Rutgers CS323 (04), Spring 2017, Homework 7

Due at 11:55pm on May 1, 2017, submitted via Sakai

Numerical Integration

1 Problem Statement

For this assignment, you are asked to code four algorithms for numerical integrations and generate a figure which should be very similar to Figure 1.

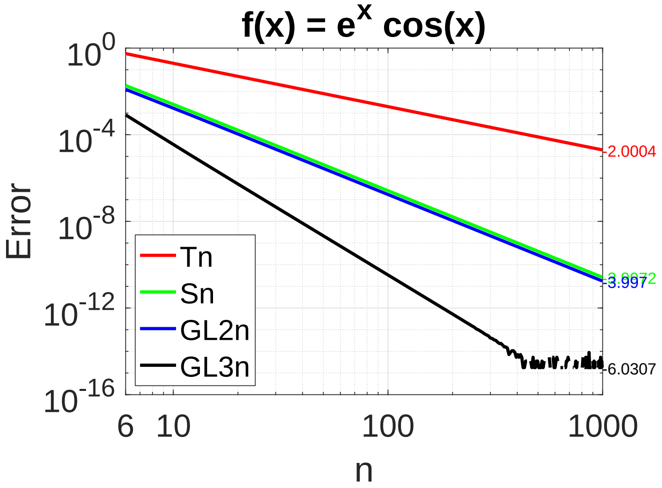

Figure 1: Absolute Errors of four numerical integration methods for approximating $I(f) = \int_0^\pi f(x)dx$ where $f(x) = e^x \cos x$.

Basically, we need to compute $I(f) = \int_0^\pi f(x)dx$ where $f(x) = e^x \cos x$. We divide the interval $[0,\ \pi]$ into n equal-length sub-intervals and implement the following four algorithms for approximating $I(f)$:

1. T_n: Trapezoidal rule.
2. S_n: Simpson's rule.
3. $GL_{2,n}$: Gauss-Legendre Quadrature with "$n = 2$" in each sub-interval.
4. $GL_{3,n}$: Gauss-Legendre Quadrature with "$n = 3$" in each sub-interval.

Note that when we use Gauss-Legendre Quadrature $GL_{2,n}$, we need to group the n sub-intervals into $n/2$ intervals and apply the algorithm with "$n = 2$" in each of the $n/2$ intervals. When we use Gauss-Legendre Quadrature $GL_{3,n}$, we need to group the n sub-intervals into $n/3$ intervals and apply the algorithm with "$n = 3$" in each of the $n/3$ intervals. In Figure 1, we plot the absolute errors for $n = 6 : 6 : 1000$. Pay attention to the subscript of the data points x_i when applying the formulas for $GL_{2,n}$ and $GL_{3,n}$ because we have to combine the original n sub-intervals.

The numbers (e.g., -2.0004) at the end of the lines indicate the slopes of the lines (in the log-log scale). Basically, for each method, we fit a straight line

$$\log(\text{Absolute Error}) = \beta_0 + \beta_1 \log n$$

The numbers at the end of the lines are the β_1 values. To do this, you will need to implement your own linear regression. For numerical reasons, please only use the portion with absolute error $> 10^{-14}$ to fit each line.

2 Coding Instruction

Your submission should have at least one file named TestNumInt.m . We will execute this function by simply typing "TestNumInt" from the matlab console and your code is expected to produce a figure very similar to Figure 1, **within one minute**. You can write as many functions as needed, either in separate files or in the same TestNumInt.m file. In summary, your code should

- implement the four numerical integration methods.
- implement a least square function.
- plot the results in a figure very similar to Figure 1.

3 Submission

You should submit all .m files needed in order to complete this assignment.