Table of Contents

Assignment

Integration

```
% Name        : Mohamed Mafaz
% Roll Number : AM25M009
% Department  : Applied Mechanics

clc;
clear;
close all;

answer = 0.74306944;

% f =@(x) exp(-x^2);
f =@(x) 1 / ((1+x^3)^0.5);

a = 1;
b = 3;
```

Single Step

```
% Trapezoid
h = (b-a);
trap_integral = h/2 * (f(a) + f(b));
fprintf("Trapezoid Single step: %f\n", trap_integral)
fprintf("Trapezoid Single step ERROR: %f\n\n", abs(trap_integral-answer))

% Simpsion's 1/3rd
h = (b-a)/2;
simp_1_3_integral = (b-a)/6 * (f(a) + 4*f(a+h) + f(b));
fprintf("Simpsion's 1/3rd Single step: %f\n", simp_1_3_integral)
fprintf("Simpsion's 1/3rd Single step ERROR: %f\n", abs(simp_1_3_integral-
answer))

% Simpsion's 3/8rd
h = (b-a)/3;
simp_3_8_integral = (3*h/8) * (f(a) + 3*f(a+h) + 3*f(a+2*h) + f(b));
fprintf("Simpsion's 3/8th Single step: %f\n", simp_3_8_integral)
fprintf("Simpsion's 3/8th Single step ERROR: %f\n", abs(simp_3_8_integral-
answer))
```

```
Trapezoid Single step: 0.896089
Trapezoid Single step ERROR: 0.153020

Simpsion's 1/3rd Single step: 0.743141
Simpsion's 1/3rd Single step ERROR: 0.000071
Simpsion's 3/8th Single step: 0.742721
Simpsion's 3/8th Single step ERROR: 0.000348
```

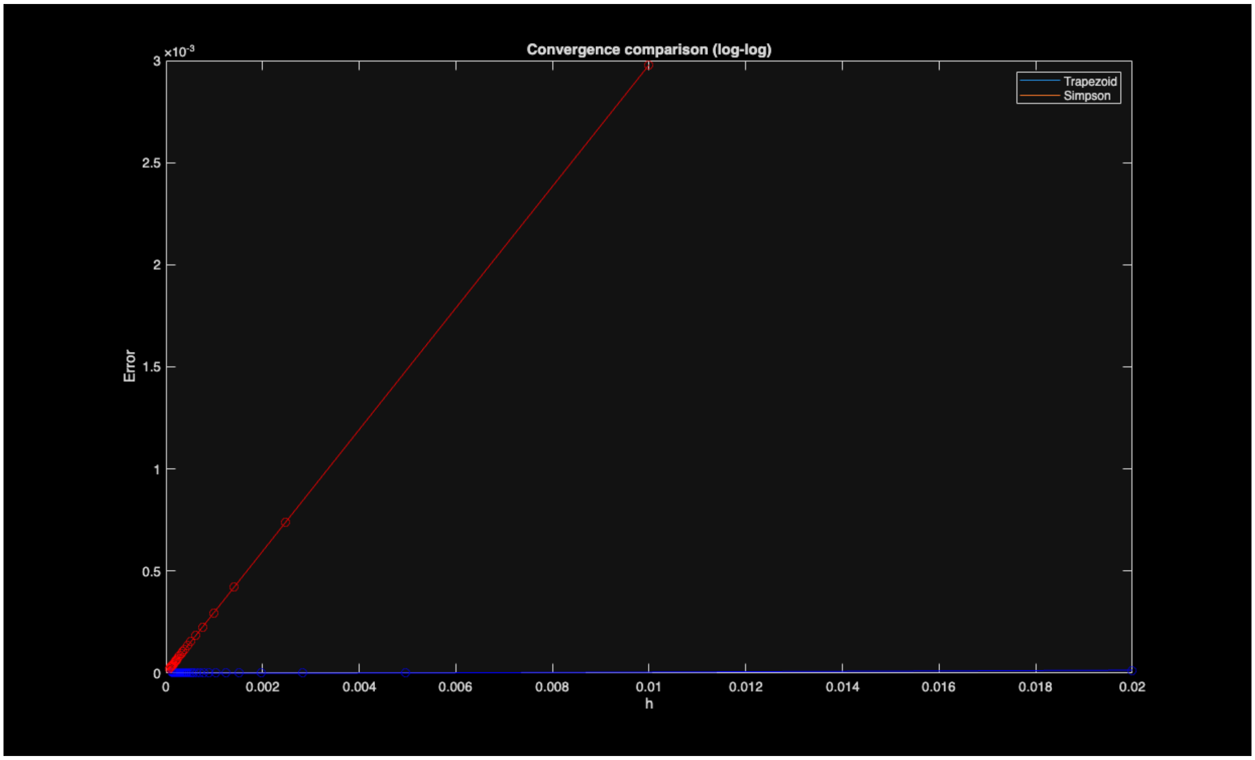
Multiple Steps

```
% Trapezoid
trap_mul_hs = [];
trap_mul_errors = [];
M = linspace(100, 15000, 50);

for m = 1:length(M)
    n = round(M(m)); % number of subintervals
    h = (b - a) / n;
    trap_mul_hs(end+1) = h;

    x_vals = linspace(a, b, n + 1);

    % Sum interior points only once multiplied by 2
    sum_interior = 0;
    for i = 2:n
        sum_interior = sum_interior + f(x_vals(i));
    end

    trap_mul = (h/2) * (f(a) + 2 * sum_interior + f(b));
    trap_mul_errors(end+1) = abs(answer - trap_mul);
end


figure()
plot(trap_mul_hs, trap_mul_errors)
xlabel('h');
ylabel('abs error');
title("Trapezoid Multi-Step optimal h value")
fprintf("\n\nTrapezoid Multi step: %f\n", trap_mul)
fprintf("Trapezoid Multi step ERROR: %f\n", abs(trap_mul - answer))
[~, idx_min] = min(trap_mul_errors);
opt_h = trap_mul_hs(idx_min);
fprintf("Trapezoid Optimal h: %f\n\n", opt_h)
hold on

% Simpsion's 1/3rd

simp_1_3_mul_hs = [];
simp_1_3_mul_errors = [];
% M = linspace(100, 1500, 10);
```

```
for m = 1:length(M)
    n = 2 * round(M(m)); % must be even

    h = (b - a) / n;
    simp_1_3_mul_hs(end+1) = h;

    x_vals = linspace(a, b, n + 1);

    sum_odd = 0;  % odd
    sum_even = 0; % even

    for i = 2:n
        if mod(i, 2) == 1 % odd index
            sum_odd = sum_odd + f(x_vals(i));
        else
            sum_even = sum_even + f(x_vals(i));
        end
    end

    simp_1_3_integral = (h/3) * (f(a) + 4 * sum_odd + 2 * sum_even + f(b));
    simp_1_3_mul_errors(end+1) = abs(answer - simp_1_3_integral);
end

fprintf("Simpsion's 1/3rd Multi step: %f\n", simp_1_3_integral)
fprintf("Simpsion's 1/3rd Multi ERROR: %f\n", abs(simp_1_3_integral-answer))
[~, idx_min] = min(simp_1_3_mul_errors);
opt_h = simp_1_3_mul_hs(idx_min);
fprintf("Simpsion's 1/3rd Multi Optimal h: %f\n\n", opt_h)

% figure()
plot(simp_1_3_mul_hs, simp_1_3_mul_errors)
xlabel('h');
ylabel('abs error');
title("Simpsion's 1/3rd Multi-Step optimal h value")
legend("Trap", "SImposon")


% PLotting
loglog(trap_mul_hs, trap_mul_errors, 'b-o')
hold on
loglog(simp_1_3_mul_hs, simp_1_3_mul_errors, 'r-o')
xlabel('h')
ylabel('Error')
legend('Trapezoid', 'Simpson')
title('Convergence comparison (log-log)')

p_trap = polyfit(log(trap_mul_hs), log(trap_mul_errors), 1);
p_simp = polyfit(log(simp_1_3_mul_hs), log(simp_1_3_mul_errors), 1);
fprintf('Trap order ~ %.2f\n', -p_trap(1))
fprintf('Simpson order ~ %.2f\n', -p_simp(1))
```

```
Trapezoid Multi step: 0.743069
```

```
Trapezoid Multi step ERROR: 0.000000
Trapezoid Optimal h: 0.000133

Simpsion's 1/3rd Multi step: 0.743050
Simpsion's 1/3rd Multi ERROR: 0.000020
Simpsion's 1/3rd Multi Optimal h: 0.000067

Trap order ~ -1.48
Simpson order ~ -1.00
```



Gauss Legendre

```
f_gl =@(x) 1 / sqrt(1+((x+2)^3));

% 2 point
x_2_1 = -1/sqrt(3);
x_2_2 = 1/sqrt(3);

w_2_1 = 1;
w_2_2 = 1;

gl_2 = (w_2_1 * f_gl(x_2_1)) + (w_2_2 * f_gl(x_2_2));
fprintf("Gauss Legendre 2 point: %f\n", gl_2)
fprintf("Gauss Legendre 2 point ERROR: %f\n\n", abs(gl_2-answer))

% 3 point
x_3_1 = -sqrt(3/5);
x_3_2 = 0;
x_3_3 = sqrt(3/5);
```

```
w_3_1 = 5/9;
w_3_2 = 8/9;
w_3_3 = 5/9;

gl_3 = (w_3_1 * f_gl(x_3_1)) + (w_3_2 * f_gl(x_3_2)) + (w_3_3 * f_gl(x_3_3));
fprintf("Gauss Legendre 3 point: %f\n", gl_3)
fprintf("Gauss Legendre 3 point ERROR: %f\n\n", abs(gl_3-answer))
```

```
Gauss Legendre 2 point: 0.742632
Gauss Legendre 2 point ERROR: 0.000437

Gauss Legendre 3 point: 0.743441
Gauss Legendre 3 point ERROR: 0.000371
```

BONUS Adaptive Simpson

```
% Adaptive Quadrature using Simpson's 1/3 Multi step method
tol = 1e-5;

function [simp_1_3_integral, h] = Simposon1_3 (f, a, b)
    M = 200;
    n = 2 * M; % n = 2M
    k_val = linspace(a, b, n/2);
    h = (b-a)/M;
    sum = 0;

    for k = 1:n/2
        if mod(k, 2) == 0
            sum = sum + (2 * f(k_val(k)));
        else
            sum = sum + (4 * f(k_val(k)));
        end
    end

    simp_1_3_integral = (h/3) * (f(a) + sum + f(b));
end

global operations hs
operations = 0;
hs = [];

function I = AdaptiveSimpson(f, a, b, tol)
    global operations hs

    operations = operations + 1;
    c = (a+b)/2;
    [S1, ~] = Simposon1_3(f, a, c);
    [S2, ~] = Simposon1_3(f, c, b);

    [I1, h] = Simposon1_3(f, a, b);
    I2 = S1 + S2;
```

```
    hs(end+1) = h;

    if abs(I1 - I2) < 15 * tol
        I = I2 + (I2 - I1)/15;

    else
        I = AdaptiveSimpson(f, a, c, tol/2) + ...
            AdaptiveSimpson(f, c, b, tol/2);
    end
end

A_Simposn = AdaptiveSimpson(f, a, b, tol);
fprintf("Adaptive Quadrature using Simpson: %f, number of operations:
%d\n\n", A_Simposn, operations)
fprintf("h that were changed in adaptive simpson:\n")
disp(table(hs', 'VariableNames', {'h'}))
```

```
Adaptive Quadrature using Simpson: 0.745618, number of operations: 11

h that were changed in adaptive simpson:
       h
    _______

      0.01
     0.005
    0.0025
   0.00125
   0.00125
    0.0025
   0.00125
   0.00125
     0.005
    0.0025
    0.0025
```

Published with MATLAB® R2025a