Tutoraggio #6

Federico Pichi

29 maggio 2019

Esercizio 1

Implementare in Matlab i metodi di integrazione composita di punto medio, trapezi e Cavalieri-Simpson come delle functions che prendano in input gli estremi di integrazione, il numero di sottointervalli e l'integranda, e che restituiscano il valore dell'integrale approssimato.

Esercizio 2

Si utilizzino le formule composite del punto medio, del trapezio e di Cavalieri-Simpson per calcolare l'integrale

$$\int_0^{2\pi} xe^{-x}\cos(2x)dx = \frac{\left[3(e^{-2\pi}-1)-10\pi e^{-2\pi}\right]}{25},$$

riportando in scala logaritmica l'andamento degli errori in funzione del passo $h=(b-a)/m$, dove m è il numero di sottointervalli che varia da 16 a 256 raddioppiando ad ogni step. Utilizzare il rapporto tra le iterate successive per determinare sperimentalmente l'ordine di convergenza dei tre metodi.

Esercizio 3

Si calcoli il numero minimo m di sottointervalli necessari per approssimare con punto medio, a meno di un errore di 10^{-4}, l'integrale delle seguenti funzioni negli intervalli indicati

- $f_1(x) = \frac{1}{1+(x-\pi)^2}$ in $[0,5]$,
- $f_2(x) = e^x cos(x)$ in $[0,\pi]$,
- $f_3(x) = \sqrt{x(1-x)}$ in $[0,1]$.

Si faccia lo stesso per la formula di Cavalieri-Simpson commentando i risultati.

Esercizio 4

Si vuole calcolare, con precisione dell'ordine di 10^4 km, la lunghezza dell'orbita terrestre attorno al sole. I semiassi maggiore e minore dell'orbita ellittica misurano rispettivamente $a = 149.60\cdot 10^6$ km e $b = a\sqrt{1-e^2}$ km, con $e = 0.0167086$ (eccentricità) e sapendo che la lunghezza di una ellisse è

$$L = \int_0^{2\pi} \sqrt{a^2\cos^2(t)+b^2\sin^2(t)}dt\ .$$

Utilizzando la formula composita dei trapezi, dire qual è il minimo numero di intervalli (di uguale ampiezza h) che permette di approssimare l'integrale dato con l'accuratezza richiesta. Fornire infine il valore calcolato dell'orbita terrestre.

Soluzione 1

Codice 1: Soluzione 1

```
function int = puntomedio(a,b,m,fun)
% puntomedio formula composita del punto medio.
% INT=PUNTOMEDIO(A,B,M,FUN) calcola un'approssimazione dell'integrale della
% funzione FUN su (A,B) con il metodo del punto medio su una griglia uniforme
% di M intervalli.  FUN riceve in ingresso un vettore reale x e restituisce un
% vettore della stessa dimensione.
h=(b-a)/m;
x=a+h/2:h:b;
y=fun(x);
if size(y)==1
  dim=length(x);
  y=y.*ones(dim,1);
end
int=h*sum(y);
end
```

Codice 2: Soluzione 1

```
function int = trapezi(a,b,m,fun)
% trapezi formula composita del trapezio.
% INT=TRAPEZI(A,B,M,FUN) calcola un'approssimazione dell'integrale della h=(b-a)/m;
x=a:h:b;
y=fun(x);
if size(y)==1
    dim=length(x);
    y=y.*ones(dim,1);
end
int=h*(0.5*y(1)+sum(y(2:m))+0.5*y(m+1));
end
```

Codice 3: Soluzione 1

```
function int = simpson(a,b,m,fun)
% simpson formula composita di Simpson.
% INT=SIMPSON(A,B,M,FUN) calcola un'approssimazione dell'integrale della h=(b-a)/m;
x=a:h/2:b;
y=fun(x);
if size(y)==1
    dim= length(x);
    y=ones(dim,1).*y;
end
int=(h/6)*(y(1)+2*sum(y(3:2:2*m-1))+4*sum(y(2:2:2*m))+y(2*m+1));
end
```

Soluzione 2

Osserviamo in scala logaritmica l'andamento degli errori in funzione di h sapendo che in questo tipo di grafici a rette di pendenza maggiore corrispondono metodi di ordine più elevato. Come previsto dalla teoria le formule composite del punto medio e del trapezio sono accurate di ordine 2, mentre quella di Simpson è di ordine 4.

Codice 4: Soluzione 2

```
a = 0;
b = 2*pi;
f = @(x) x.*exp(-x).*cos(2*x);
int_e = (3*(exp(-2*pi) - 1) - 10*pi*exp(-2*pi))/25;
int1 = []; int2 = []; int3 = [];
err1 = []; err2 = []; err3 = [];
ord1 = []; ord2 = []; ord3 = [];
H = [];
M = 2.^(4:8);

for m = M
    h=(b-a)/m;
    H = [H, h];

    int_pm = puntomedio(a,b,m,f);
    int1 = [int1, int_pm];
    err1 = [err1, abs(int_pm - int_e)];


    int_t = trapezi(a,b,m,f);
    int2 = [int2, int_t];
    err2 = [err2, abs(int_t - int_e)];

    int_cs = simpson(a,b,m,f);
    int3 = [int3, int_cs];
    err3 = [err3, abs(int_cs - int_e)];
end

loglog(H, err1,'-*')
hold on
grid on
loglog(H, err2,'-o')
loglog(H, err3,'-p')

l = length(M);
for i = 2:l
    ord1 = [ord1, log2(err1(i-1)/err1(i))];
    ord2 = [ord2, log2(err2(i-1)/err2(i))];
    ord3 = [ord3, log2(err3(i-1)/err3(i))];
end
```

Soluzione 3

L'errore di quadratura commesso con la formula composita del punto medio può essere maggiorato con $\frac{(b-a)^3}{24m^2}\max_{x\in[a,b]}|f''(x)|$ essendo $[a,b]$ l'intervallo di integrazione e m il numero (incognito) di sottointervalli. La funzione f_1 è derivabile con continuità per ogni ordine. Con uno studio grafico, si deduce che $|f_1''(x)| \leq 2$ nell'intervallo considerato. Affinché dunque l'errore sia minore di 10^{-4} si dovrà avere $2 \cdot 5^3/(24m^2) < 10^{-4}$ cioè $m > 322$. Anche la funzione f_2 è derivabile per ogni ordine. Con un semplice studio si ricava che $\max_{x\in[0,\pi]}|f_2''(x)| = \sqrt{2}e^{3\pi/4}$, di conseguenza affinchè l'errore sia minore di 10^{-4} dovrà essere $m > 439$. Si noti che le stime ottenute maggiorano ampiamente l'errore e, di conseguenza, il numero minimo di intervalli che garantisce un errore inferiore alla tolleranza fissata è assai minore (ad esempio, per f_1 bastano soltanto 71 intervalli). La funzione f_3 non ha derivata prima definita in $x = 0$ e $x = 1$: non si può quindi applicare la stima dell'errore riportata in quanto $f_3 \notin \mathbb{C}^2([0,1])$. Studiando le derivate quarte delle due funzioni osserviamo che il massimo di $|f_1^{(4)}(x)|$ è limitato da $m_1 \approx 23$, quello di $|f_2^{(4)}(x)|$ da $M_2 \approx 18$. Di conseguenza si trova nel primo caso $h < 0.21$ e nel secondo $h < 0.16$.

Codice 5: Soluzione 3

```
syms t
tol = 1.e-4;

f1 = 1./(1 + (t-pi).^2);
f_1 = matlabFunction(f1);
a_1 = 0; b_1 = 5; err_1 = []; x = linspace(a_1,b_1);
i_1 = double(int(f1,a_1,b_1));
f1_tt = matlabFunction(diff(f1,2));
f1_tt_max = max(abs(f1_tt(x)));
m_1 = sqrt((b_1 - a_1)^3*f1_tt_max/(24*tol));

f1_tttt = matlabFunction(diff(f1,4));
f1_tttt_max = max(abs(f1_tttt(x)));
m_1_cs = ((b_1 - a_1)^5*f1_tttt_max/(180*16*tol))^(1/4);

f2 = exp(t).*cos(t);
f_2 = matlabFunction(f2);
a_2 = 0; b_2 = pi; err_2 = [];
i_2 = double(int(f2,a_2,b_2));

f3 = sqrt(t.*(1-t));
f_3 = matlabFunction(f3);
a_3 = 0; b_3 = 1; err_3 = [];
i_3 = double(int(f3,a_3,b_3));

for m = 1:25:500
    int_1 = puntomedio(a_1,b_1,m,f_1);
    err_1 = [err_1, abs(int_1 - i_1)];

    int_2 = puntomedio(a_2,b_2,m,f_2);
    err_2 = [err_2, abs(int_2 - i_2)];

    int_3 = puntomedio(a_3,b_3,m,f_3);
    err_3 = [err_3, abs(int_3 - i_3)];
end

semilogy(err_1,'-o')
grid on
hold on
semilogy(err_2,'-*')
semilogy(err_3,'-p')
```

Soluzione 4

Essendo la misura del semiasse maggiore nota in milioni km, richiedere precisione di 10^4 km equivale a chiedere che l'errore di quadratura sia al più $tol = 10^4$. Per garantire ciò sfruttiamo la formula (dell'errore da cui determiniamo il minimo m. Si noti che prima abbiamo definito la funzione f come variabile simbolica, l'ab- biamo derivata simbolicamente e convertito quest'ultima in un function handle ($f2$) per valutarne il massimo su un insieme di 10000 punti equispaziati tra 0 e 2π. Otteniamo $f2max \approx 4.1770e + 04$. Quindi otteniamo che se $m \geq 10$ l'errore soddisfa la precisione richiesta. La lunghezza calcolata dell'orbita terrestre tramite la formula dei trapezi è $L = 9.398989 \cdot 10^8$ km.

Codice 6: Soluzione 4

```
a = 149.60e6; e = 0.0167086; b = a*sqrt(1-e^2);
a_1 = 0; b_1 = 2*pi;
syms t;

f = sqrt(a^2*cos(t).^2+b^2*sin(t).^2);
f2 = matlabFunction(diff(f,2));
tt = linspace(a_1,b_1,10000);
f2max = max(f2(tt));

h = sqrt(6/(pi*f2max)*1.e4);
m = fix(2*pi/h)+1;

f1 = matlabFunction(f);
i_1 = trapezi(a_1,b_1,m,f1);
```