

Computational methods and applications (AMS 147)

Homework 5 - Due Sunday, March 3

Please submit to CANVAS a .zip file that includes the following Matlab functions:

```
int_midpoint_rule.m  
int_trapezoidal_rule.m  
int_Simpson_rule.m  
test_integration.m
```

Exercise 1 Consider the following integral of a function $f(x)$ on a finite interval $[a, b]$

$$I(f) = \int_a^b f(x) dx. \quad (1)$$

Write three Matlab/Octave functions implementing, respectively, the composite midpoint rule, the composite trapezoidal rule, and the composite Simpson rule to compute the numerical approximation of $I(f)$. Such functions should be of the form

```
function [I]=int_midpoint_rule(fun,a,b,n)      (composite midpoint rule)  
function [I]=int_trapezoidal_rule(fun,a,b,n)    (composite trapezoidal rule)  
function [I]=int_Simpson_rule(fun,a,b,n)         (composite Simpson rule)
```

Input:

fun: function handle representing $f(x)$
a, b: endpoints of the integration interval
n: number of evenly-spaced points in $[a, b]$ (including endpoints)

$$x_j = a + (j - 1)h, \quad h = \frac{b - a}{n - 1}, \quad j = 1, \dots, n.$$

Output:

I: numerical approximation of the integral (1).

Exercise 2 Use the functions you coded in Exercise 1 to compute the numerical approximation of the integral

$$I = \int_{-3}^1 \left[\frac{1}{1+x^2} \cos\left(\frac{3}{2}e^{-x^2}\right) - \frac{x^3}{30} \right] dx. \quad (2)$$

To this end, write a Matlab/Octave function

```
function [em,et,es] = test_integration()
```

that returns the following items:

- **em, et, es**: row vectors with components the absolute values of the integration errors

$$|I_{\text{ref}} - I_n| \quad n = 2, 3, \dots, 100 \quad (3)$$

obtained with the midpoint (vector **em**), trapezoidal (vector **et**) and Simpson (vector **es**) rules. Here,

$$I_{\text{ref}} = 1.6851344770476$$

is the reference value of the integral (2) while I_n is the numerical approximation obtained by using the composite midpoint, trapezoidal, and Simpson rules with n nodes.

- The function `test_integration()` should also return the plot of the integrand function appearing in (2) in `figure(1)`, and the plots of the errors **em**, **et** and **es** versus n in a log-log scale in `figure(2)` (one figure with three plots). (Hint: use the Matlab command `loglog()`).