

UWTSD

MSc

Task 1

Alternative Fuels Combustion Modelling

ACHA7002

Koa Brook

2203480

07/12/2025

Contents

1	Plotting and differentiation.....	1
2	Trapezoidal Method.....	2
3	Composite	2
4	Simpsons.....	3
5	Integration results tables.....	4
5.1	Composite convergence iterations	4
5.2	Simpsons Convergence iterations.....	4
6	Appendices.....	4
6.1	Composite convergence iterations table (full)	4
6.2	Your code script and functions.....	13
6.2.1	Lab 1 (main MATLAB script)	13
6.2.2	Functions	18

1 Plotting and differentiation

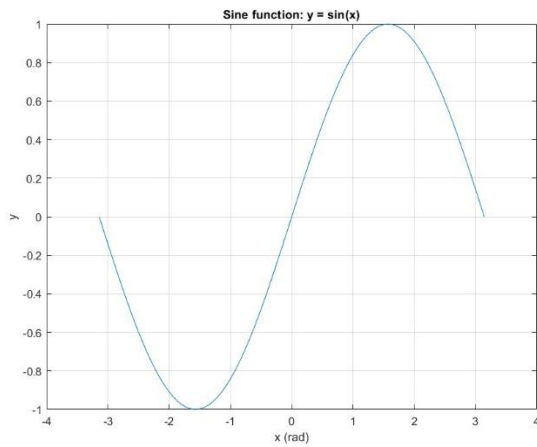


Figure 1 $y = \sin(x)$ plot

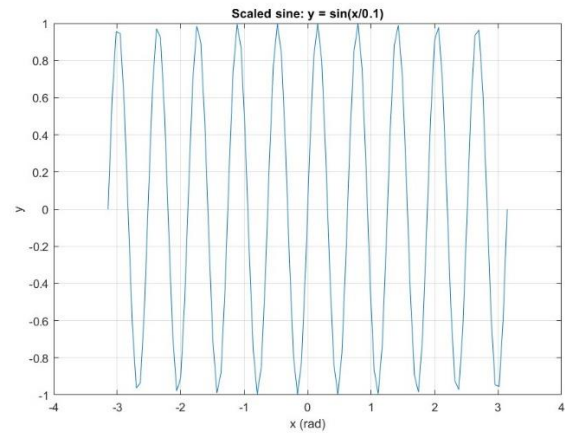


Figure 2 $y = \sin_scale(x, sf)$ plot (0.1 steps)

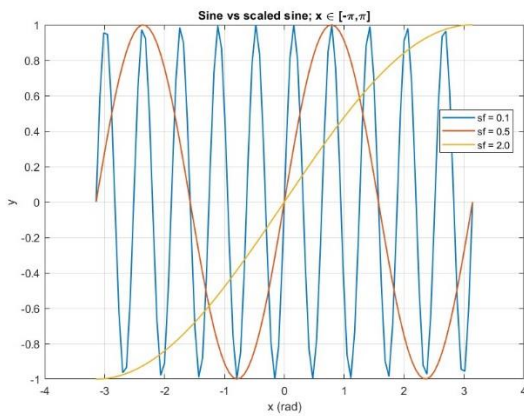


Figure 3 $y = \sin_scale(x, sf)$ plot ([0.1 0.5 2] steps)

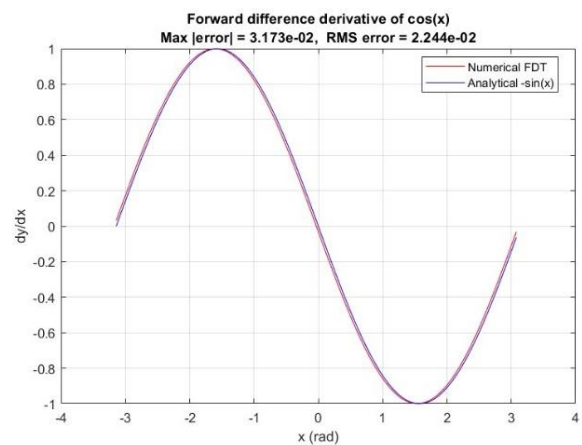


Figure 4 diff cos (x) to $-\sin(x)$

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_i)}{h}$$

Equation 1-1 numerical differentiation

2 Trapezoidal Method

$$I \approx \frac{1}{2}(f_a + f_b)(b - a)$$

Equation 2-1 trapezoidal method

A function was created to integrate 'x sin(x)' between 0 and pi/2 the code is in appendix

$$I = \int_0^{\pi/2} x \sin(x) dx ;$$

n	IntTrap (your code result)	Analytical (result)	percentage_error_Trap
5	1.0083	1	0.82654
50	1.0001	1	0.0082251
100	1	1	0.0020562

3 Composite

A function was created to integrate further functions in a discretised way from equation

$$I \approx \frac{1}{2} \Delta x_n (f_a + 2 \sum_{i=1}^{n-1} f_i + f_b)$$

Equation 3-1 composite

Table 3-1 results for integration of $f(x) = 1/x$ using composite method a while loop was used to find when the result was within 1e-6 (5dp)

	Analytical	Composite Trapezium			Converge 1e-6
'Result'	0.69315	0.69563	0.69317	0.69315	0.69315
'n iterations'	Na	5	50	100	250
'percentage difference'	Na	0.35891	0.0036066	0.00090167	0.00014427

4 Simpsons

A function was created to integrate further functions in a discretised separating odd and even crossing over in a quadratic manner way from equation 4.1

$$I \approx \frac{1}{3} \Delta x_n (f_a + 4 \sum_{i=1, \text{ odd only}}^{n-1} f_i + 2 \sum_{i=1, \text{ even only}}^{n-1} f_i + f_b)$$

Equation 4-1 Simpsons

Table 4-1 results for integration of $f(x) = 1/x$ using Simpson's method a while loop was used to find when the result was within $1e-6$ of the analytical

	Analytical	Simpsons Trapezium			Converge 1e-6
'Result'	0.69315	0.69317	0.69315	0.69315	0.69315
'n iterations'	NA	5	50	100	14
'percentage difference'	NA	0.0032623	7.2063e-07	4.5073e-08	0.00011589

5 Integration results tables

5.1 Composite convergence iterations

ITERATION K	N	ESTIMATE	ABS ERROR
1	1	0.75000	5.68530e-02
2	2	0.70833	1.51860e-02
3	3	0.70000	6.85280e-03
4	4	0.69702	3.87660e-03
5	5	0.69563	2.48770e-03
...	...	...	...
245	245	0.69315	1.0412e-06
246	246	0.69315	1.0328e-06
247	247	0.69315	1.0244e-06
248	248	0.69315	1.0162e-06
249	249	0.69315	1.0080e-06
250	250	0.69315	1.0000e-06 <i>(stopping point)</i>

5.2 Simpsons Convergence iterations

ITERATION K	N	ESTIMATE	ABS ERROR
2	2	0.69444	0.0012973
4	4	0.69325	0.00010679
6	6	0.69317	2.2613e -05
8	8	0.69315	7.3501e -06
10	10	0.69315	3.0501e -06
12	12	0.69315	1.4816e -06
14	14	0.69315	8.0332e -07

6 Appendices

6.1 Composite convergence iterations table (full)

ITERATION K	N	ESTIMATE	ABS ERROR
1	1	0.75000	5.68530e-02

2	2	0.70833	1.51860e-02
3	3	0.70000	6.85280e-03
4	4	0.69702	3.87660e-03
5	5	0.69563	2.48770e-03
6	6	0.69488	1.73020e-03
7	7	0.69442	1.27230e-03
8	8	0.69412	9.74670e-04
9	9	0.69392	7.70420e-04
10	10	0.69377	6.24220e-04
11	11	0.69366	5.16000e-04
12	12	0.69358	4.33650e-04
13	13	0.69352	3.69550e-04
14	14	0.69347	3.18670e-04
15	15	0.69342	2.77620e-04
16	16	0.69339	2.44020e-04
17	17	0.69336	2.16170e-04
18	18	0.69334	1.92830e-04
19	19	0.69332	1.73070e-04
20	20	0.69330	1.56200e-04
21	21	0.69329	1.41680e-04
22	22	0.69328	1.29100e-04
23	23	0.69327	1.18120e-04
24	24	0.69326	1.08480e-04
25	25	0.69325	9.99800e-05
26	26	0.69324	9.24390e-05
27	27	0.69323	8.57190e-05
28	28	0.69323	7.97070e-05
29	29	0.69322	7.43050e-05
30	30	0.69322	6.94350e-05
31	31	0.69321	6.50280e-05
32	32	0.69321	6.10280e-05

33	33	0.69320	5.73860e-05
34	34	0.69320	5.40600e-05
35	35	0.69320	5.10150e-05
36	36	0.69320	4.82210e-05
37	37	0.69319	4.56500e-05
38	38	0.69319	4.32790e-05
39	39	0.69319	4.10880e-05
40	40	0.69319	3.90590e-05
41	41	0.69318	3.71770e-05
42	42	0.69318	3.54280e-05
43	43	0.69318	3.38000e-05
44	44	0.69318	3.22810e-05
45	45	0.69318	3.08620e-05
46	46	0.69318	2.95350e-05
47	47	0.69318	2.82920e-05
48	48	0.69317	2.71250e-05
49	49	0.69317	2.60290e-05
50	50	0.69317	2.49990e-05
51	51	0.69317	2.4028e-05
52	52	0.69317	2.3113e-05
53	53	0.69317	2.2249e-05
54	54	0.69317	2.1433e-05
55	55	0.69317	2.0660e-05
56	56	0.69317	1.9929e-05
57	57	0.69317	1.9236e-05
58	58	0.69317	1.8578e-05
59	59	0.69317	1.7954e-05
60	60	0.69316	1.7361e-05
61	61	0.69316	1.6796e-05
62	62	0.69316	1.6259e-05
63	63	0.69316	1.5747e-05

64	64	0.69316	1.5258e-05
65	65	0.69316	1.4792e-05
66	66	0.69316	1.4348e-05
67	67	0.69316	1.3923e-05
68	68	0.69316	1.3516e-05
69	69	0.69316	1.3127e-05
70	70	0.69316	1.2755e-05
71	71	0.69316	1.2398e-05
72	72	0.69316	1.2056e-05
73	73	0.69316	1.1728e-05
74	74	0.69316	1.1413e-05
75	75	0.69316	1.1111e-05
76	76	0.69316	1.0820e-05
77	77	0.69316	1.0541e-05
78	78	0.69316	1.0273e-05
79	79	0.69316	1.0014e-05
80	80	0.69316	9.7654e-06
81	81	0.69316	9.5258e-06
82	82	0.69316	9.2949e-06
83	83	0.69316	9.0723e-06
84	84	0.69316	8.8576e-06
85	85	0.69316	8.6504e-06
86	86	0.69316	8.4504e-06
87	87	0.69316	8.2572e-06
88	88	0.69316	8.0706e-06
89	89	0.69316	7.8903e-06
90	90	0.69315	7.7159e-06
91	91	0.69315	7.5473e-06
92	92	0.69315	7.3841e-06
93	93	0.69315	7.2262e-06
94	94	0.69315	7.0732e-06

95	95	0.69315	6.9251e-06
96	96	0.69315	6.7816e-06
97	97	0.69315	6.6425e-06
98	98	0.69315	6.5076e-06
99	99	0.69315	6.3768e-06
100	100	0.69315	6.2499e-06
101	101	0.69315	6.1268e-06
102	102	0.69315	6.0072e-06
103	103	0.69315	5.8912e-06
104	104	0.69315	5.7784e-06
105	105	0.69315	5.6689e-06
106	106	0.69315	5.5624e-06
107	107	0.69315	5.4589e-06
108	108	0.69315	5.3583e-06
109	109	0.69315	5.2604e-06
110	110	0.69315	5.1652e-06
111	111	0.69315	5.0726e-06
112	112	0.69315	4.9824e-06
113	113	0.69315	4.8946e-06
114	114	0.69315	4.8091e-06
115	115	0.69315	4.7259e-06
116	116	0.69315	4.6447e-06
117	117	0.69315	4.5657e-06
118	118	0.69315	4.4886e-06
119	119	0.69315	4.4135e-06
120	120	0.69315	4.3402e-06
121	121	0.69315	4.2688e-06
122	122	0.69315	4.1991e-06
123	123	0.69315	4.1311e-06
124	124	0.69315	4.0647e-06
125	125	0.69315	4.0000e-06

126	126	0.69315	3.9367e-06
127	127	0.69315	3.8750e-06
128	128	0.69315	3.8147e-06
129	129	0.69315	3.7558e-06
130	130	0.69315	3.6982e-06
131	131	0.69315	3.6420e-06
132	132	0.69315	3.5870e-06
133	133	0.69315	3.5332e-06
134	134	0.69315	3.4807e-06
135	135	0.69315	3.4293e-06
136	136	0.69315	3.3791e-06
137	137	0.69315	3.3299e-06
138	138	0.69315	3.2819e-06
139	139	0.69315	3.2348e-06
140	140	0.69315	3.1888e-06
141	141	0.69315	3.1437e-06
142	142	0.69315	3.0996e-06
143	143	0.69315	3.0564e-06
144	144	0.69315	3.0141e-06
145	145	0.69315	2.9726e-06
146	146	0.69315	2.9321e-06
147	147	0.69315	2.8923e-06
148	148	0.69315	2.8533e-06
149	149	0.69315	2.8152e-06
150	150	0.69315	2.7778e-06
151	151	0.69315	2.7411e-06
152	152	0.69315	2.7051e-06
153	153	0.69315	2.6699e-06
154	154	0.69315	2.6353e-06
155	155	0.69315	2.6014e-06
156	156	0.69315	2.5682e-06

157	157	0.69315	2.5356e-06
158	158	0.69315	2.5036e-06
159	159	0.69315	2.4722e-06
160	160	0.69315	2.4414e-06
161	161	0.69315	2.4112e-06
162	162	0.69315	2.3815e-06
163	163	0.69315	2.3524e-06
164	164	0.69315	2.3238e-06
165	165	0.69315	2.2957e-06
166	166	0.69315	2.2681e-06
167	167	0.69315	2.2410e-06
168	168	0.69315	2.2144e-06
169	169	0.69315	2.1883e-06
170	170	0.69315	2.1626e-06
171	171	0.69315	2.1374e-06
172	172	0.69315	2.1126e-06
173	173	0.69315	2.0883e-06
174	174	0.69315	2.0643e-06
175	175	0.69315	2.0408e-06
176	176	0.69315	2.0177e-06
177	177	0.69315	1.9949e-06
178	178	0.69315	1.9726e-06
179	179	0.69315	1.9506e-06
180	180	0.69315	1.9290e-06
181	181	0.69315	1.9077e-06
182	182	0.69315	1.8868e-06
183	183	0.69315	1.8663e-06
184	184	0.69315	1.8460e-06
185	185	0.69315	1.8261e-06
186	186	0.69315	1.8066e-06
187	187	0.69315	1.7873e-06

188	188	0.69315	1.7683e-06
189	189	0.69315	1.7497e-06
190	190	0.69315	1.7313e-06
191	191	0.69315	1.7132e-06
192	192	0.69315	1.6954e-06
193	193	0.69315	1.6779e-06
194	194	0.69315	1.6606e-06
195	195	0.69315	1.6437e-06
196	196	0.69315	1.6269e-06
197	197	0.69315	1.6104e-06
198	198	0.69315	1.5942e-06
199	199	0.69315	1.5782e-06
200	200	0.69315	1.5625e-06
201	201	0.69315	1.5470e-06
202	202	0.69315	1.5317e-06
203	203	0.69315	1.5167e-06
204	204	0.69315	1.5018e-06
205	205	0.69315	1.4872e-06
206	206	0.69315	1.4728e-06
207	207	0.69315	1.4586e-06
208	208	0.69315	1.4446e-06
209	209	0.69315	1.4308e-06
210	210	0.69315	1.4172e-06
211	211	0.69315	1.4038e-06
212	212	0.69315	1.3906e-06
213	213	0.69315	1.3776e-06
214	214	0.69315	1.3647e-06
215	215	0.69315	1.3521e-06
216	216	0.69315	1.3396e-06
217	217	0.69315	1.3273e-06
218	218	0.69315	1.3151e-06

219	219	0.69315	1.3031e-06
220	220	0.69315	1.2913e-06
221	221	0.69315	1.2797e-06
222	222	0.69315	1.2682e-06
223	223	0.69315	1.2568e-06
224	224	0.69315	1.2456e-06
225	225	0.69315	1.2346e-06
226	226	0.69315	1.2237e-06
227	227	0.69315	1.2129e-06
228	228	0.69315	1.2023e-06
229	229	0.69315	1.1918e-06
230	230	0.69315	1.1815e-06
231	231	0.69315	1.1713e-06
232	232	0.69315	1.1612e-06
233	233	0.69315	1.1512e-06
234	234	0.69315	1.1414e-06
235	235	0.69315	1.1317e-06
236	236	0.69315	1.1222e-06
237	237	0.69315	1.1127e-06
238	238	0.69315	1.1034e-06
239	239	0.69315	1.0942e-06
240	240	0.69315	1.0851e-06
241	241	0.69315	1.0761e-06
242	242	0.69315	1.0672e-06
243	243	0.69315	1.0584e-06
244	244	0.69315	1.0498e-06
245	245	0.69315	1.0412e-06
246	246	0.69315	1.0328e-06
247	247	0.69315	1.0244e-06
248	248	0.69315	1.0162e-06
249	249	0.69315	1.0080e-06

250	250	0.69315	1.0000e-06 (stopping point)
-----	-----	---------	--------------------------------

6.2 Your code script and functions

6.2.1 Lab 1 (main MATLAB script)

```

%% Task 1
clear; clc;
format short g

%% SINE PLOTS
N = 100;
x = linspace(-pi, pi, N)';

y = sin(x);
figure(1);
plot(x, y); grid on;
title('Sine function: y = sin(x)');
xlabel('x (rad)'); ylabel('y');

sf = 0.1;
y = sin_scale(x, sf);
figure(2);
plot(x, y); grid on;
title(sprintf('Scaled sine: y = sin(x/%.3g)', sf));
xlabel('x (rad)'); ylabel('y');

sf = [0.1 0.5 2];
N_sf = numel(sf);
sin_array = zeros(N, N_sf);
for ndx = 1:N_sf
    sin_array(:, ndx) = sin_scale(x, sf(ndx));
end
figure(3);
plot(x, sin_array, 'LineWidth', 1.2); grid on
title('Sine vs scaled sine; x \in [-\pi, \pi]');
xlabel('x (rad)'); ylabel('y');
legendStrings = arrayfun(@(s) sprintf('sf = %.1f', s), sf, ...
    'UniformOutput', false);
legend(legendStrings, 'Location', 'best');

%% FORWARD DIFFERENCE
x = linspace(-pi, pi, N)';
y = cos(x);
dy_dx = fwd_diff_cos(x, y);

analytical = -sin(x(1:end-1));
err_FDT = dy_dx - analytical;
maxErr_FDT = max(abs(err_FDT));
rmsErr_FDT = sqrt(mean(err_FDT.^2));

figure(4);
plot(x(1:end-1), dy_dx, 'r'); hold on
plot(x(1:end-1), -sin(x(1:end-1)), 'b'); hold off; grid on
legend('Numerical FDT', 'Analytical -sin(x)', 'Location', 'best')

```

```

xlabel('x (rad)'); ylabel('dy/dx')
title(sprintf(['Forward difference derivative of cos(x)\n', ...
              'Max |error| = %.3e, RMS error = %.3e'], ...
        maxErr_FDT, rmsErr_FDT));

%% TRAPEZOID: SINGLE INTERVAL
I_trap1 = trap1('sin_mult', 0, pi/2);
I_true_xsinx = (sin(pi/2) - (pi/2)*cos(pi/2)) - (sin(0) - 0*cos(0)); % = 1

perErr_trap1 = (I_trap1 - I_true_xsinx) / I_true_xsinx * 100;
fprintf('trap1:  $\int_0^{\pi/2} x \sin x \, dx \rightarrow$  %% error = %.2f %%\n', perErr_trap1);

%% COMPOSITE TRAPEZOID (N=100)
I_trap2_xsinx = trap2('sin_mult', 0, pi/2, N);
perErr_trap2_xsinx = abs(I_trap2_xsinx - I_true_xsinx) / I_true_xsinx * 100;
fprintf('trap2 (N=100):  $\int_0^{\pi/2} x \sin x \, dx \rightarrow$  %% error = %.6f %%\n', ...
        perErr_trap2_xsinx);

%% Table for composite trapezoid on  $x \sin(x)$ 
n_list_xsinx = [5, 50, 100];
analytical_xsinx = I_true_xsinx;
TrapVals_xsinx = zeros(size(n_list_xsinx));
TrapErr_xsinx = zeros(size(n_list_xsinx));

for i = 1:length(n_list_xsinx)
    TrapVals_xsinx(i) = trap2('sin_mult', 0, pi/2, n_list_xsinx(i));
    TrapErr_xsinx(i) = abs(TrapVals_xsinx(i) - analytical_xsinx) ...
        / analytical_xsinx * 100;
end

T_xsinx = table(n_list_xsinx', TrapVals_xsinx', ...
                repmat(analytical_xsinx, length(n_list_xsinx), 1), ...
                TrapErr_xsinx', ...
                'VariableNames', ...
                {'n', 'IntTrap', 'Analytical', 'PercentError_Trap'});

disp(' ');
disp('Composite Trapezoid Table for  $\int_0^{\pi/2} x \sin(x) \, dx$ ');
disp(T_xsinx);

%% COMPOSITE TRAPEZOID on  $1/x$ 
I_true_ln2 = log(2);
n_list = [5, 50, 100];

I_comp = zeros(size(n_list));
perErr = zeros(size(n_list));
for k = 1:numel(n_list)
    I_comp(k) = trap2('one_over_x', 1, 2, n_list(k));
    perErr(k) = abs(I_comp(k) - I_true_ln2) / I_true_ln2 * 100;
end

Interval = {'true'; '5'; '50'; '100'};
Values = [I_true_ln2; I_comp(:)];
ErrorPct = [nan; perErr(:)];
Iterations = [nan; n_list(:)];
T_trap = table(Interval, Iterations, Values, ErrorPct);
disp(' ');
disp('Composite Trapezoid -  $\int_1^2 (1/x) \, dx$  vs  $\ln(2)$ ');

```

```

disp(T_trap);

tol = 1e-6;
n = 1;

k = 1;
k_trap = [];           % iteration counter
n_trap = [];           % n at each step
I_trap = [];           % integral estimate
err_trap = [];         % absolute error

while true
    I_est = trap2('one_over_x', 1, 2, n);
    err = abs(I_est - I_true_ln2);

    k_trap(end+1) = k;
    n_trap(end+1) = n;
    I_trap(end+1) = I_est;
    err_trap(end+1) = err;

    if err <= tol
        break
    end

    n = n + 1;
    k = k + 1;
end

TrapezoidConvergenceTable = table( ...
    k_trap(:), n_trap(:), I_trap(:), err_trap(:), ...
    'VariableNames', {'Iteration_k', 'n', 'Estimate', 'AbsError'});

disp(TrapezoidConvergenceTable)

iter_trap = k_trap(end) - 1;
n_trap_final = n_trap(end);
I_est_trap = I_trap(end);

fprintf('[Trapezoid] n = %d, iterations = %d, I ≈ %.10f, abs err = %.3g\n', ...
    n_trap_final, iter_trap, I_est_trap, err_trap(end));

Trap_ConvTable = table(k_trap', n_trap', I_trap', err_trap', ...
    'VariableNames', {'Iteration_k', 'n', 'IntegralEstimate', 'AbsError'});

n_trap_final = n;
err_trap_final = abs(I_est_trap - I_true_ln2);
perErr_conv_trap = err_trap_final / I_true_ln2 * 100;
fprintf('\n[Trapezoid] n = %d, iterations = %d, I ≈ %.10f, abs err = %.3g\n', ...
    ...
    n_trap_final, iter_trap, I_est_trap, err_trap_final);

figure(5);
semilogx(n_list, perErr, '-o'); grid on
xlabel('Number of intervals n');
ylabel('Percentage error (%)');
title(sprintf('Composite trapezoid convergence for  $\int_1^2 (1/x) dx$ \n', ...

```

```

        '6 dp achieved at n = %d, iterations = %d, |err| = %.3g'], ...
        n_trap_final, iter_trap, err_trap_final));

RowLabel      = {'Result'; 'n iterations'; 'percentage difference'};
Analytical_col = [I_true_ln2; NaN; NaN];
Trap5_col     = [I_comp(1); n_list(1); perErr(1)];
Trap50_col    = [I_comp(2); n_list(2); perErr(2)];
Trap100_col   = [I_comp(3); n_list(3); perErr(3)];
Converge_col   = [I_est_trap; n_trap_final; perErr_conv_trap];

Table31 = table(RowLabel, Analytical_col, Trap5_col, Trap50_col, ...
                Trap100_col, Converge_col, ...
                'VariableNames', ...
                {'Label', 'Analytical', 'Trap_n5', 'Trap_n50', 'Trap_n100', ...
                'Converge_1e6'});

disp(' ');
disp('Table 3-1 style data (copy into Word):');
disp(Table31);

%% COMPOSITE SIMPSON on 1/x

n_req = [5, 50, 100];
n_even = n_req + mod(n_req, 2); % ensures even: [6 50 100]

I_comp_S = zeros(size(n_even));
perErr_S = zeros(size(n_even));
for k = 1:numel(n_even)
    I_comp_S(k) = Simps('one_over_x', 1, 2, n_even(k));
    perErr_S(k) = abs(I_comp_S(k) - I_true_ln2) / I_true_ln2 * 100;
end

Interval = {'true'; '5 (→6)'; '50'; '100'};
Values    = [I_true_ln2; I_comp_S(:)];
ErrorPct   = [nan; perErr_S(:)];
Iterations = [nan; n_even(:)];
T_simp = table(Interval, Iterations, Values, ErrorPct);
disp(' ');
disp('Composite Simpson -  $\int_1^2 (1/x) dx$  vs  $\ln(2)$  (note: 5→6 because n must be even)');
disp(T_simp);

tol = 1e-6;
n = 2;

k_simp = [];
n_simp = [];
I_simp = [];
err_simp = [];

while true
    I_est = Simps('one_over_x', 1, 2, n);
    err = abs(I_est - I_true_ln2);

    k_simp(end+1) = n;
    n_simp(end+1) = n;
    I_simp(end+1) = I_est;
    err_simp(end+1) = err;

```

```

    if err <= tol
        break
    end

    n = n + 2;
end

SimpsonConvergenceTable = table( ...
    k_simp(:), n_simp(:), I_simp(:), err_simp(:), ...
    'VariableNames', {'Iteration_k', 'n', 'Estimate', 'AbsError'});

disp(SimpsonConvergenceTable)

iter_simp    = k_simp(end) - 1;
n_simp_final = n_simp(end);
I_est_simp   = I_simp(end);

fprintf('[Simpson]    n = %d, iterations = %d, I ≈ %.10f, abs err = %.3g\n', ...
        n_simp_final, iter_simp, I_est_simp, err_simp(end));

Simp_ConvTable = table(k_simp', n_simp', I_simp', err_simp', ...
    'VariableNames', {'Iteration_k', 'n', 'IntegralEstimate', 'AbsError'});

n_simp_final    = n;
err_simp_final  = abs(I_est_simp - I_true_ln2);
perErr_conv_simp = err_simp_final / I_true_ln2 * 100;
fprintf('[Simpson]    n = %d, iterations = %d, I ≈ %.10f, abs err = %.3g\n', ...
        n_simp_final, iter_simp, I_est_simp, err_simp_final);

figure(6);
semilogx(n_even, perErr_S, '-o'); grid on
xlabel('Number of intervals n (even)');
ylabel('Percentage error (%)');
title(sprintf(['Simpson convergence for  $\int_1^2 (1/x) dx$ \n', ...
    '6 dp achieved at n = %d, iterations = %d, |err| = %.3g'], ...
        n_simp_final, iter_simp, err_simp_final));

RowLabel      = {'Result'; 'n iterations'; 'percentage difference'};
Analytical_col = [I_true_ln2; NaN; NaN];
Simp5_col     = [I_comp_S(1); n_even(1); perErr_S(1)]; % corresponds to 5→6
Simp50_col    = [I_comp_S(2); n_even(2); perErr_S(2)];
Simp100_col   = [I_comp_S(3); n_even(3); perErr_S(3)];
Converge_col   = [I_est_simp; n_simp_final; perErr_conv_simp];

Table41 = table(RowLabel, Analytical_col, Simp5_col, Simp50_col, ...
    Simp100_col, Converge_col, ...
    'VariableNames', ...
    {'Label', 'Analytical', 'Simp_n5to6', 'Simp_n50', 'Simp_n100', ...
    'Converge_1e6'});

disp(' ');
disp('Table 4-1 style data (copy into Word):');
disp(Table41);

%% SUMMARY
fprintf('\n==== Summary ==== \n');
fprintf('trap1 (x sin x, [0,π/2])           : %% error = %.2f %% \n', perErr_trap1);

```

```

fprintf('trap2 N=100 (x sin x)           : %% error = %.6f %%\n',
perErr_trap2_xsinx);
fprintf('Trap2 (1/x, 6 dp)              : n = %d, iterations = %d\n', ...
        n_trap_final, iter_trap);
fprintf('Simpson (1/x, 6 dp)           : n = %d, iterations = %d\n', ...
        n_simp_final, iter_simp);

```

6.2.2 Functions

6.2.2.1 *One over x*

```

function y = one_over_x(x)
    y = 1 ./ x;
end

```

6.2.2.2 *xSin(x)*

```

function [y] = sin_mult(x)

y = x.*sin(x);

end

```

6.2.2.3 *sin(x/sf)*

```

function [y] = sin_scale(x,sf)

y = sin(x./sf);

end

```

6.2.2.4 *forward difference*

```

function [dy_dx] = fwd_diff_cos(x,y)

dy_dx = ( y(2:end) - y(1:end-1) ) ./ ( x(2:end) - x(1:end-1) );

end

```

6.2.2.5 *trap 1*

```

function int=trap1(fun,a,b)
%Input    - f is the integrand input as a string 'f'

if a>b ;
    error('b must be greater than a') ;
end ;

h=(b-a);
int = h*(feval(fun,a)+feval(fun,b))/2;
end

```

6.2.2.6 *trap 2*

```

function [int,h,x,y] = trap2(fun, a, b, n)

```

```
h = (b - a) / n;  
x = a:h:b;  
y = feval(fun, x);  
int = (h/2) * (y(1) + 2*sum(y(2:end-1)) + y(end));  
end
```

6.2.2.7 Simpson's

```
function I = Simps(fun, a, b, n)  
    if mod(n,2) ~= 0;  
        error('n must be even for Simpson''s rule');  
    end  
    h = (b - a) / n;  
    x = a:h:b;  
    y = feval(fun, x);  
  
    I = (h/3) * (y(1) + 4*sum(y(2:2:end-1)) + 2*sum(y(3:2:end-2)) + y(end));  
end
```