Numerical Methods 2014–15 Assignment 1 — Adaptive Quadrature

Due: 5pm on 21/11/2014

See also the general information on assignments. Your submission must be a `.pdf` file containing a written account of your solutions to all the Questions below, together with any `.m` files created for use in MATLAB. All files must be zipped together in a single file and uploaded to Moodle by the submission deadline. **Do not deviate from the submission instructions.**

Background

Error estimates for quadrature rules were derived in lectures, but we did not consider mechanisms for (a) deciding just how many integration nodes are necessary to obtain a required accuracy and (b) for minimizing this number. This number will depend critically on the form of the integrand; if it oscillates rapidly then a large number of points is required; if it varies only very slowly, then a small number of points is sufficient. A procedure for concentrating the integration nodes of a given quadrature rule in subintervals where they are needed to achieve a desired accuracy with minimal effort is called *adaptive quadrature*. This Assignment will guide you in a step-by-step process of deriving, implementing and testing an *adaptive Simpson's rule*.

Simpson's and composite Simpson's rules

Consider the three-point closed simple Simpson's rule $S(f;a,b)$ with error E as introduced in lectures

$$\int_a^b f(x)\mathrm{d}x = S(f;a,b) + E(f;a,b), \qquad S(f;a,b) = \frac{(b-a)}{6}\left(f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right). \tag{1}$$

Question 1 *Using appropriate Taylor expansions, show that the error in the simple Simpson's rule (1) is*

$$E(f;a,b) = -\frac{(b-a)^5}{2880} f^{(iv)}(\xi), \qquad \textit{for some } \xi \in [a,b]. \tag{2}$$

(Marks: 5)

Question 2 *Using (1), derive the composite Simpson's rule $S_c(f;a,b,N)$ for N subintervals of $[a,b]$ of equal length. In the process you should (a) define an appropriate step length h; (b) index the integration nodes appropriately, and finally (c) provide an expression in the form of a weighted sum of function values $f(x_j)$.* (Marks: 5)

Question 3 *The error in the composite Simpson's rule $S_c(f;a,b,N)$ is given by*

$$E_c(f;a,b,N) = -\frac{(b-a)^5}{2880\, N^4} f^{(iv)}(\eta), \qquad \textit{for some } \eta \in [a,b]. \tag{3}$$

Is it possible to derive this expression directly from equation (2) without further assumptions? Why or why not? Even if possible to use (2), explain how you would obtain (3) using Taylor expansions, i.e. start from your formula for $S_c(f;a,b,N)$ derived in Question (2) and (a) write out the expansion of $f(x_j)$ at a general node x_j, (b) integrate the expansion of $f(x)$ to get an expansion of $\int_a^b f(x)\mathrm{d}x$, and finally (c) explain how you would form the error $E_c(f;a,b,N)$ but ***do not perform further calculations.*** (Marks: 5)

Question 4 *Implement the composite Simpson's rule you derived in Question 2 as a* MATLAB *function* `Sc(f,a,b,N)`. *Here* `f` *is a function handle holding the integrand $f(x)$ and* `a,b` *are the endpoints of the integration interval $[a,b]$ and* `N` *is the number of subintervals N where the simple rule is applied. You may find Listing 1 useful.* (Marks: 5)

The adaptive refinement procedure

An adaptive refinement procedure may be applied to any $(n+1)$-point quadrature rule Q as follows.

1. Consider an interval $[a,b]$.
2. Obtain a first integral approximation Q_1 on $[a,b]$ using $n+1$ nodes.
3. Obtain a second integral approximation Q_2 on $[a,b]$ using $2n+1$ nodes.
4. Obtain an estimate of the error using Q_1 and Q_2.
5. If the error estimate is less than some required tolerance ε accept a suitably corrected value of Q_1 (or Q_2).
6. Otherwise split $[a,b]$ into two equal subintervals and repeat the above process over each subinterval with tolerances adjusted to $\varepsilon/2$ (why $\varepsilon/2$?).

Note that this is a "recursive procedure" - it is most easily defined in terms of itself and ends when a "stopping condition" is met. To implement the stopping condition we now need to find an estimate of the error on a subinterval.

Question 5 *Consider an interval $[a,b]$ small enough to justify the assumption $f^{(iv)}(x)=\text{const}$ on $[a,b]$. With this assumption, eliminate $f^{(iv)}(x)$ between equations (2) and (3) to show that*

$$E(f;a,b,2) \doteq \frac{1}{15}\Big(S_c(f;a,b,2)-S(f;a,b,1)\Big) \quad \text{and then} \quad \int_a^b f(x)\mathrm{d}x = S_c(f;a,b,2)+E(f;a,b,2), \tag{4}$$

where $\doteq$ denotes equality under the stated assumption. (Marks: 6)

Question 6 *Implement an adaptive Simpson's rule as a* MatLab *function* `Sa(f,a,b,eps)` *using equations (4) at step 5 of the adaptive refinement procedure. The code in Listing 2 may be useful. Make sure you provide comments in your script file using* `%` *to explain what you are doing.* (Marks: 4)

Illustration and validation

Question 7 *Compute an approximation of the integral*

$$\int_0^2 \sin\left(1-25\,\mathrm{erf}\left(\frac{x-1}{0.2\sqrt{2}}\right)\right)\mathrm{d}x \tag{5}$$

using your MatLab *functions of the composite and the adaptive Simpson's rules from Questions 4 and 6. In order to illustrate the adaptive refinement procedure produce a plot showing (a) a graph of the integrand as well as (b) the location of the integration nodes. Compare that with a similar plot obtained when using your composite rule* MatLab *function.* (Marks: 5)

Question 8 *Compute an approximation of the integral*

$$I = \int_{-3}^{5} \exp(-50(x-1)^2)\mathrm{d}x \tag{6}$$

using your MatLab *function of the adaptive Simpson's rule from Question 6 for 10 different values of the tolerance ε from 10^{-6} to 10^{-12}. The integral has the exact value $I=\mathrm{erf}\big(20\sqrt{2}\big)\sqrt{2\pi}/10$. Plot graphs to demonstrate that your adaptive Simpson's function produces error smaller than the required tolerance.* (Marks: 5)

CONTINUED

Listing 1: Code for the composite trapezium rule T_c.

```matlab
function out = Tc(func,a,b,N)
% Tc(func,a,b,N)
% This function calculates the integral of func on the interval [a,b]
% using the Composite Trapezium rule with N subintervals.

  % Partition [a,b] into N subintervals
x = linspace (a,b,N+1);

% Evaluate f at the end points of the N subintervals
fx = func(x);

% Define the length of a subinterval
h =(b-a)/N;

% Apply the composite trapezium rule
out = h*(0.5*fx(1)+ sum(fx(2:N))+0.5*fx(N+1));

end
```

Listing 2: Incomplete code for an adaptive Simpson's rule S_a.

```matlab
function out=Sa(f,a,b,eps)
%
% Provide comments to your code, including a "help" facility.
%
c=(a+b)/2;
Q1 = Sc(f,a,b,1);
Q2 = Sc(f,a,b,2);
if ( ##### stopping condition goes here     #####)
     ##### integral approximation goes here #####
else
   out= Sa(f,a,c,eps/2.0) + Sa(f,c,b,eps/2.0);
end
end
```

END