t [CS.COMP.][RIS]: %14.14f \n\n',I_simp);
```

# Esempio 1

Ricordando che il risultato è 0.09523809523810, otteniamo i poco soddisfacenti

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

Posto  $N = 51$  nella prima riga di `demo_composte.m`

```
[TPZ.COMP.][PTS]:    51  
[TPZ.COMP.][RIS]: 0.10052328836742  
[CS.COMP.][PTS]:    51  
[CS.COMP.][RIS]: 0.09542292188917
```

# Esempio 1, I

Vediamo gli errori ottenuti

```
>> s=0.09523809523810; % SOL.
>> t11=0.20462631505024; %TRAP. 11.
>> abserr=abs(s-t11);relerr=abserr/abs(s);
>> [abserr relerr]
ans = 0.1094    1.1486
>> s11=0.13949200364447; %CS. 11.
>> abserr=abs(s-s11);relerr=abserr/abs(s);
>> [abserr relerr]
ans = 0.0443    0.4647
>> t51=0.10052328836742; % TPZ: 51.
>> abserr=abs(s-t51);relerr=abserr/abs(s);
>> [abserr relerr]
ans = 0.0053    0.0555
>> s51=0.09542292188917; % CS.51
>> abserr=abs(s-s51);relerr=abserr/abs(s);
>> [abserr relerr]
ans = 0.0002    0.0019
```

## Esempio 1, II

```
| [ TPZ . COMP . ] [ PTS ] :    11  
| [ TPZ . COMP . ] [ RIS ] : 1.71971349138931  
| [ CS . COMP . ] [ PTS ] :    11  
| [ CS . COMP . ] [ RIS ] : 1.71828278192482
```

## Esempio 2

Approssimiamo  $\int_0^1 \exp(x) dx = \exp(1) - 1 \approx 1.718281828459046$  tramite le formule composte dei trapezi e Cavalieri-Simpson. Perchè il paragone sia veritiero, decidiamo che il numero di valutazioni della funzione integranda sia uguale. Modifichiamo la funzione  $f$  nel file `demo_composte.m` e ricaviamo

```
[ TPZ . COMP . ] [ PTS ] :    11  
[ TPZ . COMP . ] [ RIS ] :  1.71971349138931  
[ CS . COMP . ] [ PTS ] :    11  
[ CS . COMP . ] [ RIS ] :  1.71828278192482
```

Gli errori relativi sono rispettivamente circa  $8.33 \cdot 10^{-4}$  e  $5.55 \cdot 10^{-7}$ , quelli assoluti circa  $1.43 \cdot 10^{-3}$  e  $9.55 \cdot 10^{-7}$ .



## Esempio 2: errore

Nel caso della formula dei trapezi composta, abbiamo

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

per qualche  $\xi \in (a, b)$  e quindi

$$|E_1^{(c)}(f)| \leq \left| \frac{-(b-a)}{12} h^2 \max_{x \in (0,1)} \exp(x) \right| \leq \frac{1}{12} h^2 \exp(1)$$

Nel nostro esempio  $h = 0.1$  e quindi

$$|E_1^{(c)}(f)| \leq \frac{1}{12} h^2 \exp(1) = \frac{1}{12} 0.1^2 \exp(1) \approx 2.3 \cdot 10^{-3},$$

mentre l'errore assoluto effettivo era circa  $1.43 \cdot 10^{-3}$ .

## Esempio 2: errore

Nel caso della formula di Cavalieri-Simpson composta, abbiamo

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

da cui

$$|E_3^{(c)}(f)| \leq \left| \frac{-(b-a)}{180} \left(\frac{h}{2}\right)^4 \max_{x \in (0,1)} \exp(x) \right| \leq \frac{1}{180} \left(\frac{h}{2}\right)^4 \exp(1)$$

Nel nostro esempio  $h = (b-a)/N = 0.2$  ( $N$  subintervalli) e quindi

$$|E_3^{(c)}(f)| \leq \frac{1}{180} \left(\frac{h}{2}\right)^4 \exp(1) = \frac{1}{180} \left(\frac{0.2}{2}\right)^4 \exp(1) \approx 1.5102 \cdot 10^{-6},$$

mentre l'errore assoluto effettivo era circa  $9.55 \cdot 10^{-7}$ .

Nelle formule interpolatorie di Newton-Cotes (come ad esempio la regola del Trapezio o di Cavalieri-Simpson) i nodi  $x_1, \dots, x_n$  sono equispaziati e il grado di precisione  $\delta$  è generalmente uguale almeno a  $n - 1$  ma in alcuni casi, come per la regola di Cavalieri-Simpson, uguale al numero di nodi  $n$ . Vediamo ora formule  $\sum_{k=1}^n w_k f(x_k)$

- ▶ hanno grado di precisione  $\delta$  maggiore di  $2n - 1$ ;
- ▶ per qualche funzione  $f : (a, b) \rightarrow \mathbb{R}$  continua approssimino  $I^{(w)}(f) := \int_a^b f(x)w(x) dx$ , con  $(a, b)$  non necessariamente limitato.

**Teorema.** Per ogni  $n \geq 1$  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 almeno  $2n - 1$ . I nodi sono gli **zeri del polinomio ortogonale di grado  $n$** ,

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

e i corrispettivi pesi sono

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

**Importante:** Si dimostra che gli zeri sono **semplici** e **interni** all'intervallo  $(a, b)$ .

# Vantaggi integrali con peso

Diamo di seguito l'idea sul perchè si utilizzino funzioni peso  $w$  per il calcolo di integrali. Se dobbiamo approssimare  $\int_a^b g(x) dx$  con  $g$  regolare allora di solito non sarà difficile calcolare numericamente il valore dell'integrale, mentre se  $g$  è poco regolare sarà molto più oneroso ottenere buoni risultati.

Introducendo una funzione peso  $w$ , e supposto che  $f = g/w$  sia regolare, si considera il calcolo di

$$\int_a^b g(x) dx = \int_a^b \frac{g(x)}{w(x)} w(x) dx = \int_a^b f(x) w(x) dx.$$

La regolarità di  $f$  e la particolarità del calcolo dipendentemente dalla funzione peso permette di approssimare meglio

$$\int_a^b g(x) dx = \int_a^b f(x) w(x) dx.$$

**Teorema.** Sia  $(a, b)$  un intervallo limitato e si supponga  $w : (a, b) \rightarrow \mathbb{R}$  sia una funzione peso,  $I^{(w)} = \int_a^b f(x)w(x)dx$  e  $I_m^{(w)}(f) := \sum_{k=1}^m w_i f(x_i)$  una formula di quadratura avente g.d.p.  $n$ . Allora ricordato che  $\|I_n\|_\infty = \sup_{f \in C(a,b) \setminus 0} |I_n(f)| / \|f\|_\infty$ ,  $\|w\|_1 = \int_a^b w(x)dx$ , abbiamo

$$|I^{(w)} - I_m^{(w)}(f)| \leq \left( \|w\|_1 + \|I_m^{(w)}\|_\infty \right) \cdot \min_{q_n \in \mathbb{P}_n} \|f - q_n\|_\infty.$$

Quindi è importante avere formule con grado di precisione  $n$  alto e  $f$  regolare così da avere minore  $\min_{q_n \in \mathbb{P}_n} \|f - q_n\|_\infty$ .

Consideriamo il calcolo approssimato

$$\int_{-1}^1 \exp(x) \sqrt{1-x} dx = 1.7791436546919097925911790299941. \quad (4)$$

- ▶ Non è suggerito il calcolo dell'integrale con formule composte poichè la funzione  $\exp(x)\sqrt{1-x}$  è poco regolare.
- ▶ Non è indicato usare formule gaussiane con peso  $w(x) = 1$  poichè la funzione  $\exp(x)\sqrt{1-x}$  è poco regolare.
- ▶ Si suggerisce di usare formule gaussiane con peso di Jacobi  $w(x) = \sqrt{1-x} = (1-x)^{1/2}$  poichè la funzione  $\exp(x)$  è  $C^\infty(-1, 1)$ .

# Formule gaussiane in Matlab

Il calcolo di nodi e pesi delle formule gaussiane, per una fissata funzione peso  $w$  non è semplice. Di seguito ci interesseremo al caso particolare delle funzioni peso di Jacobi. A tal proposito aiutano le routines `gauss.m` e `r_jacobi.m` di W. Gautschi e D. Laurie che possono essere scaricate dalla homepage del primo autore. La funzione `gauss_jacobi.m`, se registrata nella stessa cartella di `gauss.m` e `r_jacobi.m`, calcola i nodi  $\mathbf{x} = \{x_i\}_{i=1,\dots,n}$  e i pesi  $\mathbf{w} = \{w_i\}_{i=1,\dots,n}$  della formula gaussiana (avente g.d.p.  $2n - 1$ )

$$\int_{-1}^1 f(x) \cdot (1-x)^\alpha (1+x)^\beta \approx \sum_{i=1}^n w_i f(x_i)$$

```
function [x,w]=gauss_jacobi(n,alpha,beta)
ab=r_jacobi(n,alpha,beta);
xw=gauss(n,ab);
x=xw(:,1);
w=xw(:,2);
```



Specialmente per le formule con peso  $w(x) \equiv 1$  si ha spesso da effettuare l'integrale  $\int_a^b f(x)dx$  con  $(a, b)$  diverso da  $(-1, 1)$ . In tal caso si osserva che se  $\gamma(t) = (a(1 - t)/2) + (b(1 + t)/2)$ , per  $\{t_k\}$  e  $\{w_k\}$  nodi e pesi della formula di Gauss-Legendre,

$$\int_a^b f(x)dx = \int_{-1}^1 \frac{b-a}{2} f(\gamma(t))dt \approx \sum_{k=1}^n \frac{b-a}{2} w_k f(\gamma(t_k))$$

e quindi

$$\int_a^b f(x)dx \approx \sum_{k=1}^n w_k^* f(x_k^*)$$

con  $w_k^* = \frac{b-a}{2} w_k$ ,  $x_k = \gamma(t_k) = (a(1 - t_k)/2) + (b(1 + t_k)/2)$ .  
Questo processo si chiama *scalatura* dei nodi e dei pesi.

Scriviamo quindi nel file `demo_gauss_jacobi`

```
N=11;
alpha=0; beta=0;
a=-1; b=1; f=inline('x.^20');
% a=0; b=1; f=inline('exp(x)');
[x,w]=gauss_jacobi(N,alpha,beta);
xx=a*(1-x)/2+b*(1+x)/2; ww=(b-a)*w/2;
fxx=fval(f,xx);
I=ww'*fxx;
fprintf('\n\t[N]:%3.0f [I]: %1.15e \n\n',N,I)
```

calcolando così con una formula gaussiana di 11 nodi l'integrale

$$\int_0^1 \exp(x) dx = \exp(1) - 1 \approx 1.718281828459046.$$

# Formule gaussiane in Matlab: demo ed esempio 1

Dalla shell di Matlab/Octave

```
>> demo_gauss_jacobi  
[N]: 11 [I]: 9.523809523809551e-02  
>> I  
I = 9.523809523809551e-02  
>> Iex=2/21; abs(I-Iex)  
ans = 2.775557561562891e-16
```

che va confrontato, a parità di nodi, con  $5.28 \cdot 10^{-3}$  e  $1.85 \cdot 10^{-4}$  rispettivamente delle formule dei trapezi e Cavalieri-Simpson composte. Si osservi che il grado di precisione della formula gaussiana con  $n = 11$  è  $2n - 1 = 21$  e quindi l'integrale  $\int_{-1}^1 x^{20} dx$  è calcolato, a meno di errori macchina, esattamente.

## Formule gaussiane in Matlab: demo ed esempio 2

Dalla shell di Matlab/Octave, rimaneggiando i commenti del codice,

```
>> demo_gauss_jacobi  
[N]: 11 [I]: 1.718281828459045 e+00  
>> Iex=exp(1)-exp(0); I=1.718281828459045 e+00;  
>> abs(Iex-I)  
ans = 4.440892098500626 e-16
```

che va confrontato, a parità di nodi, con  $1.43 \cdot 10^{-3}$  e  $9.55 \cdot 10^{-7}$  rispettivamente delle formule dei trapezi e Cavalieri-Simpson composte.

## Formule gaussiane in Matlab: demo ed esempio 3

Approssimiamo in un file Matlab/Octave `demo_comparison.m`

$\int_{-1}^1 \exp(x) \sqrt{1-x} dx = 1.77914365469190979259 \dots$  mediante la formula dei trapezi composta, Cav.-Simpson composta, Gauss-Legendre e Gauss-Jacobi con  $\alpha = 1/2$ ,  $\beta = 0$ .

```
N=11; a=-1;b=1; Iex=1.7791436546919097925911790299941;
g=inline('exp(x).*sqrt(1-x)'); f=inline('exp(x)');
[xt,wt]=trapezi_composta(N,a,b); % TPZ.
fxt=feval(g,xt); It=wt'*fxt; et=abs(Iex-It);
[xcs,wcs]=simpson_composta(N,a,b); % C.S.
fxcs=feval(g,xcs); Ics=wcs'*fxcs; ecs=abs(Iex-Ics);
[xgl,wgl]=gauss_jacobi(N,0,0); % G.L.
fxgl=feval(g,xgl); Igl=wgl'*fxgl; egl=abs(Iex-Igl);
[xj,wj]=gauss_jacobi(N,0.5,0); % G.J.
fxj=feval(f,xj); Ij=wj'*fxj; ej=abs(Iex-Ij);
fprintf('\n TPZ:%1.1e CS:%1.1e',et,ecs);
fprintf('GL:%1.1e GJ:%1.1e \n\n',egl,ej);
```

# Formule gaussiane in Matlab: demo ed esempio 3

Con ovvia notazione, a parità di valutazioni della funzione integranda, abbiamo

```
>> demo_comparison  
[TPZ]: 4.4e-02 [CS]: 6.2e-03 [GL]: 5.3e-04 [GJ]: 6.7e-16  
>>
```

che evidenzia la superiorità delle formule gaussiane, una volta che il peso è scelto opportunamente.

# Esercizio sulla formula di Fejér, I

- ▶ Calcolare con una routine Matlab `nodi_chebyshev.m` i nodi di Gauss-Chebyshev

$$x_k = \cos(\theta_k), \quad \theta_k = (2k - 1) \cdot \pi / (2n), \quad k = 1, 2, \dots, n$$

per un certo  $n$  prefissato.

- ▶ Calcolare con una routine Matlab `momenti_chebyshev.m` i valori

$$\gamma_m = 2/(1 - m^2), \quad \text{se } m \text{ pari}$$

$$\gamma_m = 0, \quad \text{se } m \text{ dispari}$$

per  $m = 0, \dots, n$ , per un certo  $n$  prefissato. Questi valori corrispondono con i momenti dei polinomi di Chebyshev

$$T_m(x) = \cos(m \cdot \arccos(x))$$

rispetto alla misura di Legendre, cioè ai valori

$$\gamma_m := \int_{-1}^1 T_m(x) dx.$$

## Esercizio sulla formula di Fejér, II

- Aiutandosi col comando

```
|>> help gallery
```

valutare la *matrice di Vandermonde per i polinomi di Chebyshev*  $V$  di grado  $n$  in  $n + 1$  nodi di Chebyshev.

Se  $w$  è il vettore che risolve il problema  $V \cdot w = \gamma$  (attenzione alle dimensioni!), allora si ottiene la formula di quadratura di Fejér

$$\int_{-1}^1 f(x) dx \approx \sum_{k=1}^{n+1} w_k f(x_k).$$



## Esercizio sulla formula di Fejér, III

Effettuare una routine Matlab `Formula_Fejer.m` che

1. calcoli  $n + 1$  nodi di Chebyshev  $\{x_k\}$ ;
2. valuti la matrice di Vandermonde per i polinomi di Chebyshev  $V$  di grado  $n$  in  $\{x_k\}$ ;
3. calcoli i momenti  $\gamma$  dei polinomi di Chebyshev;
4. risolva tramite il comando `backslash` di Matlab il sistema lineare  $V \cdot w = \gamma$ .

## Esercizio sulla formula di Fejér, IV

- ▶ Usare la formula di Fejér con  $n = 11$  per calcolare l'integrale  $\int_{-1}^1 x^{20} dx = 2/21 \approx 0.09523809523810$ ;
- ▶ Usare la formula di Fejér con  $n = 11$  per calcolare l'integrale  $\int_0^1 \exp(x) dx = \exp(1) - 1 \approx 1.718281828459046$ . Se opportuno, sostituire il calcolo di questo integrale con uno in  $(-1, 1)$ .