University of Toronto
CSC 270H/B70H: Fundamental data structures and techniques
Fall 1999

Assignment 1 – Quadrature

Due Date: October 12, 1999

Purpose:

There are two purposes for this assignment:

- practice writing a basic C program. The solution program includes:
  input and output,
  arithmetic expressions,
  loops, and
  defining and using functions.
- provide experience with numerical methods, specifically quadrature.

Introduction:

It is difficult to integrate some functions. In these cases an approximate value for the integral may be adequate.

There are several rules for approximating the value of a two dimensional integral. You have studied the rectangle (called rectangle left endpoint in the readings), midpoint (called rectangle midpoint in the readings), trapezoid and Simpson's rules in the course. Each rule approximates the value of the integral of a function by finding the area of a shape bounded by the x-axis, x-values at the extremes of the integral range and a line through one or more of the points on the line defined by the function. These methods are fully described in the numerical section of the Course Readings, pages 432 and 433.

For functions that do not "fit" the rules, a better approximation may be obtained by subdividing the integral range into subintervals and summing the approximate areas of the subintervals. This method is called composite quadrature. It is described more fully on page 439. It is assumed that dividing the integral interval into more subintervals will produce a better approximation. (Theoretically this is true. Practically because of floating-point error and round-off error this is not true but more about that later in the course.)

Each of the rules described above can be extended to approximate three dimensional (3D) integrals. Instead you are finding the volume bounded by six surfaces, the plane defined by the x and y axes (the $x = 0$ $y = 0$ plane), the four planes defined by the x-values and the y-values at the extremes of the integral range, and a surface that passes through one or more of the points on the surface defined by the function.

In this assignment, you will extend the rectangle, midpoint and trapezoid rules to approximate 3D integration. (You do not have to extend Simpson's rule.)

Details:

For composite 3D integration, the integral range is subdivided into rectangular intervals. Each interval/rectangle is bounded in the x direction by a pair of x-values, e.g. x_1 and x_2, where $x_1 < x_2$. Each interval/rectangle is bounded in the y direction by a pair of y-values, e.g. y_1 and y_2, where $y_1 < y_2$.

For each of the three rules, you will approximate the integral by calculating the volume bounded by six planes. Five of the planes are the $x = 0$ $y = 0$ plane, and the four planes defined by the x and y-values at the extremes of the integral interval.

For the rectangle rule, the sixth surface is a plane parallel to the $x = 0$ $y = 0$ plane. For the interval $[x_1, x_2]$ $[y_1, y_2]$, this plane passes through the point $(x_1, y_1, f(x_1, y_1))$. The volume of the interval is $(x_2 - x_1) * (y_2 - y_1) * f(x_1, y_1)$.

For the midpoint rule, the sixth surface is a plane parallel to the $x = 0$ $y = 0$ plane. For the interval $[x_1, x_2]$ $[y_1, y_2]$, this plane passes through the point $((x_1 + x_2)/2, (y_1 + y_2)/2, f((x_1 + x_2)/2, (y_1 + y_2)/2))$. The volume of the interval is $(x_2 - x_1) * (y_2 - y_1) * f((x_1 + x_2)/2, (y_1 + y_2)/2)$.

For the trapezoid rule, the sixth surface is a plane perpendicular to the plane $y = 0$. For the interval $[x_1, x_2]$ $[y_1, y_2]$, this plane passes through the two points $(x_1, y_1, f(x_1, y_1))$ and $(x_2, y_1, f(x_2, y_1))$. The volume of the interval is $(x_2 - x_1) * (y_2 - y_1) * ((f(x_1, y_1) + f(x_2, y_1))/2)$.

What to do:

In this assignment, you will implement composite 3D quadrature for each of the three rules. Design and code a C program to implement your solution. We will provide the code for the input of six values: the range of the integral in the x and y directions, and the number of subintervals in the x and y directions. We will also provide the code for reporting the approximate value of the integral. Test your program with several different functions. In table form report the result for each function, for each of the different rules, and for several different numbers of subintervals.

Programming advice and imperatives:

Develop your program in a modular way. Develop the program with appropriate subtasks defined as functions. Divide the program into appropriate separate files. The file that stores the function being integrated must be called `func.c`. (This is required because your program will be tested with functions of our choosing for marking purposes.) You must use the code that we have provided in your solution.

What to hand in:

A printout of your program files.
The outputs of several test runs illustrating that your program works correctly.
The table of results of running your program on several different functions.
A brief report describing the influence of the different rules and the number of subintervals on the results of your approximations.

Assignment submission:

For details regarding handing in your program refer to your campus web page. You must submit your program electronically for marking purposes.