MONASH University

Office Use Only

Semester Two 2017
Examination Period

Faculty of Engineering

EXAM CODES: **ENG1060**
TITLE OF PAPER: **COMPUTING FOR ENGINEERS - PAPER 1**
EXAM DURATION: 3 hours writing time
READING TIME: 10 minutes

THIS PAPER IS FOR STUDENTS STUDYING AT:(tick where applicable)

☐ Berwick	☑ Clayton	☑ Malaysia	☐ Off Campus Learning	☐ Open Learning
☐ Caulfield	☐ Gippsland	☐ Peninsula	☐ Monash Extension	☐ Sth Africa
☐ Parkville	☐ Other (specify)			

During an exam, you must not have in your possession any item/material that has not been authorised for your exam. This includes books, notes, paper, electronic device/s, mobile phone, smart watch/device, calculator, pencil case, or writing on any part of your body. Any authorised items are listed below. Items/materials on your desk, chair, in your clothing or otherwise on your person will be deemed to be in your possession.

No examination materials are to be removed from the room.

This includes retaining, copying, memorising or noting down content of exam material for personal use or to share with any other person by any means following your exam.

Failure to comply with the above instructions, or attempting to cheat or cheating in an exam is a discipline offence under Part 7 of the Monash University (Council) Regulations.

<u>AUTHORISED MATERIALS</u>

OPEN BOOK	☐ **YES**	☑ **NO**
CALCULATORS	☑ **YES**	☐ **NO**

(Only calculators with an 'Approved for Use' Faculty sticker are permitted)

SPECIFICALLY PERMITTED ITEMS	☐ **YES**	☑ **NO**

if yes, items permitted are: NONE

Candidates must complete this section if required to write answers within this paper

STUDENT ID: _ _ _ _ _ _ _ _ DESK NUMBER: _ _ _ _ _

EXAM INSTRUCTIONS

- Complete the **Student ID** and **Desk Number** details on the cover page
- Write all **answers in the answer boxes**
- Write your answers with a pen
- **DO NOT** use a red pen or marker
- Blank sheets are provided at the back of the exam for workings. These workings will not be marked.
- You may detach the blank sheets and formula sheets from the back of the exam paper.

EXAM OUTLINE

PART A (40 MARKS)

Attempt ALL Questions

PART B (60 MARKS)

Attempt ALL Questions

Blank sheets for workings (not marked)

MATLAB Information and FORMULAS

Office Use Only

A1 /7	A2 /8	A3 /6	A4 /8	A5 /6	A6 /5	B1 /15	B2 /15	B3 /15	B4 /15	TOTAL

PART A: ATTEMPT **ALL** QUESTIONS

Question A1 (7 marks)

Consider the following matrices:

$$S = \begin{bmatrix} 8 & 9 & 3 & 9 \\ 0 & 6 & 5 & 2 \\ 4 & 1 & 11 & 9 \\ 7 & 6 & 2 & 16 \end{bmatrix}, \qquad T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \qquad U = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix}$$

Where S, T and U are double types.
Note: if an output returns an error, write down "error".

(a) Provide the syntax to create the **U vector**

U = [4; 5; 6] 0.5 marks
Or U = [4,5,6]' or U=transpose([4,5,6])

(b) Provide the syntax to extract the **3rd column of matrix S**

S(:, 3) 0.5 marks

(c) Provide the output of **transpose(T)**

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

0.5 marks

(d) Provide the output of **C = [T; U]**

Error: Dimensions of matrices being concatenated are not consistent. 0.5 marks
Only "error" is needed for full marks.

(e) Provide the output of **D = eye(size(S))**

1 mark

1	**0**	**0**	**0**
0	**1**	**0**	**0**
0	**0**	**1**	**0**
0	**0**	**0**	**1**

(f) Provide the outputs of **[F1, F2] = max(S)**

F1 = 8	**9**	**11**	**16**	**1 mark**
F2 = 1	**1**	**3**	**4**	

(g) Provide a **one-line syntax that modifies T** to the following matrix without addressing individual elements:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

T(3,:)=[1 1 1 1] or T(3,:)=1 **1 mark**

(h) Provide the **syntax to extract the 3x3 matrix** comprised of elements common from **the 1st, 3rd and 4th rows, and the 1st, 2nd and 4th columns of S**. You must not address individual elements and you must complete it in a single line.

S([1 3 4], [1 2 4]) **1 mark**

[S(1,[1 2 4]); S(3,[1 2 4]); S(4,[1 2 4])] **1 mark**

(i) Provide the output of **K=sum(S,2)**

K = **1 mark**

29
13
25
31

Question A2 (8 marks)

Answer the following short questions:
<u>Note: if an output returns an error, write down “error”.</u>

(a) Consider result = [1 5 7 9]. Provide the output of result = [result 7].

result = [1 5 7 9 7]	**0.5 marks**

(b) Consider **D = 4:3:571**. Provide the output of **B = D(91) – D(85)**

B=18	**0.5 marks**

(c) Provide the function that computes the **logarithmic value of X to the base 10**

log10(X) or log10	**0.5 marks**

(d) Provide the output of **imag(5 + 9i)**

9	**0.5 marks**

(e) Consider x, y and z to be vectors of the same size. State which of the following variables [A, B, C, D] contain unnecessary element-by-element operation(s):

A = x.*y +5*sin(z);
B = z./y - 7.*pi;
C = 10.^x + 4./y
D = exp(z./y) + 2/9 – cos(5.*x)

B, D	**0.5 marks for each (1 mark total)** **-0.5 marks for each wrong variable (A, C) – min total 0 marks**

(f) Describe the line specifications of the following function:

plot(k,m,'px')

Error in color/linetype argument **Only "error" is needed for full marks.**	**1 mark**

(g) Describe the **eps** built-in variable in MATLAB

eps is the smallest distance from one number to the next that MATLAB can recognise	**1 mark**

(h) Describe why the following line produces an **error**:

1st_line = 1:100;

Variable name cannot start with a numerical character	**1 mark**

(i) Create a vector that is **logarithmically spaced between 10^0 to 10^5 (inclusive) using 700 points.**

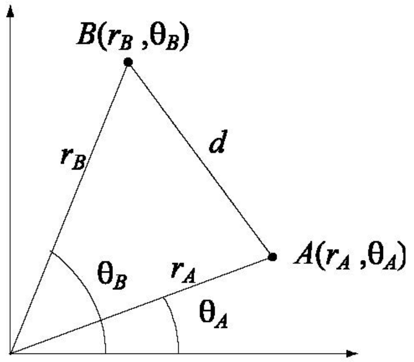
logspace(0,5,700)	**1 mark**
logspace(10^0, 10^5,700)	**0.5 marks**

(j) Describe the primary difference between **while loops** and **if statements**

If statements will only check the condition once whereas while loops will continue to check until the condition becomes false	**1 mark**

Question A3 (6 marks)

The distance between two points in polar coordinates can be calculated using the Law of Cosines equation $d = \sqrt{r_A^2 + r_B^2 - r_A r_B \cos(\theta_A - \theta_B)}$ where r_A, r_B, θ_A and θ_B are defined in the following figure:



(a) Provide MATLAB syntax to create a **function file** which calculates the distance (d) between two points in polar coordinates given the radii (r_A and r_B) and angles (θ_A and θ_B) of the points. You are not required to document the function and all inputs should be considered as scalars. **It should NOT work with vectors**. Use the following information for your function file:

- **Function name:** LOC
- **Input variables:** ra, rb, tha, thb
- **Output variables:** d

```
function d = LOC(ra,rb,tha,thb)                    0.5 marks
d = sqrt(ra^2 + rb^2 -ra*rb*cos(tha-thb));         0.5 marks

[still give marks if element-by-element operations are used]
```

(b) You are now working in a separate m-file. Provide MATLAB code to calculate the distance between **point A(3, π/3)** and **point B(4, 3π/4)**.

```
ra=3;
rb=4;
tha=pi/3;
thb=3*pi/4;

d = LOC(ra,rb,tha,thb);   or
d = sqrt(ra^2 + rb^2 -ra*rb*cos(tha-thb))
```

If all variables correct = 0.5 marks
(one π symbol is okay)
~~Function call~~ calculate d = 0.5 marks

Note: Any variable names can be used as long as the same

variables are called in the function

(c) Provide the syntax to print out the following statement using fprintf.

The distance between points A and B is <value>

where **<value>** represents the value contained in the variable **d printed in fixed point notation with a precision of 5 decimal places**.

fprintf ('The distance between points A and B is %.5f',d) Correct specifier and precision = 0.5 marks Correct fprintf statement = 0.5 marks

(d) The variable d is classified as "short" if its value is less than or equal to 5, and classified as "long" if it is greater than 5. Provide MATLAB code to assign the **strings 'short' or 'long' to the variable L** using if statements.

if d<=5 L = 'short'; or <var> = 'short' elseif d>5 L = 'long'; or <var> = 'long' end	Correct if-statement structure and conditions = 1 mark Correct string assignment = 0.5 marks

(e) **Assume that you are now provided row vectors for the variables ra, rb, tha, and thb.** These vectors are of length n. You are unable to provide these inputs to the function file you wrote in part (a) because they are vectors. Instead, provide the syntax to **compute the corresponding distances d, as a vector using a for loop**.

for i=1:n d(i) = LOC(ra(i),rb(i),tha(i),thb(i)); or d(i) = sqrt(ra(i)^2 + rb(i)^2 –ra(i)*rb(i)*cos(tha(i)-thb(i))) end	Correct for loop = 0.5 marks Correct body = 1 mark

Question A4 (8 marks)

Consider the following MATLAB function:

```
function [souls, HF] = DS3(nameless,king,ng)
king = round(king);

if ng < 5
    HF = nameless + ng - king;
else
    HF = nameless - ng + king;
end

souls = nameless.*(3/2) + ng./king;
Gael = sin(souls).^5;
end
```

Note: if an output returns an error, write down "error".

a) Provide the **name and extension format** of this function file

Name = DS3	0.5 marks
Format = .m	0.5 marks

b) Provide the output of **[souls, HF] = DS3(1, 3.5, 5)**

souls = 2.7500	0.5 marks
HF = 0	0.5 marks
[2.75, 0] is acceptable, although not strictly the output of MATLAB	1 mark

When "king" is not rounded (0.5 marks)

souls = 2.9286

HF = -0.5

c) Provide the output of **[in, valid] = DS3([1 1], [4.3 4.8], [4 6])**

in = [2.5000 2.7000]	1 mark
valid = [1 0]	1 mark

"king" not rounded with "ng" as vector (1.5 marks)

in = [2.4302, 2.75]
valid = [1.3, -0.2]

"king" is not rounded with "ng" treated individually (1.5 marks)
in = [2.4302, 2.75]
valid = [0.7, -0.2]

"king" is rounded with "ng" treated individually = same answer as red text (2 marks)
in = [2.5000, 2.7000]
valid = [1, 0]

2-by-2 matrix is not acceptable as it is not clear which values belong to which variable. 1 mark

d) Provide the output of **Y = DS3(1, 2, 3)**

Y = 3 1 mark

(f) Is it possible to turn the function provided at the start of this question into an **anonymous function**? If yes, provide the syntax for the anonymous function. If no, explain why it is not possible.

No 0.5 marks
Because there is more than one output 0.5 marks

(g) MATLAB provides a **single warning** for the function file. Identify and describe what the warning is.

Something along the lines of: "The value assigned to Gael might be unused" 1 mark

(h) Describe one difference between sub functions and nested functions in relation to the inputs of the parent function.

Sub-functions can only call the parent's inputs if they are declared in the function header	0.5 marks
Nested functions can always call the parent's inputs even if they aren't declared in the function header	0.5 marks

Question A5 (6 marks)

The figure below depicts a gear represented by an *x*,*y* curve and the location of its defects as represented by the asterisk markers. The equations for *x* and *y* are given as:

$$x = r\cos(t)$$

$$y = r\sin(t)$$

where

$$r = a + \frac{1}{b}\tanh(b\sin(nt))$$

with ***a*=1**, ***b*=10**, ***n*=9** and ***t* is a vector from 0 to 2π (inclusive) with 7000 points** in total The coordinates of the defects correspond **to the first (x,y) point and then every 600th (*x,y*) point thereafter,** i.e. defects occur at the 1st, 601st, 1201st, 1801st, … points.

Write MATLAB code in the following parts to successfully reproduce the figure.

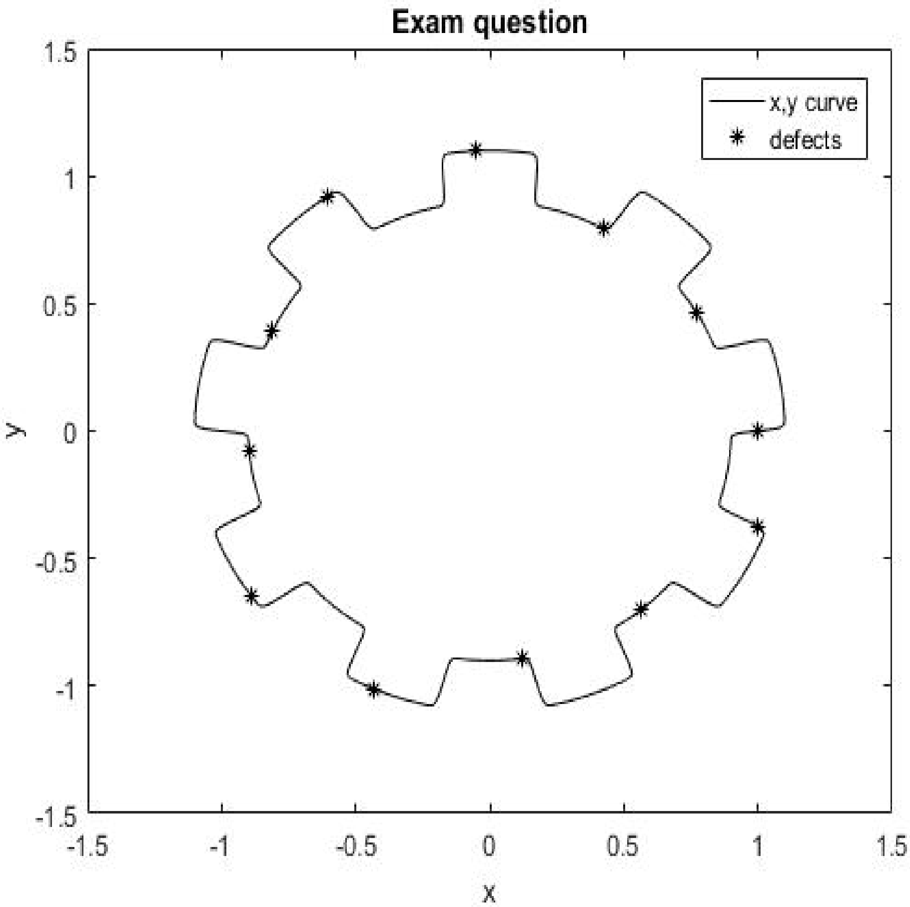


Figure 1: Gear with defect locations.

(a) This is the start of the m-file. Clear all variables, close all figure windows and clear the command window.

```
% start of m-file
Clear all; close all; clc;
```
(0.5 marks total)

(b) **Create all relevant variables for plotting.** Use element-by-element operators where appropriate.

```
% variable creation
a=1;
b=10;
n=9;

t = linspace(0,2*pi,7000);
r = a + 1/b*tanh(b*sin(n*t));
x = r.*cos(t);
y = r.*sin(t);
```
No marks for a, b and n

0.5 marks each for t, r, x and y

r, x and y can be function handles but must be called properly in subsequent parts

No marks if element-by-element operator used incorrectly.

(2 marks total)

(c) **Plot *y* against *x*** and label the plot accordingly. The line specification is a black continuous line.

```
% plotting y against x

plot(x,y,'k-')
xlabel('x')
ylabel('y')
title('Exam question')
```
0.5 marks for plot of y vs. x

0.5 marks for correct line specification

0.5 marks for labelling and title

(1.5 marks total)

(d) **Create variables which define the *x* and *y* coordinates of the defect locations.**

```
% defect locations
x_marker = x(1:600:end);
y_marker = y(1:600:end);
```
0.5 marks each

(1 mark total)

```
x_marker = x(1:600:7000)
y_marker = y(1:600:7000)
```

0.5 marks each
(1 mark total)

(e) **Plot the defect locations on the same figure produced in part c.** The line specification is a black asterisk marker. Ensure you include the legend.

```
% plotting the defect locations
hold on
plot(x_marker, y_marker, 'k*')
legend('x,y curve','defects')
hold off
```

0.5 marks each
(1 mark total)

Question A6 (5 marks)

Consider the following matrices:

$$\mathrm{X} = \begin{bmatrix} 70 & 4 & 69 & 3 \\ 3 & 9 & 31 & 43 \\ 27 & 82 & 95 & 38 \end{bmatrix}, \quad Y = \begin{bmatrix} 76 & 48 & 70 & 67 \\ 79 & 44 & 75 & 65 \\ 18 & 64 & 27 & 16 \end{bmatrix}, \quad Z = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

Where X, Y and Z are double types.
Note: if an output returns an error, write down "error".

a) Provide the output of **A = (X > 50)**

```
A =
    1    0    1    0
    0    0    0    0
    0    1    1    0
```

0.5 marks

b) Provide the output of **B = (X == 95) | (Y < 50)**

```
B =
    0    1    0    0
    0    1    0    0
    1    0    1    1
```

0.5 marks

c) Provide the output of **C = (Z == 1) & (X < Y)**

C = 1 mark

1	1	0	1
1	0	1	1
0	0	0	0

d) Provide the output of **[D1, D2] = find(Y==75)**

D1 = 2 0.5 marks

D2 = 3 0.5 marks

[2, 3] is acceptable, although not strictly the output of MATLAB 1 mark

e) Provide the output of **E = X(Z)**

Error: Subscript indices must either be real positive integers or logicals. 1 mark

Only "error" is needed for full marks.

f) Provide the output of **F = ~logical(-Z)**

F = 1 mark

0	0	1	0
0	1	0	0
1	0	0	0

PART B: ATTEMPT **ALL** QUESTIONS

Question B1 (15 marks)

Figure 2 shows a plot of the function $Y = 3\,X^5 - 12.5\,X^3 + 8\,X + 12$ over the range of $0 \le X \le 2$.

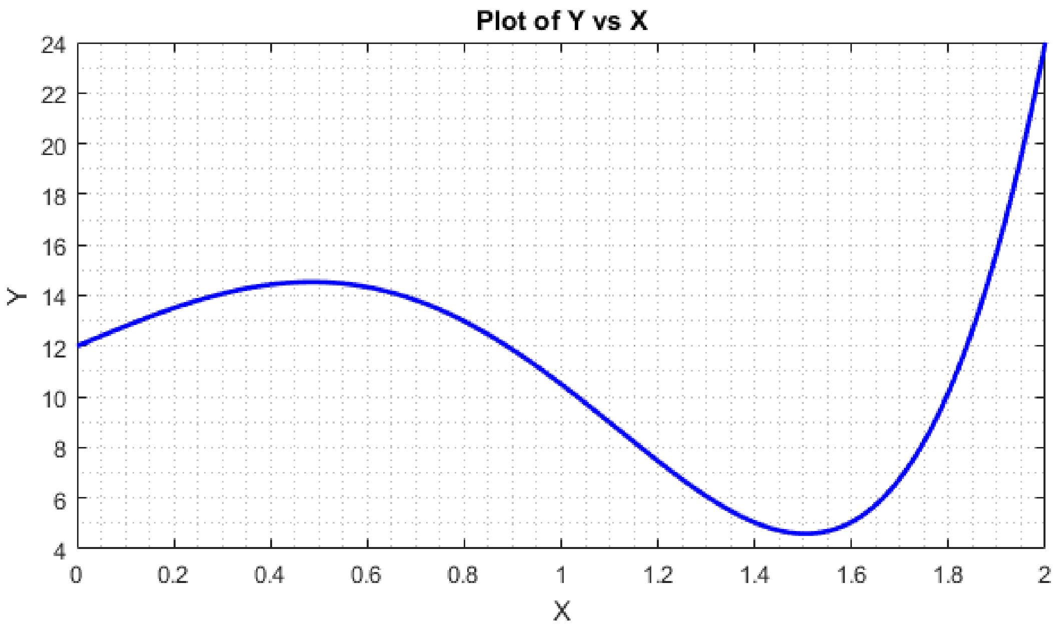


Figure 2

(a) Write the equation whose root must be found to determine the value of X which has a Y-value equal to 10.

$\boldsymbol{f(X) = 3\,X^5 - 12.5\,X^3 + 8\,X + 2}$

1 mark

(b) Use the method of **False Position** to calculate X at $Y = 10$. You should use a precision of **0.05** and employ initial guesses for the lower limit to be **0** and the upper limit to be **1.5**. Fill in the details of each iteration in the table below, provide 3 decimal places.

Note: You might need less rows, but should not need more. Workings for this part are ***not*** required)

Iteration number	Lower Limit X_l	Upper Limit X_u	Estimated Root X_r	$f(X_r)$
1	0.0000	1.5000	0.4051	4.4425

2	0.4051	1.5000	0.8990	1.8720
3	0.8990	1.5000	1.0535	-0.2952
4	0.8990	1.0535	1.0325	0.0215

(1 mark for each correct value of X_r) Total 4 marks

(c) Write an M-file that uses the **Newton-Raphson** method to locate the **local maximum** seen in Figure 2. Use a precision of 10^{-6} and determine an initial guess that will converge to the local maximum of the function (but NOT the local minimum seen Figure 2). Complete the M-file below by filling in the answer boxes with the missing code, making sure to follow the instructions in the comments.

% Define the anonymous functions to find the maximum of the function using the Newton-Raphson method

```
g = @(x) 15*x.^4 - 37.5*x.^2 + 8;
                                                    1 mark

dg = @(x) 60*x.^3 - 75*x;
                                                    1 mark
```

% Define initial guess and precision

```
xi = 0.9 ; %any value between 0.12 - 0.98          0.5 marks
precision = 1.0e-6  (or e-6 or 0.000001);          0.5 marks
```

% Calculate initial values for the functions

```
gxi = g(xi);                                        0.5 marks
dgxi = dg(xi);                                      0.5 marks
```

% Jump start the while loop

```
gxr = 1; %or any value greater than precision                0.5 marks
```

% Iteration for Newton-Raphson method starts

```
while abs(gxr) > precision                                   0.5 marks

   xr = xi - gxi/dgxi;                                       0.5 marks


   gxr = g(xr);                                              0.5 marks


   xi = xr;                                                  0.5 marks

   gxi = g(xi);                                              0.5 marks

   dgxi = dg(xi);                                             for both


end
```

% return root value

```
root = xr;                                                   0.5 marks
```

% Print root to 4 decimal places

```
fprintf('%.4f', root)                                        0.5 marks
```

(d) The three open root finding methods (Newton-Raphson, Secant and Modified Secant) share a similarity in concept. Given a current guess for the root, x_i, briefly describe in words the underlying idea used to find the next estimate x_{i+1}. Clearly state the way in which each method is different (**ZERO** marks for just writing down the formulae).

In the open methods, the idea is to follow the tangent to the curve at x_i and where that crosses the x-axis, that is the new estimate. 0.5 Marks

The methods differ in the way the derivative (slope) is estimated. N-R needs an analytic derivative, the Secant method approximates the derivative with the line that passes through the points of the last 2 guesses (x_{i-1}, x_i) and the modified secant draws the line passing through the function values at x_i, and x_i, + delta. 0.5 marks for each, 1.5 marks total

TOTAL question worth 2 marks

Question B2 (15 marks)

The net force (N) acting on an object can be represented by the following function:

$$f(s) = 5.26e^{0.04s}$$

where s is the displacement of the object measured in metres (m).

The work done by the net force exerted on the object can be determined by integrating the function over the distance travelled between two points (a and b) along the x-axis.

$$W = \int_a^b f(s)\ \mathrm{d}s$$

(a) Use the **Composite Trapezoidal** rule with 4 segments to calculate the total work done by the force exerted on the object to travel a distance of 5 m from the starting point (0 m). Show ALL your working and provide answers to 3 decimal places.

$h = \frac{5}{4} = 1.25$ 1 mark

i	1	2	3	4	5
s	0.0000	1.2500	2.5000	3.7500	5.0000
$f(s)$	5.2600	5.5297	5.8132	6.1112	6.4246

1 marks

$I = \frac{1.25}{2}[5.26 + 2 \times (5.5297 + 5.8132 + 6.1112) + 6.4246]$ 1 marks

I = 29.1205 J 1 mark

(b) Use the **Composite Simpson's 1/3** rule with 4 segments to calculate the total work done by the force exerted on the object to travel a distance of 5 m from the starting point (0 m). Show your working and provide answers to 3 decimal places.

$h = \frac{5}{4} = 1.25$

$I = \frac{1.25}{3}[5.26 + 4 \times (5.5297 + 6.1112) + 2 \times 5.8132 + 6.4246]$ 1 marks

I = 29.1144 J 1 mark

(c) Without calculating the analytical solution, which of your answers from part (a) and (b) do you think would be closer to the real solution. Briefly explain your reasoning.

More accurate answer: Simpson's 1/3 Rule 1 Mark

Reasoning: Based on truncation error, the Simpson's 1/3 rule 4^{th}-order accurate whereas the trapezoid rule error is 2^{nd} order. Generally for non-linear functions, Simpson's rules will be better than the trapezoid rule for the same step size.

1 Mark

(d) Provide one method of improving the integral estimates when using the Composite Simpson's 1/3 rule and say why it would improve the estimate.

A smaller step size i.e. more segments can improve the integral estimates. 1 Mark

This is because the truncation error (per step) scales like h^5 and smaller steps result in substantially less error (even if there are more steps)

1 Mark

(e) The local and global truncation error in the composite Simpson's 1/3 rule scale like

$$E_{Local} \text{ » } h^5 \frac{d^4 f}{dx^4} \qquad\qquad E_{Global} \text{ » } h^4 \frac{d^4 f}{dx^4}$$

If you wanted to decrease the error in your answer in part (b) by a factor of 4, what step size would you need to take? Provide brief reasoning for your answer.

h = h/sqrt(2) 1 Mark

To reduce error by a factor of 4, the global error needs to be smaller. The derivative term is approx. the same regardless of h, thus we need h^4 to be 4 times smaller, or h to be sqrt(2) times smaller 1 Mark

(f) Consider the function plotted in the following figure:

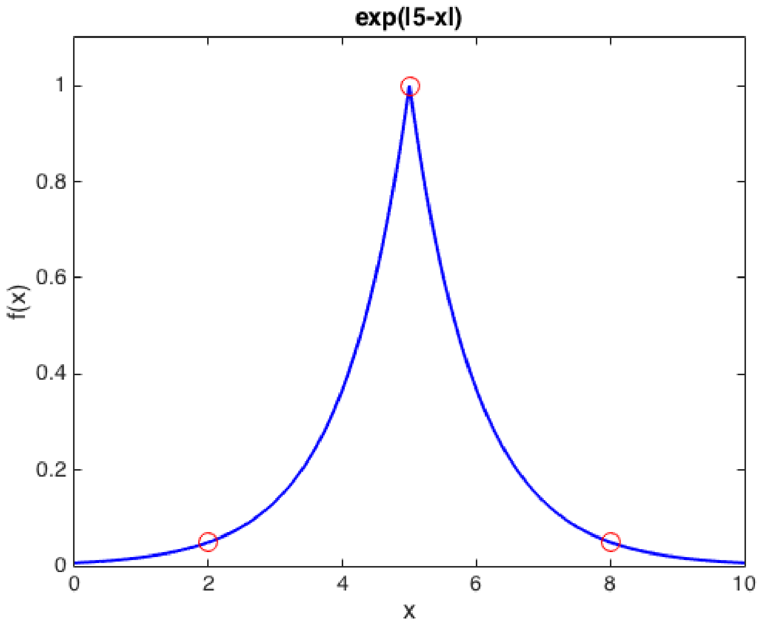


Figure 3

(i) Consider the 2-segment Composite Trapezoidal rule AND a single application of Simpson's 1/3 rule to calculate its integral from x=2 to 8. Which would likely give a better answer for the integral?

Answer: Trapezoidal rule 1 Mark

(ii) Next consider using 4 segments and both the Composite Trapezoidal and Composite Simpson's 1/3 rule. Which would likely give a better answer in this case?

Answer: Comp Simpson's rule 1 Mark

(iii) Do your answers for parts (i) and (ii) seem contradictory and are they consistent with your response to part (c)? If so, explain what is happening. If they don't explain why you expect these answers.

Answer: They seem contradictory on face value. However, what is happening is that the step size being taken with 2 segments is insufficient to pick up the key features of the function. Purely by luck the Comp Trap rule with 2 segments is able to pick up the shape of the function better than the single parabola of Simpson's rule. As soon as the step size decreases to 4 segments, Simpson's rule is able to fit more representative function segments to the points and thus will pick up the behavior better 1 Mark

Question B3 (15 marks)

The concentration of harmful bacteria in a water collection basin was measured after a storm and resulted in the following data:

t (hours)	3	6	9	12
c (CFU/100ml)	1230	890	670	490

The time t is measured in hours after the end of the storm. The concentration of bacteria c is measured in CFU per 100ml, and is observed to decay approximately **exponentially**. CFU stands for "Colony Forming Unit", which measures the number of living bacteria.

(a) You are required to perform curve fitting on the above data set, to estimate a suitable model that would represent bacterial growth in the water collection basin. From your observation of the data in the table, which of the following is the best functional form you would choose? Provide a reason:

Put a tick against the best model fit:

(i) A positive exponential model $c = a\,e^{bt}$ ☐

(ii) A negative exponential model $c = a\,e^{-bt}$ ☐

(iii) Either model ☒ X   1 mark

Provide the reason for your answer

- The positive model is the same as the negative model with an opposite sign for beta, so either will give the same fit   1 mark
- (ZERO marks for either of the other choices)

(b) Assume that model (i) (the positive exponential fit) is the fit you want to use. Linearise this non-linear model (Show your working and STATE CLEARLY in words what you are doing – **ZERO** marks for just writing the answer).

Take the log of both sides of (1 Mark)

$$c = a e^{bt}$$

to give

$$\ln c = \ln\left(a e^{bt}\right)$$

$$= \ln a + bt$$

(1 Mark)

Write the linear equation that needs to be solved, identifying the correspondence between your equation above and the straight line of the form:

	y	=	a_0	+	a_1	x
Linearized model:	$\ln(c)$	=	$\ln(a)$	+	b	t
	(0.5 Mark)		(0.5 Mark)		(0.5 Mark)	(0.5 Mark)

(c) You will be required to fit a straight line to the linearized data using Least Squares Regression to obtain an equation of the form $\boldsymbol{y} = a_0 + a_1 \boldsymbol{x}$. Show the values you need to first calculate by filling in the table below (Do ***NOT*** show the arithmetic to calculate sums).

i	X_i	Y_i	$X_i Y_i$	X_i^2
1	3	7.115	21.344	9
2	6	6.791	40.747	36
3	9	6.507	58.565	81

4	12	6.194	74.333	144
SUM	30	26.608	194.99	270
MEAN	7.5	6.652		

2 marks for correct Sums (0.5 each)

(d) **ASSUME** you obtained the values in the table below (instead of the values you calculated above in part (c)) and then calculate the linear coefficients a_0 and a_1. SHOW YOUR WORKING.

i	X_i	Y_i	$X_i\, Y_i$	X_i^2
SUM	**30**	**26**	**190**	**270**
MEAN	**7.5**	**6.5**		

$$a_1 = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - \left(\sum x_i\right)^2}$$

$$a_1 = \frac{4 \times 190 - 30 \times 26}{4 \times 270 - 30^2} = -0.111$$

1 Mark

$$a_0 = \bar{y} - a_1 \bar{x}$$

$$a_0 = 6.5 + 0.111 \times 7.5 = 7.333$$

1 Mark

(e) From your results in part (d), calculate the non-linear coefficients (α and β). Finally, show the non-linear equation in the box as requested.

$$\ln(a) = a_0 \Rightarrow a = e^{a_0} = e^{7.333} = 1530.47$$

$$b = a_1 = -0.111$$

1 Mark for each correct value (2 marks TOTAL)

Equation of fitted curve:

$$c = 1530.47 \times \exp(-0.111t)$$

1 Mark

(f) When curve fitting using MATLAB, the syntax of the MATLAB built-in functions **polyfit** and **polyval** are

`P = polyfit(x,y,N)` finds the coefficients of a polynomial P(x) of degree N that fits the data y best in a least-squares sense. P is a row vector of length N+1 containing the polynomial coefficients in reverse order, i.e. the polynomial is written

`y(x) = P(1)*x^N + P(2)*x^(N-1) +...+ P(N)*x + P(N+1).`

`y = polyval(P,x)` returns the value of a polynomial P evaluated at x. Here, P is the same vector that is defined above (using polyfit).

For the raw data in the table at the top of this question, complete the M-file below to fit the population data ***without linearising***, by directly fitting a linear model using **polyfit().** Once you have the fit, plot the function using polyval and 50 evenly spaced values of **t** between 0 to 20 hours inclusive (do not worry about axis labels, etc.).

```
Time = [3 6 9 12];
Conc = [1230 890 670 490];


P = polyfit (Time, Conc, 1);                 0.5 Mark
tplot = linspace (0, 20 ,50);                0.5 Mark
cplot = polyval ( P, tplot );                0.5 Mark
plot (tplot, cplot)                          0.5 Mark
```

(2 Marks TOTAL)

Question B4 (15 marks)

(a) When considering numerical solution of the ODE $\frac{dy}{dt} = f(t, y)$, the midpoint method is called a Predictor-Corrector method. Given the solution y_i at time t_i, y_{i+1} is the solution at time t_{i+1} (= $t_i + h$) and is found using

$$y_{i+1/2} = y_i + f(t_i, y_i)\frac{h}{2}$$

$$t_{i+1/2} = t_i + \frac{h}{2}$$

$$y_{i+1} = y_i + f(t_{i+1/2}, y_{i+1/2})h$$

In words, describe what $f(t_i, y_i)$ and $f(t_{i+1/2}, y_{i+1/2})$ represent (**ZERO** marks for writing "the function values at ...").

$f(t_i, y_i)$ is

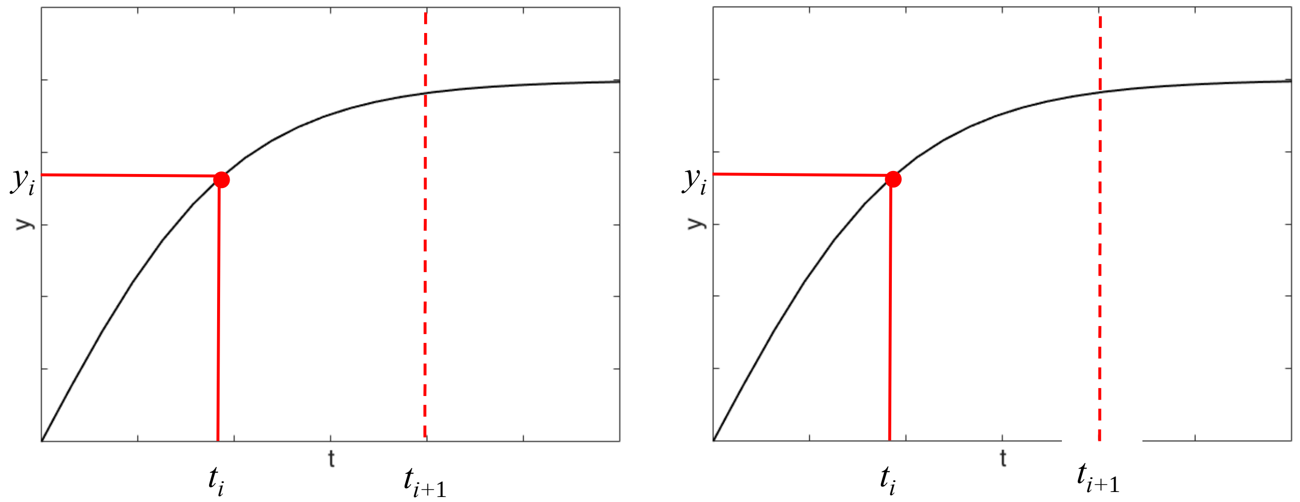
The slope of the solution evaluated at the start of the interval (t_i)

$f(t_{i+1/2}, y_{i+1/2})$ is

An estimate of the slope of the solution evaluated at the midpoint of the interval ($t_{i+1/2}$)

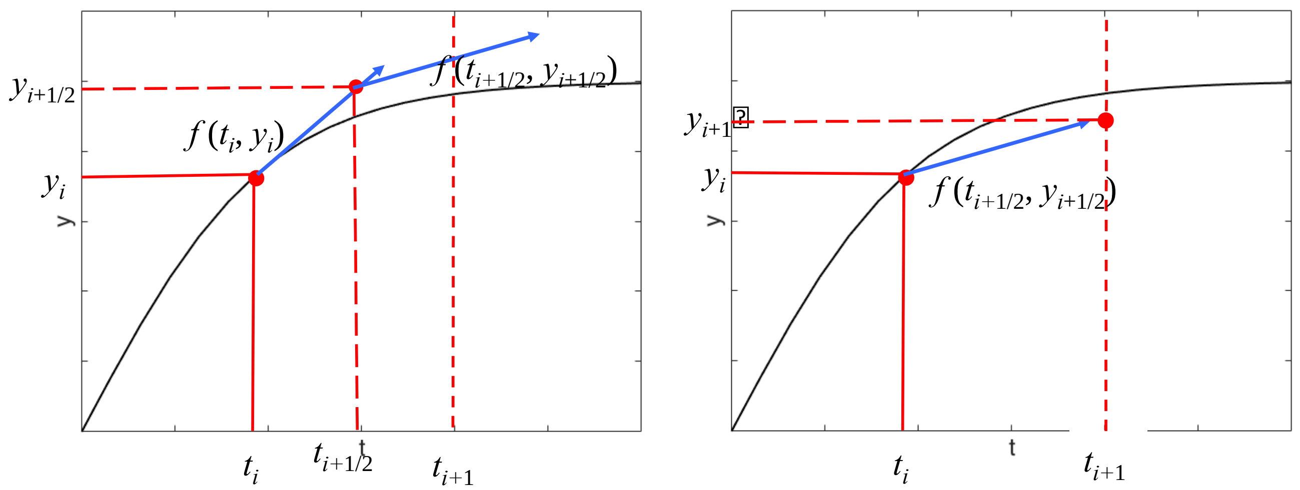
1 mark for both (0.5 each)

On the figure below, describe the predictor step and corrector step graphically, including the following features $f(t_i, y_i)$ $t_{i+1/2}$, $y_{i+1/2}$, $f(t_{i+1/2}, y_{i+1/2})$ and y_{i+1} (You could use arrows to represent the values of f and include one or two very brief comments for clarity if desired.)



Predictor step   Corrector step

ANSWER below. Arrows should represent the tangent vector at the 2 points. Some words could be used to describe that the midpoint slope $f\,(t_{i+1/2},\ y_{i+1/2})$ is used at $(t_i,\ y_i)$ to perform the corrector full step. 1 Mark for each figure (2 marks Total)



Predictor step   Corrector step

(b) Consider the following statement:

"Heun's method for solving an ODE is more accurate than the midpoint method."

In the answer box below state whether this statement is correct or not. Provide a brief written justification for your answer. You may wish to mention the order of accuracy of the methods or the differences between them when estimating the right hand side.

"Heun's method for solving an ODE is more accurate than the midpoint method."

Circle your answer

TRUE / FALSE 1 Mark

REASON:

Something like: Both Heun's and midpoint method have an order of accuracy in a single step of O(h^3), (and globally O(h^2)) thus they are similarly accurate. Even though the answers will differ, one is not more accurate than the other

(c) Consider the case where $f(t,y) = 1\text{-}y^2$ and thus the ODE is $\frac{dy}{dt} = 1 - y^2$.

Solve this initial value problem using ***Euler's method*** over the interval from $t = 0$ to 1.5 using a step size of h=0.5. The initial condition is $y(0) = 0$. Fill in the blank entries in the table below to 3 decimal places (if a cell is blacked out, you do not need to calculate it).

ERROR in f(t,y) in next table

The following table contains the answer **if** $f\,(t_0\ ,\ y_0)=0$ **was used** as given in the question

i	t_i	y_i	$f\,(t_i\ ,\ y_i)$
0	0	0	0
1	0.5	***0.0***	1.0
2	1.0	***0.5***	0.75
3	1.5	***0.875***	

The following table contains the answer **if** $f\,(t_0\ ,\ y_0)=1$ **was used** as is actually correct

i	t_i	y_i	$f\,(t_i\ ,\ y_i)$
0	0	0	1
1	0.5	***0.5***	0.75
2	1.0	***0.875***	0.234
3	1.5	***0.992***	

EITHER set of answers is acceptable

1 mark for each correct value of y_i **(3 marks total)**

(d) Use ***Heun's method*** to do the same integration, filling in the table below to 3 decimal places (NOTE: the first predictor step has been done for you).

i	t_i	y_i	$f(t_i, y_i)$	y^0_{i+1}	t_{i+1}	$f(t_{i+1}, y^0_{i+1})$
0	0	0	1.0	0.5	0.5	0.75
1	0.5	***0.438***	0.809	***0.842***	1.0	0.291
2	1.0	***0.713***	0.492	***0.959***	1.5	0.081
3	1.5	***0.856***				

The space below can be used for working – IT WILL NOT BE MARKED

1 mark for each value of y_i and y^0_{i+1} **(5 marks total)**

(e) The ODE in Part (c) has an analytic solution given by $y = \frac{e^{2t} - 1}{e^{2t} + 1}$

Calculate the percentage error in your predicted solutions from parts (c) and (d) at t=1.5 and write them in the box below (use 1 decimal point in the %). Is this what you expect? Why or why not? (NOTE: (%Error=(predicted value – actual value)÷(actual value) × 100%)

The analytic solution at t=1.5 is 0.90515

Euler's method gives a % error = (0.992-0.905)/0.905 × 100% = 9.6%

1 Mark for both errors to +/- 0.5%

Heun's method gives a % error = (0.856-0.905)/0.905 × 100% = -5.4%

Yes it is what I expect – Heun's method is 2nd order (i.e. more) accurate than Euler's method (that is 1st order).

1 Mark

END of EXAM

Blank page for workings (will not be marked)

Blank page for workings (will not be marked)

Blank pages for working (will not be marked)

MATLAB Information and Formulas

OPERATOR PRECEDENCE

<table>
<tr><td>1</td><td>()</td><td>Parentheses</td></tr>
<tr><td>2</td><td>.' '
.^ ^</td><td>Transpose, Matrix Transpose,
Power, Matrix Power</td></tr>
<tr><td>3</td><td>~</td><td>Logical Negation</td></tr>
<tr><td>4</td><td>.* *
./ /
.\ \</td><td>Multiplication, Matrix Multiplication,
Right Division, Matrix Right Division,
Left Division, Matrix Left Division</td></tr>
<tr><td>5</td><td>+
-</td><td>Addition
Subtraction</td></tr>
<tr><td>6</td><td>:</td><td>Colon Operator</td></tr>
<tr><td>7</td><td>< <=
> >=
== ~=</td><td>Less Than, Less Than Or Equal To,
Greater Than, Greater Than Or Equal To,
Equal To, Not Equal To</td></tr>
<tr><td>8</td><td>&</td><td>Element-wise AND</td></tr>
<tr><td>9</td><td>|</td><td>Element-wise OR</td></tr>
<tr><td>10</td><td>&&</td><td>Short-circuit AND</td></tr>
<tr><td>11</td><td>||</td><td>Short-circuit OR</td></tr>
</table>

fprintf SPECIFIER

%d	Integer
%f	Fixed-Point Notation
%e	Exponential Notation
%s	String of Characters
%c	Single Character
\t	Horizontal Tab
\n	New Line
%%	Percent Character
\''	Single Quote Mark
\\	Backslash
\b	Backspace

Fixed-Point Notation Syntax

%<field_width>.<precision>f

COLOR SPECIFIER

r	Red
g	Green
b	Blue
c	Cyan
m	Magenta
y	Yellow

LINE STYLE SPECIFIER

-	Solid Line
--	Dashed Line
:	Dotted Line
-.	Dash-dot Line

MARKER TYPE SPECIFIER

+	Plus Sign
o	Circle
*	Asterisk
.	Point
x	Cross
s	Square
d	Diamond
^	Triangle (Up)
v	Triangle (Down)

`k`	Black
`w`	White

`>`	Triangle (Right)
`<`	Triangle (Left)

Root Finding

Bisection Method

$$x_r = \frac{x_l + x_u}{2}$$

False Position Method

$$x_r = x_u - \frac{f(x_u)(x_l - x_u)}{f(x_l) - f(x_u)}$$

Newton-Raphson Method

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

Secant Method

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

Modified Secant Method

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

Alternative Modified Secant Method

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

Curve Fitting

Linear Regression:

$$y = a_o + a_1 x$$

$$a_1 = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - \left(\sum x_i\right)^2}$$

$$a_0 = \bar{y} - a_1 \bar{x}$$

Coefficient of Determination

$$r^2 = \frac{S_t - S_r}{S_t}$$

Linearizing Nonlinear Models

Nonlinear	Linearized
$y = \alpha_1 e^{\beta_1 x}$	$\ln y = \ln \alpha_1 + \beta_1 x$
$y = \alpha_2 x^{\beta_2}$	$\log y = \log \alpha_2 + \beta_2 \log x$
$y = \alpha_3 \frac{x}{\beta_3 + x}$	$\frac{1}{y} = \frac{1}{\alpha_3} + \frac{\beta_3}{\alpha_3}\frac{1}{x}$

Standard Deviation

$$S_t = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

$$s_y = \sqrt{\frac{S_t}{n-1}}$$

Standard Error of the Regression Estimate

$$S_r = \sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_i\right)^2$$

$$s_{y/x} = \sqrt{\frac{S_r}{n-2}}$$

Numerical Integration (*n* is the number of points)

Trapezoidal Rule:

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

$$E_t = -\frac{1}{12}f''(\xi)(b-a)^3$$

Composite Trapezoidal Rule

$$I = \frac{h}{2}\left[f(x_1) + 2\sum_{i=2}^{n-1} f(x_i) + f(x_n)\right]$$

$$\text{where} \qquad h = \frac{(b-a)}{n-1}$$

Composite Trapezoidal Rule with Unequal Segments

$$I = (x_2 - x_1)\frac{f(x_2)+f(x_1)}{2} + (x_3 - x_2)\frac{f(x_3)+f(x_2)}{2} + \cdots + (x_n - x_{n-1})\frac{f(x_{n-1})+f(x_n)}{2}$$

Simpson's 1/3 Rule

$$I = \frac{h}{3}[f(x_1) + 4f(x_2) + f(x_3)]$$

$$E_t = -\frac{1}{2880}f^{(4)}(\xi)(b-a)^5$$

Simpson's 3/8 Rule

$$I = \frac{3h}{8}[f(x_1) + 3f(x_2) + 3f(x_3) + f(x_4)]$$

$$E_t = -\frac{1}{6480}f^{(4)}(\xi)(b-a)^5$$

Composite Simpson's 1/3 Rule: $$I = \frac{h}{3}\left[f(x_1) + 4\sum_{\substack{i=2,4,6,\ldots\\ i,\ \text{even}}}^{n-1} f(x_i) + 2\sum_{\substack{j=3,5,7,\ldots\\ j,\ \text{odd}}}^{n-2} f(x_j) + f(x_n)\right]$$

ODE: Initial Value Problems

Euler's Method

$$y_{i+1} = y_i + f(t_i, y_i)h$$

Heun's Method

$$y_{i+1}^0 = y_i + f(t_i, y_i)h$$

$$y_{i+1} = y_i + \frac{f(t_i, y_i) + f(t_{i+1}, y_{i+1}^0)}{2}h$$

Midpoint Method

$$y_{i+1/2} = y_i + f(t_i, y_i)\frac{h}{2}$$

$$t_{i+1/2} = t_i + \frac{h}{2}$$

$$y_{i+1} = y_i + f(t_{i+1/2}, y_{i+1/2})h$$