Term Project Part 2: Numerical Integration V1.0

Islam Faisal Ibrahim Abdelrasoul

May 10, 2016

Abstract

In this report, we implement three methods for numerical integration; namely trapezoidal rule, Simpson's $\frac{1}{3}$ and $\frac{3}{8}$ rules. We provide the design for and implement an algorithm for integrating a function with given unequal segments. Finally, the implementation is tested against a given set of points.

1 Introduction

We describe the methods implemented and provide pseudocode for the main algorithm and flowchart as well as a test matrix to test the implementation. **The flowcharts capture the main idea while we leave the nitty-gritty details for the pseudocode.** We reserve this technical paper for the comment on the overall process. For the source code and the library documentation and folder structures, please see the readme.md if you have markdown supported by using atom [1] or viewing from github [2] or by directly opening readme.pdf.

2 Design

In this section, we provide the formulae and pseudocode as in [1] for each procedure of the integration methods as well as the main algorithm. In all the following sections, we assume that we are computing $I := \int_a^b f(x)dx$ for a continous real-valued function f. We omit the pseudocode for these methods because it is trivial and consists only of the formula.

2.1 Trapezoidal Rule

The trapezoidal rule uses the following formula to estimate the integral using two points:

$$I = (b-a)\frac{f(a)+f(b)}{2} \tag{1}$$

2.2 Simpson's $\frac{1}{3}$ rule

The Simpson's $\frac{1}{3}$ rule uses the following formula to estimate the integral using three points:

$$I = (b-a)\frac{f(x_0)+4f(x_1)+f(x_2)}{6} \tag{2}$$

2.3 Simpson's $\frac{3}{8}$ rule

The Simpson's $\frac{3}{8}$ rule uses the following formula to estimate the integral using four points:

$$I = (b-a)\frac{f(x_0)+3\left(f(x_1)+f(x_2)\right)+f(x_3)}{8} \tag{3}$$

2.4 Algorithm for integration with unequal segments

As in [1], the following algorithm uses a combination of the previously mentioned three rules to estimate the integral for a given set of unequally [3]-spaced data. It is a greedy algorithm that will try applying the 3/8 rule, then 1/3 rule then the trapezoidal rule. Figure 1 is the flowchart for the algorithm.

3 Implementation

We provide the matlab code of the implementation and some guides about the files in the readme.md file.

[1] `https://atom.io/`

[2] `https://github.com/decltypeme/smiley-epsilon`

[3] The set does not have to be unequally spaced though, but this naming is conventional.

Algorithm 1: Algorithm to estimate an integral based on unequally-spaced data

Data: result, n, x, f, tol
Result: The estimate of the integral
$h := x_2 - x_1$;
$k := 1$;
$sum := 0$;
for $j = 2 \rightarrow n + 1$ **do**
  $hf := x_{j+1} - x_j$;
  if $ABS(h - hf) < tol$ **then**
    if $k == 3$ **then**
      $sum := sum +$
       $simpson13(h, f_{j-3}, f_{j-2}, f_{j-1})$;
      $k := k - 1$;
    else
      $k := k + 1$;
    end
  else
    if $k == 1$ **then**
      $sum :=$
       $sum + trapezoidal(h, f_{j-1}, f_j)$;
    else
      if $k == 2$ **then**
        $sum := sum +$
         $simpson13(h, f_{j-2}, f_{j-1}, f_j)$;
      else
        $sum := sum +$
         $simpson38(h, f_{j-3}, f_{j-2}, f_{j-1}, f_j)$;
      end
      $k := 1$;
    end
  end
  $h := hf$;
end
$result := sum$;
return;

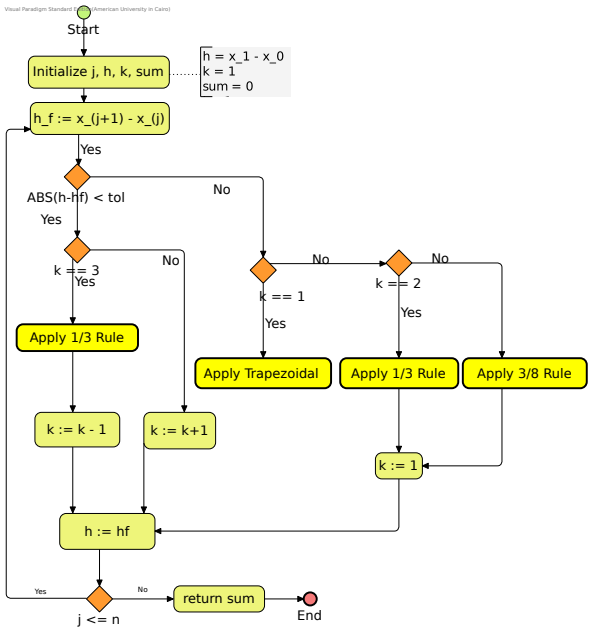

Figure 1: Flowchart of the algorithm for numerical integration with unequal segments

4 Testing

We test the implementation to find the given integral:

$$I := \int_0^{0.6} f(x)dx \tag{4}$$

where $f : \mathbb{R} \rightarrow \mathbb{R}$ such that

$$f(x) = 2e^{-1.5x} \tag{5}$$

using the following unequally-spaced data

$$X = \begin{pmatrix} 0.00 \\ 0.05 \\ 0.15 \\ 0.25 \\ 0.35 \\ 0.475 \\ 0.6 \end{pmatrix} \text{and} f(X) = \begin{pmatrix} 2.0000 \\ 1.8555 \\ 1.5970 \\ 1.3746 \\ 1.1831 \\ 0.9808 \\ 0.8131 \end{pmatrix} \tag{6}$$

After running the script test.m, we obtained.

$$I_{numerical} \approx 0.7913 \tag{7}$$

We know the solution analytically is

$$I_{analytic} = -\frac{4}{3}\left(e^{-1.5x}|_0^{0.6}\right) = -\frac{4}{3}(e^{-0.9} - e^0) \approx 0.7912 \tag{8}$$

Now, we compute the relative true error:

$$\epsilon_t = |\frac{I_{numerical} - I_{analytic}}{I_{analytic}}|100\% = 0.0126\% \quad (9)$$

References

[1] S. Chapra and R. Canale. *Numerical Methods for Engineers*. McGraw-Hill Education, 2009.