Calcolo Numerico (a.a. 2008/2009)
Prof. L. D'Amore
22 Dicembre 2008

Sviluppo di un elemento di software matematico per la Quadratura numerica

1. Sviluppare un integratore automatico, chiamato, ad esempio, `TRAPEZIO`, in linguaggio **C** per il calcolo dell'integrale definito di una funzione f su un intervallo $[a, b]$, basato sulla **formula trapezoidale composita** (strategia non adattativa), con un opportuno criterio di arresto, che restituisca all'utente:

   - una stima del valore dell'integrale *integ*,
   - una stima dell'errore di discretizzazione *error*,
   - un indicatore di errore, *ifail* che segnali se è stato raggiunto il massimo numero di valutazioni o se la tolleranza richiesta è stata soddisfatta.

2. Sviluppare un integratore automatico `QUADRA`, basato sulla **strategia adattativa locale o sulla strategia adattativa globale** (a scelta), che faccia uso della formula trapezoidale composita per la stima dell' errore di discretizzazione. Utilizzare una opportuna struttura dinamica (tipo *lista concatenata*) per l'implementazione della pila in cui memorizzare le informazioni relative ai sottointervalli esaminati. In particolare:

   - **Se si utilizza la strategia adattiva locale** eseguire il test sulla tolleranza locale e sulla minima ampiezza dell'intervallo.
     - ◇ Se la tolleranza locale non è soddisfatta, suddividere l'intervallo in due sottointervalli e memorizzare le informazioni relative all'intervallo destro (sovrascrivendo la testa della pila) ed aggiungendo un nuovo nodo per la memorizzazione delle informazioni relativo all'intervallo sinistro;
     - ◇ altrimenti, aggiornare i valori che forniranno, in output, una stima dell'errore e dell'integrale e passare ad esaminare l'intervallo di destra.
   - **Se si utilizza la strategia adattattiva globale** eseguire il test sulla tolleranza e sulla minima ampiezza dell'intervallo.
     - ◇ Se la tolleranza non è soddisfatta, cercare l'intervallo in cui la stima dell'errore è massima in modulo, dividere l'intervallo in due ulteriori sottointervalli e calcolare le stime del modulo dell'errore e dell'integrale;
     - ◇ altrimenti, memorizzare le stime dei valori degli integrali e degli errori di discretizzazione relativi agli intervalli considerati.

   Per entrambe le strategie:

- Aggiornare, ad ogni passo, il *numero delle valutazioni di funzione*.
- stabilire un opportuno criterio di arresto, utilizzando in particolare un indicatore di errore $ifail$ che restituisca all'utente:
  - 0 se l'esecuzione si è conclusa correttamente (tolleranza soddisfatta);
  - 1 se è stato raggiunto il massimo numero di valutazioni della funzione integranda ($nval \geq maxval$);
  - 2 in presenza di errori di input, ovvero $tol < 0$ o $maxval < 3$, essendo 3 il minimo numero di valutazioni richieste dall'algoritmo che implementa la formula trapezoidale composita.
- Segnalare, inoltre, mediante un ulteriore indicatore di errore, $ifail2$, se durante l'esecuzione del programma è stata raggiunta, almeno una volta, la minima ampiezza possibile per la suddivisione in sottointervalli dell'intervallo di integrazione.
- Stampare, infine, il contenuto della pila, se l'implementazione si arresta con `numnod`$\neq$`0`.

3. Si realizzino test per il calcolo di integrali definiti, *confrontando*, a parità di tolleranza richiesta sulla stima dell'integrale, *il numero di valutazioni di funzione* effettuate dalla subroutine `TRAPEZIO` con quelle effettuate dalla `QUADRA`; osservare, inoltre, quale delle due procedure fornisce *risultati più accurati* attraverso il confronto tra le stime calcolate degli errori di discretizzazione.

4. Sia f la funzione integranda; assegnare un intervallo di punti di valutazione, valutare, mediante `matlab`, la f nei punti di tale intervallo [1] e tracciare il suo grafico con le istruzioni:

   $>>$ $x = [a : h : b]$

   $>>$ $y = f(x)$

   $>>$ `plot(x,y)`

   dove a e b sono gli estremi dell'intervallo di integrazione e h è il passo di discretizzazione dell'intervallo $[a, b]$.

[1] La funzione integranda, ad esempio $f = |sin(x)|$, può essere scritta sottoforma di function, in uno script `fun.m` del tipo:

```
function y=fun(x)
y=abs(sin(x))
end
```

o, assegnato il vettore z dei punti di valutazione, valutata dal prompt di `matlab` come

```
>> fz=abs(sin(z))
```

Inserire, opportunamente, nel codice realizzato, delle stampe affinchè l'esecuzione del software restituisca i nodi in cui è valutata la funzione integranda, secondo i passi della strategia implementata (adattativa e non adattativa).
Sia *nod* il vettore colonna contenente tali nodi, indicarli nella figura in cui è tracciato il grafico di f, con le istruzioni `matlab`

```
>> hold on
>> plot(nod,zeros(size(nod,1)))
```

Ponendo particolare attenzione all'andamento della funzione integranda ed ai risultati forniti dagli elementi di software implementati, realizzare opportune considerazioni sull'*efficienza* delle strategie implementate, in relazione all'andamento della funzione integranda.

5. **Esercizi §4.4: 1,4,5,6,7,8,9,10**

6. In `matlab` la funzione `quad` implementa una strategia adattativa basata sulla formula di Simpson. Digitando

   ```
   help quad
   ```

   si osserva che [2]:

   ```
   [Q,fcnt]=quad('fun',a,b,tol)
   ```

   restituisce una stima dell'integrale, `Q`, ed il numero di valutazioni della funzione integranda necessarie per ottenerla, `fcnt`. Tuttavia, aggiungendo un parametro di input non nullo, ad esempio $tab = 1$, con

   ```
   [Q,fcnt]=quad('fun',a,b,tol,tab)
   ```

   si visualizza la tabella contenente i valori

   ```
   [fcnt a b-a Q]
   ```

   dove `fcnt` è il numero di valutazioni, `a` il primo estremo dei sottointervalli esaminati, `b-a` la sua ampiezza, `Q` la stima dell'integrale calcolata su ciascun sottointervallo.
   Eseguendo le istruzioni fornite dai tre esempi seguenti, si visualizza il

[2] Nell'utilizzo di `quad` la funzione deve essere specificata realizzando una function `fun.m` del tipo:

```
function y=fun(x)
y=abs(sin(x))
end
```

se, ad esempio, $f = |sin(x)|$.

grafico della funzione integranda nonché, su di esso, il valore che la funzione assume negli estremi inferiori dei sottointervalli esaminati:

```
%
% Funzione 1
%
a=0.00001;
b=2.0;
tol=0.000001;
F = inline('cos(1./x)');
diary('Funzione1.diary')
[Q,FCNT]=quad(F,a,b,tol,1);
diary off
```

Aprire il file 'Funzione1.diary' e cancellare le righe non appartenenti alla tabella. Procedere, dunque, con le istruzioni in ambiente `matlab`:

```
load 'Funzione1.diary'
infa=Funzione1(:,2);
x=[a:1e-3:b];
figure
plot(infa,F(infa),'+',x,F(x));
legend('F(a)','F(x)')
```

Analogamente, si svolga il seguente esercizio:

```
clear
close all
%
% Funzione 2
%
a=2.0;
b=10.0;
tol=0.000001;
G = inline('log(x+exp(x))');
diary('Funzione2.diary')
[Q,FCNT]=quad(G,a,b,tol,'graf');
diary off
```

Aprire il file 'Funzione2.diary' e cancellare le righe non appartenenti alla tabella. Procedere, dunque, con le istruzioni in ambiente `matlab`:

```
load 'Funzione2.diary'
infa=Funzione2(:,2);
x=[a:1e-3:b];
figure
plot(infa,G(infa),'+',x,G(x));
legend('G(a)','G(x)')
```

Si svolga, infine, il seguente esercizio:

```
clear
close all
%
% Funzione 3
%
a=0.;
b=1.;
tol=0.001;
H = inline('(x.^(0.1)).*(1.2-x).*(1-exp(20*(x-1)))');
diary('Funzione3.diary')
[Q,FCNT]=quad(H,a,b,tol,'graf');
diary off
```

Aprire il file 'Funzione3.diary' e cancellare le righe non appartenenti alla tabella. Procedere, dunque, con le istruzioni in ambiente `matlab`:

```
load 'Funzione3.diary'
infa=Funzione3(:,2);
x=[a:1.0e-3:b];
figure
plot(infa,H(infa),'+',x,H(x));
legend('H(a)','H(x)')
```

Si ripeta l'esercizio precedente e si osservi il grafico di ciascuna delle due funzioni, con particolare attenzione all'andamento della funzione

$$x^{\alpha}(1.2-x)(1-e^{\beta(x-1)}),$$

in relazione alle diverse scelte dei parametri α e β. Si confrontino i risultati numerici restituiti dal proprio software e da `quad` di `matlab`.