

**CS 205 - Programming for the Sciences**  
**Spring 2008 - Programming Assignment 2**  
**20 points**

**Out: January 29, 2008**

**Due: February 4, 2008**

**Notes on Numerical Integration**

Many mathematical functions cannot be integrated in closed form. One such function is given by

$y = \int_a^b e^{-x^2} dx$ . When such functions need to be integrated over a closed interval, satisfactory results can often be obtained by doing numerical integration. There are several methods of doing numerical integration and in this assignment we will explore two of them: Rectangular integration and trapezoidal integration.

*Rectangular Integration with Backward Differences:*

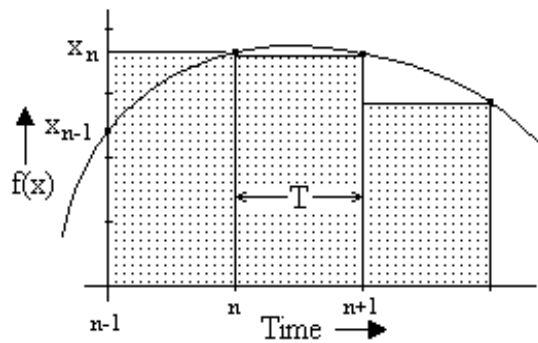
Figure 1 shows an arbitrary function of time which has been sampled at regular intervals. If the width of a single sample is  $T$ , then the area of a rectangle between point  $n-1$  and point  $n$  is  $T$  multiplied by  $x_n$ . If we write  $y_a = \int_0^a f(x) dx$  then we can approximate  $y$  as a succession of rectangular areas like this:

$$y_n - y_{n-1} = T \cdot x_n$$

or

$$y_n = y_{n-1} + T \cdot x_n \tag{1}$$

Thus if we have a sequence of values of  $f(x)$  for  $x$  going from 0 to say 5, we could find the value of the integral of  $f(x)$  for the interval 0 to 5 by using Equation 1 successively.



Rectangular Integration using  
Backward Differences

**Figure 1**

A function of  $x$  is sampled and divided into rectangles to estimate the area under the curve.

To do this with a computer program we need a loop that uses Equation 1 and sums up the area of each square. The pseudocode for such a program might look like this:

1. Obtain values for endpoints called xStart and xEnd, and the value of the width (T) called increment
2. Initialize x to xStart + increment
3. Initialize sum to 0
4. While x is less than or equal to xEnd do
  - 4.1 Add f(x) times increment to sum
  - 4.2 Add increment to x
5. Display sum

*Trapezoidal Integration:*

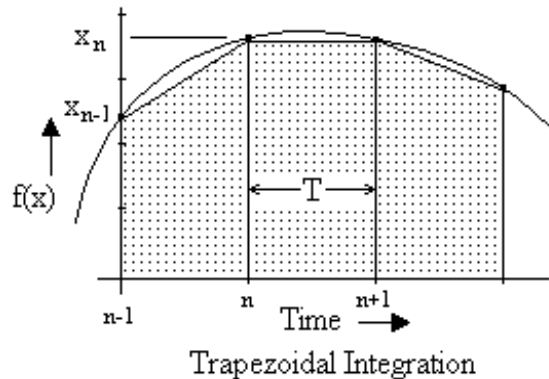
For trapezoidal integration we divide the function into trapezoids spaced at the sample intervals as shown in Figure 2. The area of a single trapezoid can be found by taking the average of the two ordinates and multiplying it by the width, T. The relevant equations are:

$$y_n - y_{n-1} = T \cdot \frac{x_n + x_{n-1}}{2}$$

or

$$y_n = y_{n-1} + T \cdot \frac{x_n + x_{n-1}}{2} \tag{2}$$

Analogously to rectangular integration, we can use Equation 2 successively to find the value of the integral.



**Figure 2**

A function of x is sampled and divided into trapezoids to estimate the area.

The pseudocode to do trapezoidal integration might look like this:

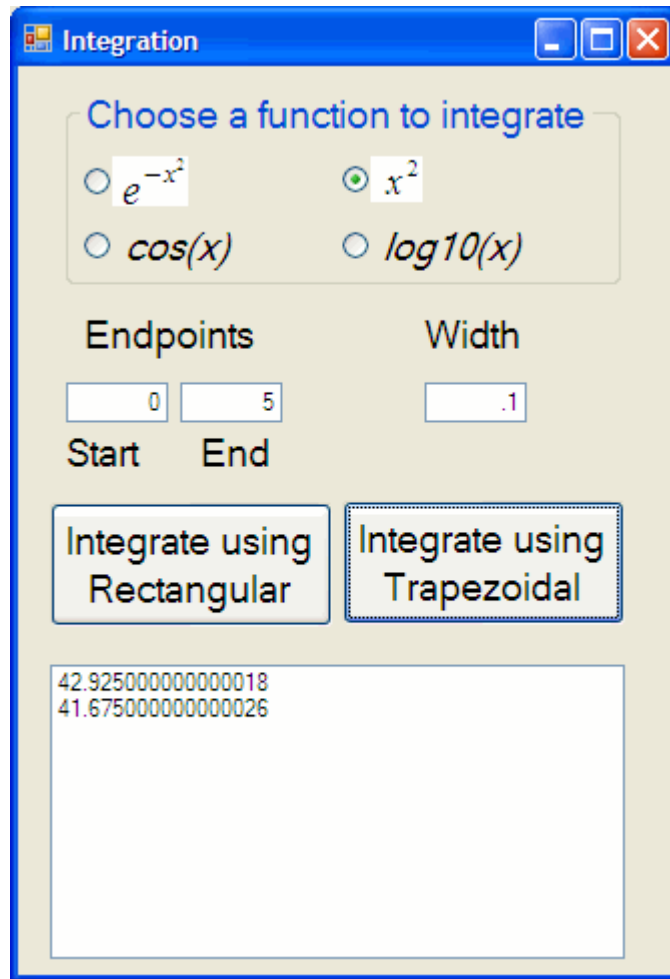
1. Obtain values for endpoints called xStart and xEnd, and the value of the width (T) called increment
2. Initialize the left ordinate called xOld to start
3. Initialize the right ordinate called x to start + increment
4. Initialize sum to 0
5. While x is less than or equal to end do
  - 5.1 Add [(f(x) + f(xOld))/2] times increment to sum
  - 5.2 Set xOld to x

- 5.3 Add increment to x
- 6. Display sum

### Assignment

For this assignment, download the compressed solution folder Integration.zip from the course webpage and extract the solution folder. Double-clicking on the solution file Integration.sln will launch C# with this project.

You are given an interface for a program integrating up to 4 different single-variable functions using either rectangular or trapezoidal integration shown below. The code for determining which function has been chosen has been provided and explained later.



The interface is used in the following way:

1. Choose one of the functions to integrate. ( $\cos(x)$  returns the cosine of  $x$  where  $x$  is in radians;  $\log_{10}(x)$  returns the log base-10 of  $x$ .)
2. Enter starting and ending endpoints for  $x$
3. Enter the width of the samples

4. Click on appropriate button to do rectangular or trapezoidal integration
5. The results appear in the listbox successively. The example above shows the result of first doing rectangular integration, then doing trapezoidal integration.

You are to implement the code for actually doing the integration based on the notes above. The code for rectangular integration must go into the `rectIntegrate_Click` handler, and the code for trapezoidal integration must go into the `trapIntegrate_Click` handler, as indicated in the provided comments for each handler. This program has been set up so that the use of the name "f" will call the appropriate function. (I.e., if the argument variable name is x, then calling any function is  $f(x)$ . How this is done is explained below.) Note that you are responsible for converting the textbox input into an appropriate value and writing the result to the provided listbox as demonstrated in class.

### **What to submit**

For full credit, a compressed (zipped) solution folder containing a C# project must be submitted as an attachment to an email to Dr. Hwang ([hwang@evansville.edu](mailto:hwang@evansville.edu)) no later than 4:30pm on the due date.

To create a compressed solution folder find the solution folder via Windows Explorer. Right-click on the solution folder, select Send To, then select Compressed (zipped) folder. This will create a compressed folder of the same name as the solution folder with extension .zip and an icon of a folder with a zipper.

### **Also briefly discuss the following questions based on using your program:**

1. In general, what effect does the width of the samples have on the accuracy of the results? Why?
2. In general, which integration method provides more accurate results? Why?

You may provide your answers directly in the email you send or attach a separate document.

### **GUI notes**

The controls for providing the choices of functions are 4 radio buttons in a group box. As the name implies, a group box allows controls to be grouped together. This is significant for radio buttons, since all radio buttons in a group are tied together automatically so that only one of the buttons may be checked at any given time. The radio button property `Checked` is set to true when a radio button is checked and set to false for all other radio buttons in the group. This property may be set in the Form Designer (which ensures only one of a group is set) and is the default checked button when the program starts.

For those that are interested (i.e., this is not part of the course material), the code for allowing all 4 functions be accessed by the name "f", is provided by the use of a programming construct called a delegate and is covered in Chapter 16. The method `SetFunction` determines which radio button is checked and then sets "f" to be the function indicated. As noted in the comments, the function can be a user written method, or it can be a built-in method. The only requirement for this delegate is that the method must have only one double argument and return a double result.