Assignment 6

Group Number 32

Software used: **Matlab**

Sanit Prashant Bhatkar (173109003)
Omkar Anil Pawar (173106001)

a. dy/dx=log(x+y)

a. Forward Euler Method

Code	Output

clc
clearvars
fprintf('Forward Euler''s Method\n\n')
h=0.2;
x=0:h:5;
[n,n]=size(x);
y=zeros(1,n);
y(1)=2;
 	
for i=1:n-1
  y(i+1)=y(i)+h*(log(x(i)+y(i)));
end
Table= table([x]',[y]');
Table.Properties.VariableNames = {'x','y'}
	a. For h = 0.2
	
	Forward Euler's Method
	
	Table =
	
	  x y
	  ___ _____
	
	  0 2
	  0.2 2.1386
	  0.4 2.3085
	  0.6 2.5078
	  0.8 2.7346
	  1 2.9871
	  1.2 3.2637
	  1.4 3.5629
	  1.6 3.8833
	  1.8 4.2237
	  2 4.5828
	  2.2 4.9597
	  2.4 5.3534
	  2.6 5.763
	  2.8 6.1878
	  3 6.627
	  3.2 7.0799
	  3.4 7.5459
	  3.6 8.0245
	  3.8 8.5151
	  4 9.0173
	  4.2 9.5306
	  4.4 10.054
	  4.6 10.589
	  4.8 11.133
	  5 11.686

b. Backward Euler Method

Code	Output

clc
clearvars
fprintf('Backward Euler''s Method\n\n')
h=0.2;
x=0:h:5;
[n,n]=size(x);
y=zeros(1,n);
y(1)=2;
 	
for i=1:n-1
  y(i+1)=y(i)+h*(log(x(i)+y(i)));
end
 	
for i=1:n-1
  error=1;
  dum=y(i+1);
  while error > 10^(-6)
  y(i+1)=y(i)+h*(log(x(i+1)+dum));
  error=y(i+1)-dum;
  dum=y(i+1);
  end
 	
end
 	
Table= table([x]',[y]');
Table.Properties.VariableNames = {'x','y'}
	b. For h = 0.2
	
	Backward Euler's Method
	
	Table =
	
	x y
	___ _____
	
	0 2
	0.2 2.1728
	0.4 2.3771
	0.6 2.6104
	0.8 2.8704
	1 3.1553
	1.2 3.4633
	1.4 3.7927
	1.6 4.1423
	1.8 4.5107
	2 4.8969
	2.2 5.2999
	2.4 5.7188
	2.6 6.1526
	2.8 6.6008
	3 7.0626
	3.2 7.5373
	3.4 8.0244
	3.6 8.5235
	3.8 9.0339
	4 9.5552
	4.2 10.087
	4.4 10.629
	4.6 11.181
	4.8 11.742
	5 12.312

c. Trapezoidal Method

Code	Output

clc
clearvars
fprintf('Trapezoidal Method\n\n')
h=0.2;
x=0:h:5;
[n,n]=size(x);
y=zeros(1,n);
y(1)=2;
 	
for i=1:n-1
  y(i+1)=y(i)+h*(log(x(i)+y(i)));
end
 	
for i=1:n-1
  error=1;
  dum=y(i+1);
  while error > 10^(-6)
  y(i+1)=y(i)+0.5*h*(log(x(i+1)+dum)+log(x(i)+y(i)));
  error=y(i+1)-dum;
  dum=y(i+1);
  end
 	
end
 	
Table= table([x]',[y]');
Table.Properties.VariableNames = {'x','y'}
	c. For h = 0.2
	
	Trapezoidal Method
	
	Table =
	
	x y
	___ _____
	
	0 2
	0.2 2.155
	0.4 2.3415
	0.6 2.5573
	0.8 2.8004
	1 3.0688
	1.2 3.3609
	1.4 3.6751
	1.6 4.01
	1.8 4.3643
	2 4.7369
	2.2 5.1268
	2.4 5.5331
	2.6 5.9549
	2.8 6.3913
	3 6.8418
	3.2 7.3057
	3.4 7.7823
	3.6 8.2712
	3.8 8.7717
	4 9.2835
	4.2 9.8061
	4.4 10.339
	4.6 10.882
	4.8 11.435
	5 11.997

Part B

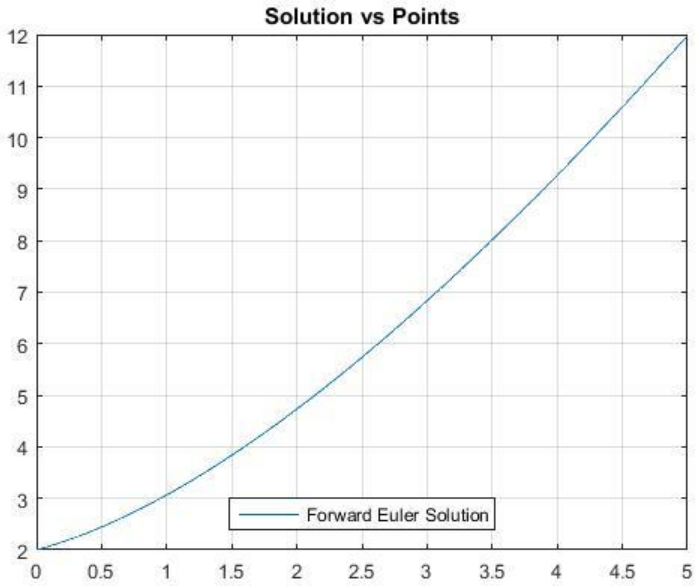
a. For n =0, the code should iterate till the solution converges. The convergence criterion can be chosen as 10^{-6}.
b. For n=1, it will be a simple predictor-corrector scheme.
c. For n =10, only 10 iterations will be computed.
d. Use the following values of h: (i) 10^{-4}, (ii) 10^{-3} and (iii) 10^{-2}.
e. Carry out solution till x = 5

a. $dy/dx=\log(x+y)$

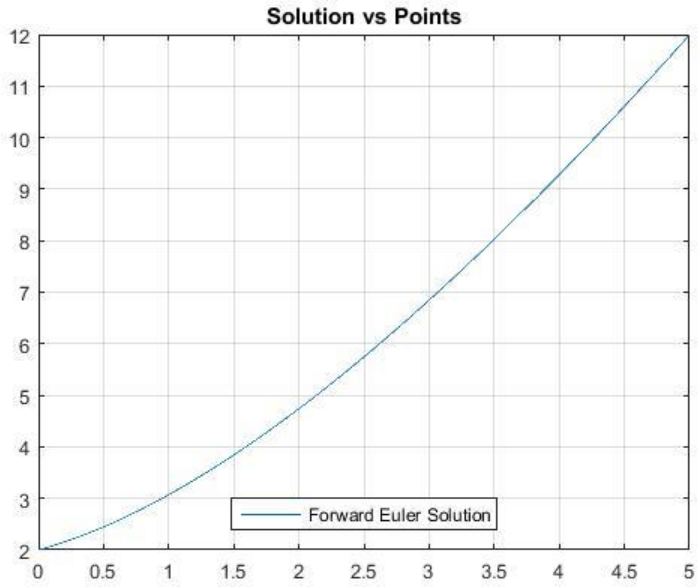
a. Forward Euler Method

Code	Output

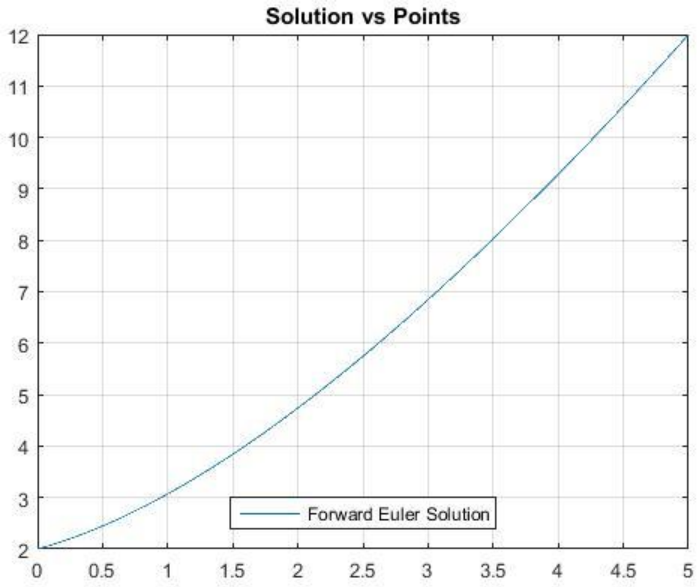
clc
clearvars
fprintf('Backward Euler''s Method\n\n')
%Interval value input
h=input('\nInterval ''h'': ');
x=0:h:5;
%Defining zero row of order n for y
[n,n]=size(x);
y=zeros(1,n);
y(1)=2;
%Forward Euler Formula
for i=1:n-1
  y(i+1)=y(i)+h*(log(x(i)+y(i)));
end
 	
fprintf('\ny at x= %d is %f\n\n',x(n),y(n));
plot(x,y)
title('Solution vs Points')
legend('Forward Euler Solution','Location','south')
grid on
	a. For h = 10^-2
	
	Backward Euler's Method
	
	Interval 'h': 10^-2
	
	y at x= 5 is 11.984346
	
	b. For h = 10^-3
	
	Backward Euler's Method
	
	Interval 'h': 10^-3
	
	y at x= 5 is 11.998442
	
	c. For h = 10^-4
	
	Backward Euler's Method
	
	Interval 'h': 10^-4
	
	y at x= 5 is 11.999852



h = 10^-2



h = 10^-3



h = 10^-4

b. Backward Euler Method

Code	Output

clc
clearvars
fprintf('Backward Euler''s Method\n\n')
%Input Iteration number
it=input('Number of iterations n: ');
h=input('\nInterval ''h'': ');
x=0:h:5;
%Defining zero row of order n for y
[n,n]=size(x);
ye=zeros(1,n);
ye(1)=2;
 	
%Forward Euler Formula for initial values
for i=1:n-1
ye(i+1)=ye(i)+h*(log(x(i)+ye(i)));
end
%For simplicity y(converging) is given variable 'yc'
%and y(iteration) is given variable 'yy'
yc=ye;
yy=ye;
 	
for i=1:n-1
%Backward Euler formula
%case 1: Full convergence
%Initial definition of error to pass while loop
error=1;
%dum stores the previous iteration value.
%Y(i+1)(j+1)=F(y(i),y(i+1)(j))
%here j represents iteration number
%i represents index of y
dum=yc(i+1);
while error > 10^(-6)
%Backward Euler formula
yc(i+1)=yc(i)+h*(log(x(i+1)+dum));
error=abs(yc(i+1)-dum);
dum=yc(i+1);
end
 	
if it==0
yy=yc;
 	
%else loop is written for predictor corrector scheme
%Loop performs 10 iterartions for input of 10
%For 1 iteration simple predictor corrector is obtained
else
for j=1:it
yy(i+1)=yy(i)+h*(log(x(i+1)+yy(i+1)));
end
end
 	
end
 	
if it==0
	a. For N = 0 & h = 10^-2
	
	Backward Euler's Method
	
	Number of iterations n: 0
	
	Interval 'h': 10^-2
	
	y at x= 5 is 12.015668
	
	residue_max =
	
	0
	
	b. For N = 0 & h = 10^-3
	
	Backward Euler's Method
	
	Number of iterations n: 0
	
	Interval 'h': 10^-3
	
	y at x= 5 is 12.001574
	
	residue_max =
	
	0
	
	c. For N = 0 & h = 10^-4
	
	Backward Euler's Method
	
	Number of iterations n: 0
	
	Interval 'h': 10^-4
	
	y at x= 5 is 12.000165
	
	residue_max =
	
	0

```
   fprintf('\ny at x= %d is %f\n\n',x(n),yc(n));
else
   fprintf('\ny at x= %d is %f\n\n',x(n),yy(n));
end
%Residue gives diffrence in converged value & iterated
value
residue=abs(yy-yc);
residue_max=max(residue)
%plot
figure
%Residue plot
subplot(2,1,1)
plot(x,residue,'-r')
title('Residue vs Points')
legend('Residue','Location','south')
grid on
%Solution plot
subplot(2,1,2)
plot(x,yc)
title('Solution vs Points')
legend('Backward Euler solution','Location','south')
grid on
```

d. For N = 1 & h = 10^-2

Backward Euler's Method

Number of iterations n: 1

Interval 'h': 10^-2

y at x= 5 is 12.005738

residue_max =

0.0099

e. For N = 1 & h = 10^-3

Backward Euler's Method

Number of iterations n: 1

Interval 'h': 10^-3

y at x= 5 is 12.000582

residue_max =

9.9199e-04

f. For N = 1 & h = 10^-4

Backward Euler's Method

Number of iterations n: 1

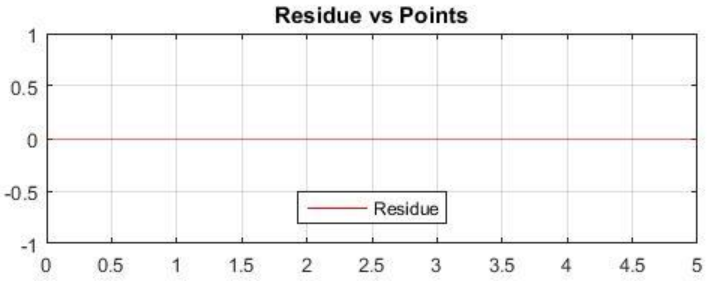
Interval 'h': 10^-4

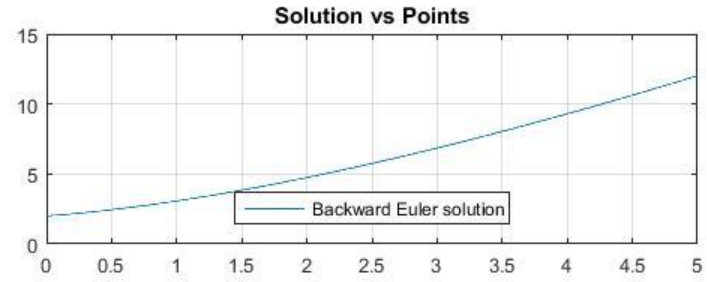
y at x= 5 is 12.000066

residue_max =

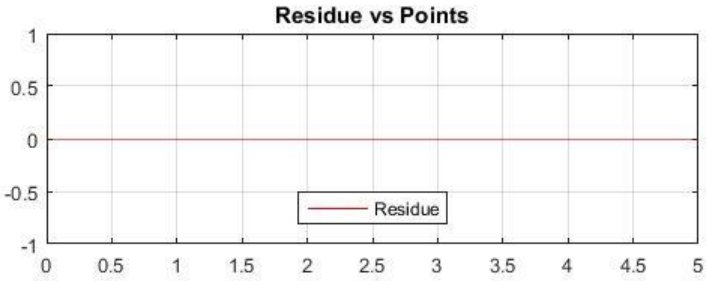
9.9203e-05

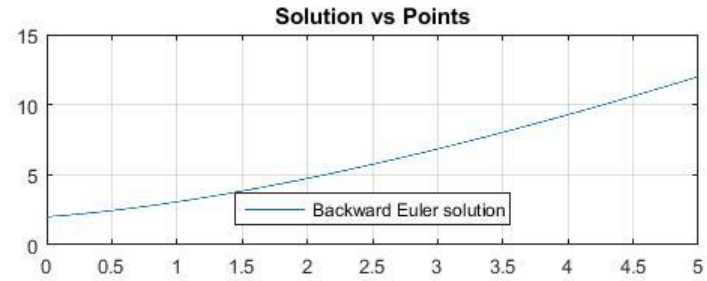
	g. For N = 10 & h = 10^-2 Backward Euler's Method Number of iterations n: 10 Interval 'h': 10^-2 y at x= 5 is 12.015668 residue_max = 5.8168e-08 **h. For N = 10 & h = 10^-3** Backward Euler's Method Number of iterations n: 10 Interval 'h': 10^-3 y at x= 5 is 12.001574 residue_max = 2.0793e-07 **i. For N = 10 & h = 10^-4** Backward Euler's Method Number of iterations n: 10 Interval 'h': 10^-4 y at x= 5 is 12.000165 residue_max = 8.7180e-09



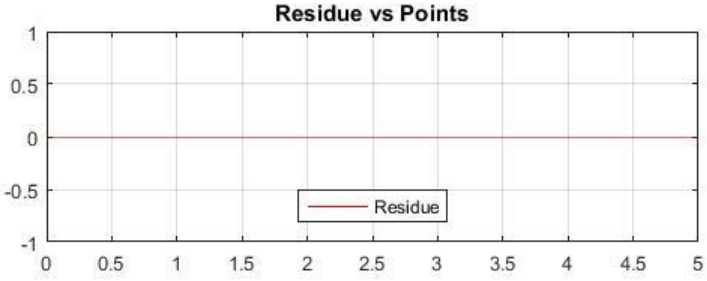


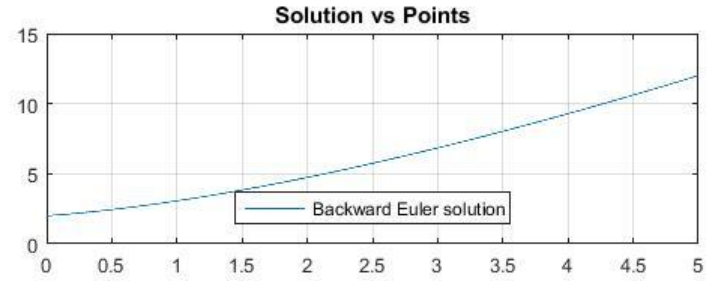
n = 0, h = 10^-2



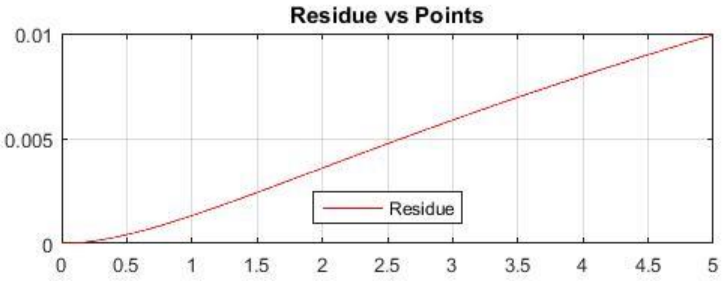


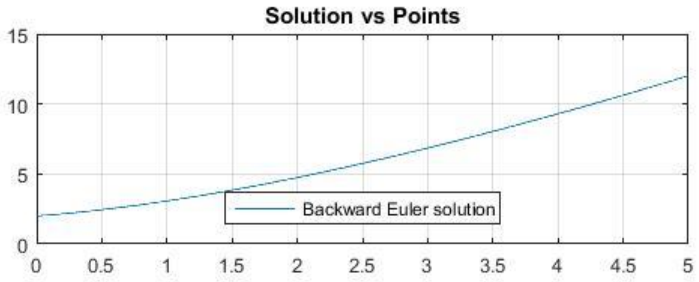
n = 0, h = 10^-3



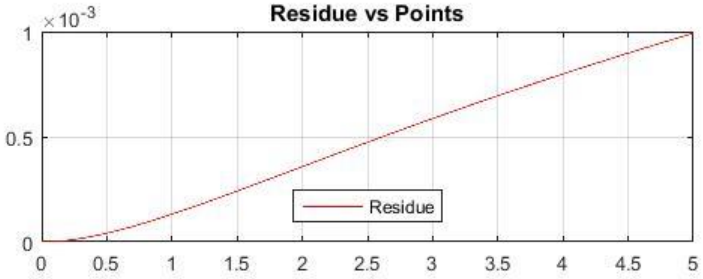


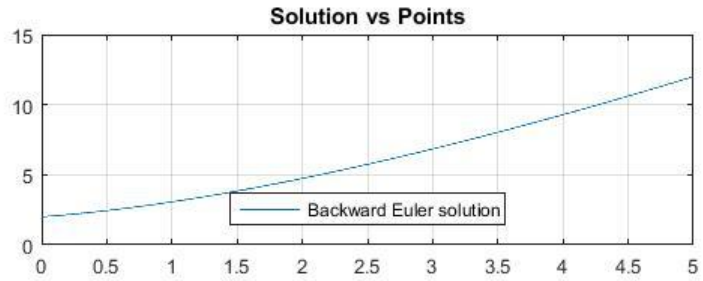
n = 0, h = 10^-4



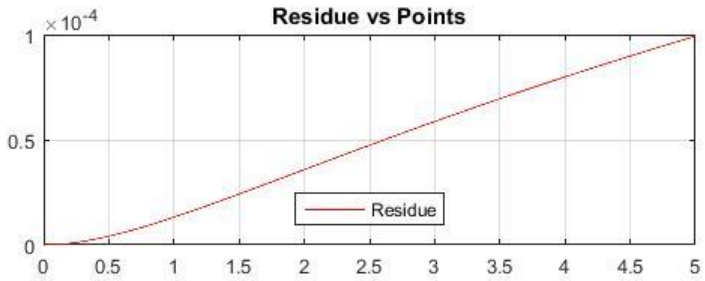


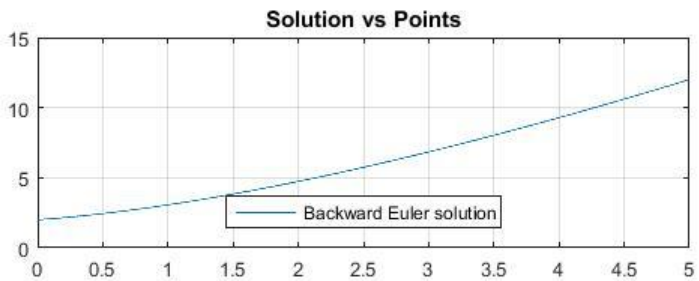
n = 1, h = 10^-2



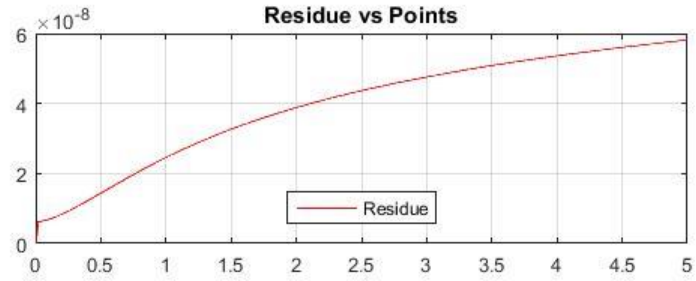


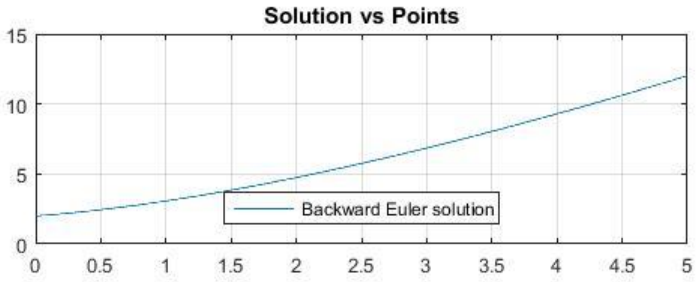
n = 1, h = 10^-3



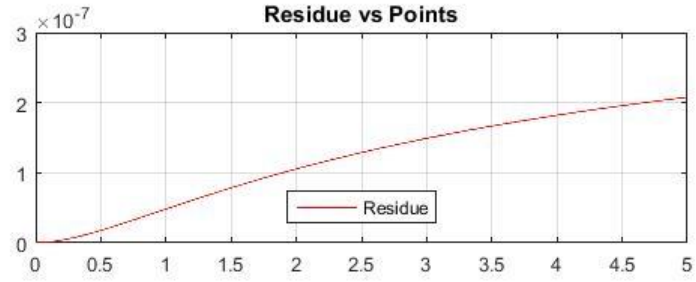


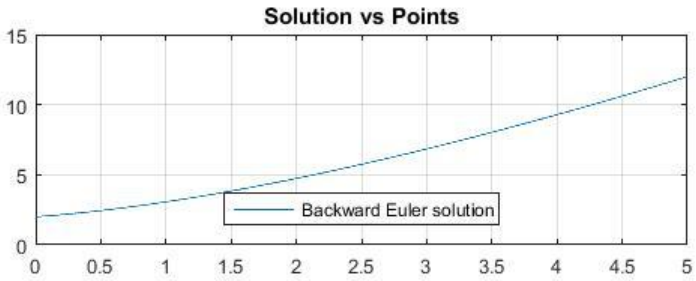
n = 1, h = 10^-4



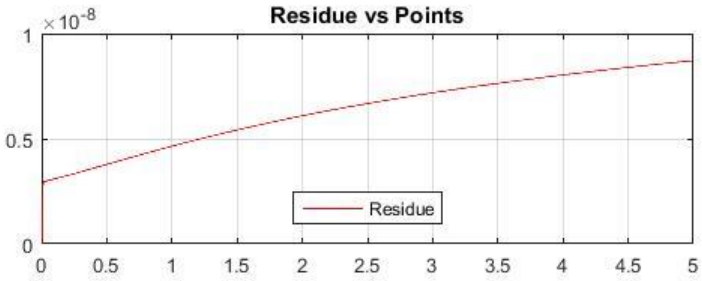


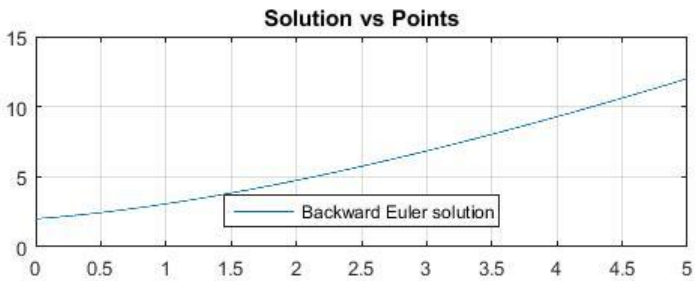
n = 10, h = 10^-2





n = 10, h = 10^-3





n = 10, h = 10^-4

c. Trapezoidal Method

Code	Output

clc
clearvars
fprintf('Trapezoidal Method\n\n')
%Input Iteration number
it=input('Number of iterations n: ');
h=input('\nInterval ''h'': ');
x=0:h:5;
%Defining zero row of order n for y
[n,n]=size(x);
ye=zeros(1,n);
ye(1)=2;
 	
%Forward Euler Formula for initial values
for i=1:n-1
  ye(i+1)=ye(i)+h*(log(x(i)+ye(i)));
end
%For simplicity y(converging) is given variable 'yc'
%and y(iteration) is given variable 'yy'
yc=ye;
yy=ye;
 	
for i=1:n-1
 	
  %Trapezoidal formula
  %case 1: Full convergence
  %Initial definition of error to pass while loop
  error=1;
  %dum stores the previous iteration value.
  %Y(i+1)(j+1)=F(y(i),y(i+1)(j))
  %here j represents iteration number
  %i represents index of y
  dum=yc(i+1);
  while error > 10^(-12)
  %Trapezoidal formula
 	
yc(i+1)=yc(i)+0.5*h*(log(x(i+1)+dum)+log(x(i)+yc(i)));
  error=abs(yc(i+1)-dum);
  dum=yc(i+1);
  end
 	
  if it==0
  yy=yc;
 	
  %else loop is written for predictor corrector scheme
  %Loop performs 10 iterartions for input of 10
  %For 1 iteration simple predictor corrector is obtained
  else
  for j=1:it
 	
yy(i+1)=yy(i)+0.5*h*(log(x(i+1)+yy(i+1))+log(x(i)+yy(i)));
  end
  end
end
	a. For N = 0 & h = 10^-2
	
	Trapezoidal Method
	
	Number of iterations n: 0
	
	Interval 'h': 10^-2
	
	y at x= 5 is 12.000000
	
	residue_max =
	
	0
	
	b. For N = 0 & h = 10^-3
	
	Trapezoidal Method
	
	Number of iterations n: 0
	
	Interval 'h': 10^-3
	
	y at x= 5 is 12.000008
	
	residue_max =
	
	0
	
	c. For N = 0 & h = 10^-4
	
	Trapezoidal Method
	
	Number of iterations n: 0
	
	Interval 'h': 10^-4
	
	y at x= 5 is 12.000008
	
	residue_max =
	
	0

```
if it==0
   fprintf('\ny at x= %d is %f\n\n',x(n),yc(n));
else
   fprintf('\ny at x= %d is %f\n\n',x(n),yy(n));
end
%Residue gives diffrence in converged value & iterated
value
residue=abs(yy-yc);
residue_max=max(residue)
%plot
figure
%Residue plot
subplot(2,1,1)
plot(x,residue,'-r')
title('Residue vs Points')
legend('Residue','Location','south')
grid on
%Solution plot
subplot(2,1,2)
plot(x,yc)
title('Solution vs Points')
legend('Trapezoidal solution','Location','south')
grid on
```

d. For N = 1 & h = 10^-2

Trapezoidal Method

Number of iterations n: 1

Interval 'h': 10^-2

y at x= 5 is 11.997194

residue_max =

0.0028

e. For N = 1 & h = 10^-3

Trapezoidal Method

Number of iterations n: 1

Interval 'h': 10^-3

y at x= 5 is 11.999728

residue_max =

2.8048e-04

f. For N = 1 & h = 10^-4

Trapezoidal Method

Number of iterations n: 1

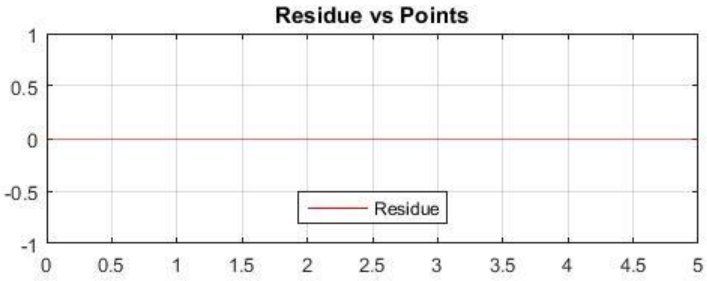
Interval 'h': 10^-4

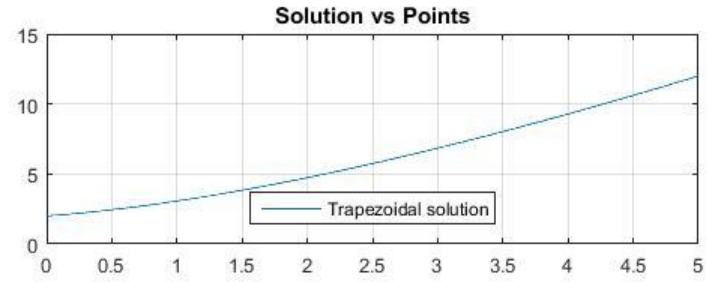
y at x= 5 is 11.999980

residue_max =

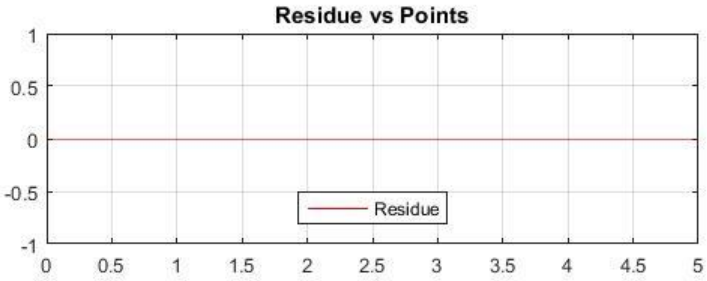
2.8046e-05

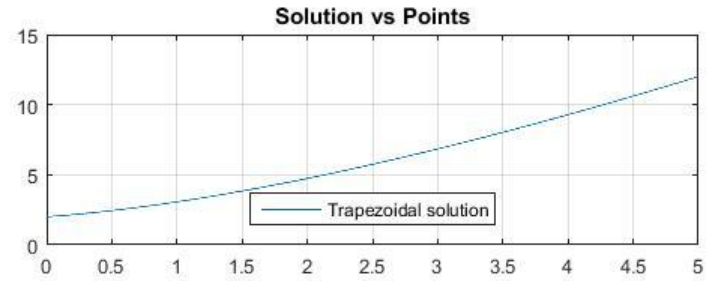
	g. For N = 10 & h = 10^-2 Trapezoidal Method Number of iterations n: 10 Interval 'h': 10^-2 y at x= 5 is 12.000000 residue_max = 4.6185e-14 **h. For N = 10 & h = 10^-3** Trapezoidal Method Number of iterations n: 10 Interval 'h': 10^-3 y at x= 5 is 12.000008 residue_max = 8.8818e-15 **i. For N = 10 & h = 10^-4** Trapezoidal Method Number of iterations n: 10 Interval 'h': 10^-4 y at x= 5 is 12.000008 residue_max = 3.5527e-15



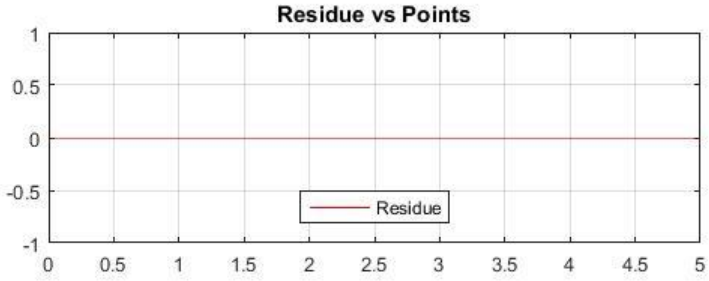


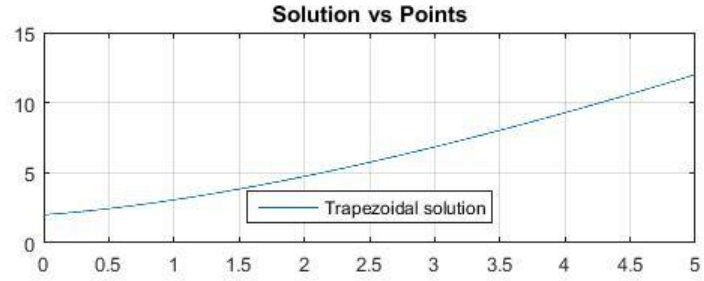
n = 0, h = 10^-2



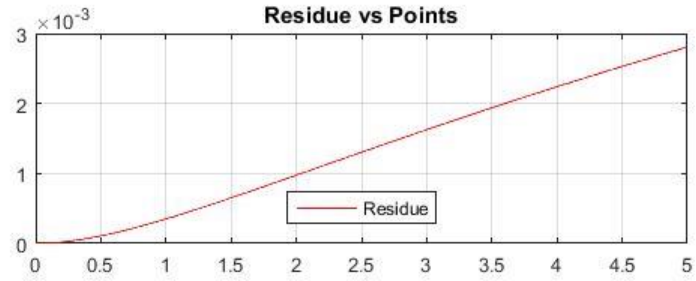


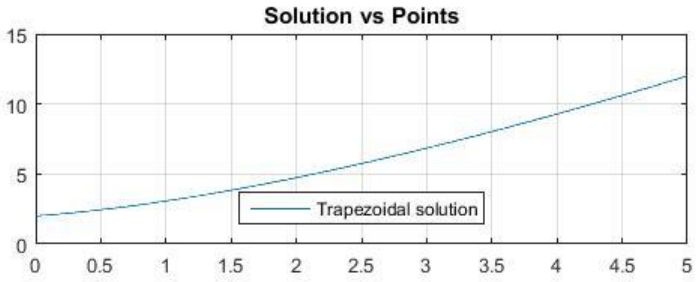
n = 0, h = 10^-3



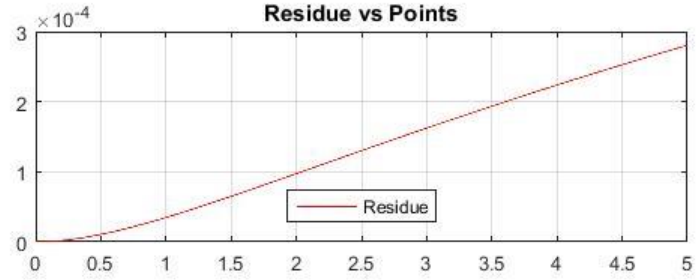


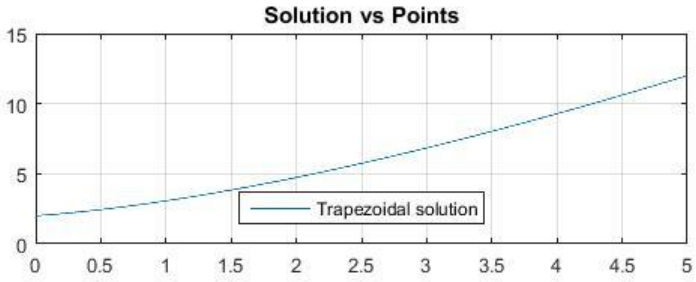
n = 0, h = 10^-4



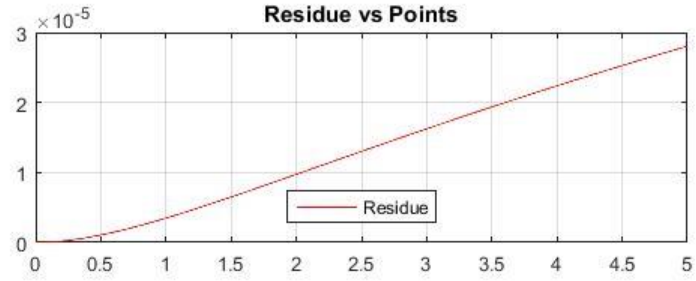


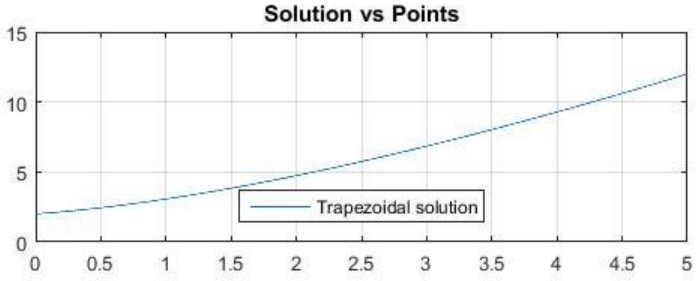
n = 1, h = 10^-2



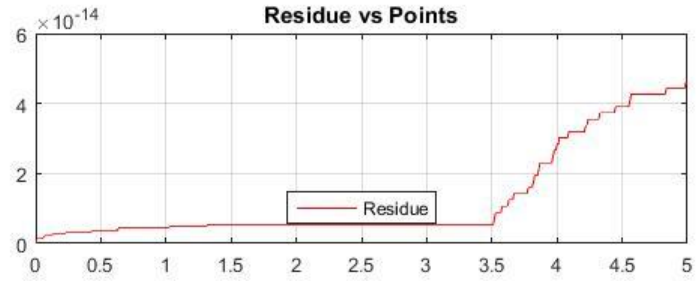


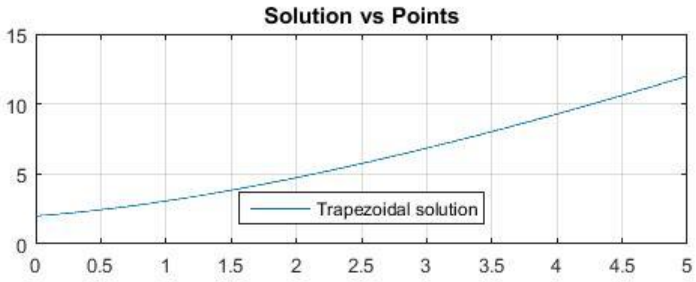
n = 1, h = 10^-3



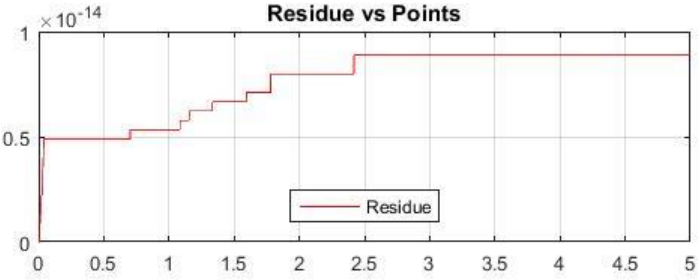


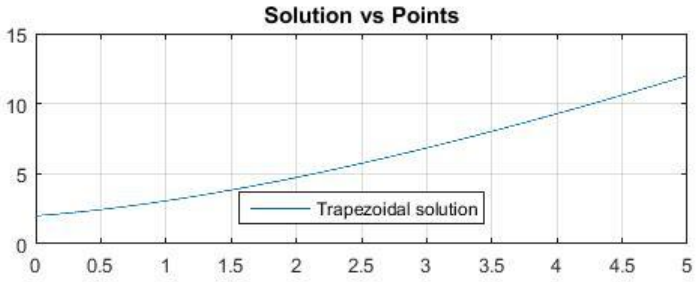
n = 1, h = 10^-4



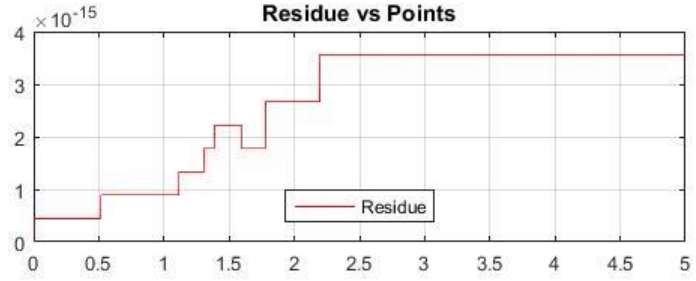


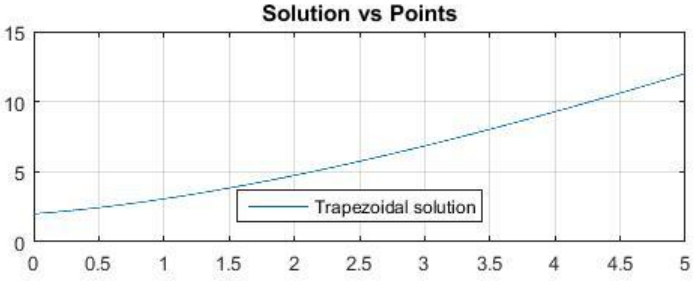
n = 10, h = 10^-2





n = 10, h = 10^-3





n = 10, h = 10^-4

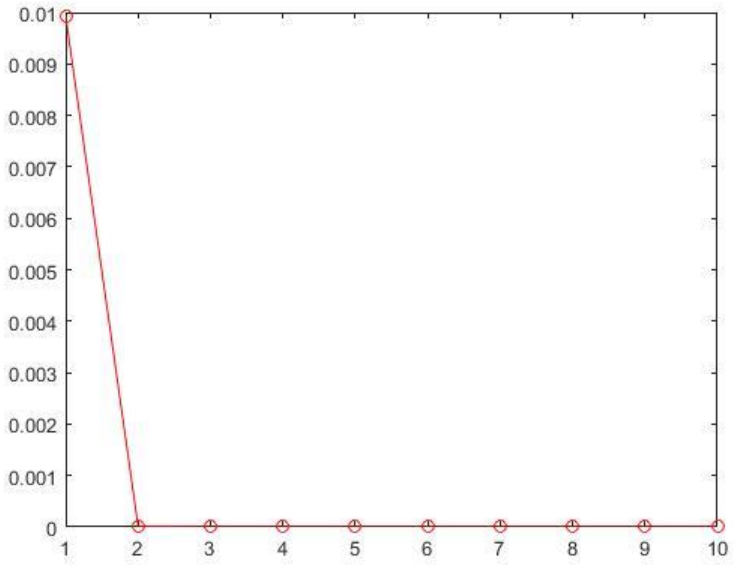
➢ For plotting Variation of residual Vs Iteration

a. Backward Euler Method

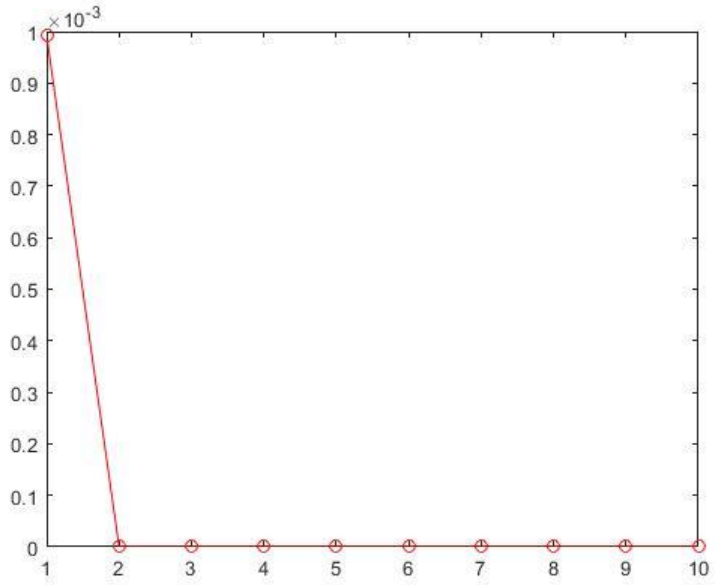
Code

```
clc
clearvars
h=input('\nInterval ''h'': ');
for k=1:10
%Input Iteration number
it=k;
x=0:h:5;
%Defining zero row of order n for y
[n,n]=size(x);
ye=zeros(1,n);
ye(1)=2;

%Forward Euler Formula for initial values
for i=1:n-1
   ye(i+1)=ye(i)+h*(log(x(i)+ye(i)));
end
%For simplicity y(converging) is given variable 'yc'
%and y(iteration) is given variable 'yy'
yc=ye;
yy=ye;

for i=1:n-1
   %Backward Euler formula
   %case 1: Full convergence
   %Initial definition of error to pass while loop
   error=1;
   %dum stores the previous iteration value.
   %Y(i+1)(j+1)=F(y(i),y(i+1)(j))
   %here j represents iteration number
   %i represents index of y
   dum=yc(i+1);
   while error > 10^(-6)
      %Backward Euler formula
      yc(i+1)=yc(i)+h*(log(x(i+1)+dum));
      error=abs(yc(i+1)-dum);
      dum=yc(i+1);
   end

   if it==0
      yy=yc;

 %else loop is written for predictor corrector scheme
 %Loop performs 10 iterartions for input of 10
 %For 1 iteration simple predictor corrector is obtained
  else
     for j=1:it
    yy(i+1)=yy(i)+h*(log(x(i+1)+yy(i+1)));
     end
   end

end
```

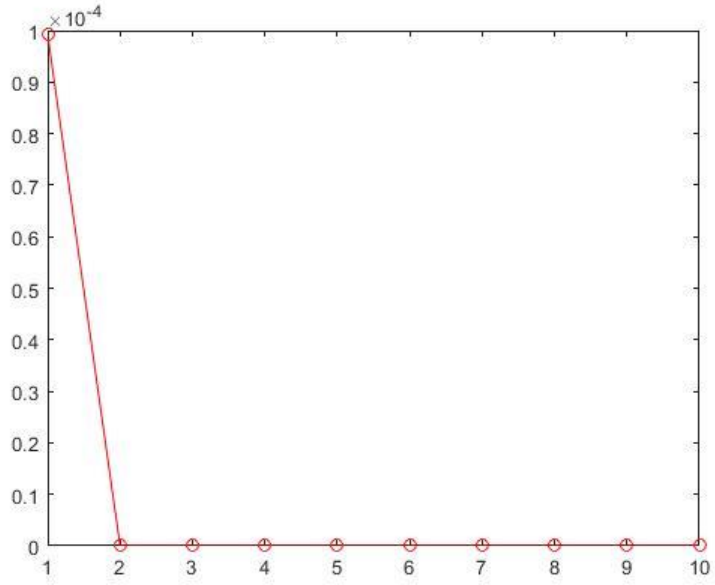
```
%Residue gives diffrence in converged value & iterated value
residue=abs(yy-yc);
residue_max(k)=max(residue);

end
plot([1:it],residue_max,'-ro')
```



h = 10^-2



h = 10^-3



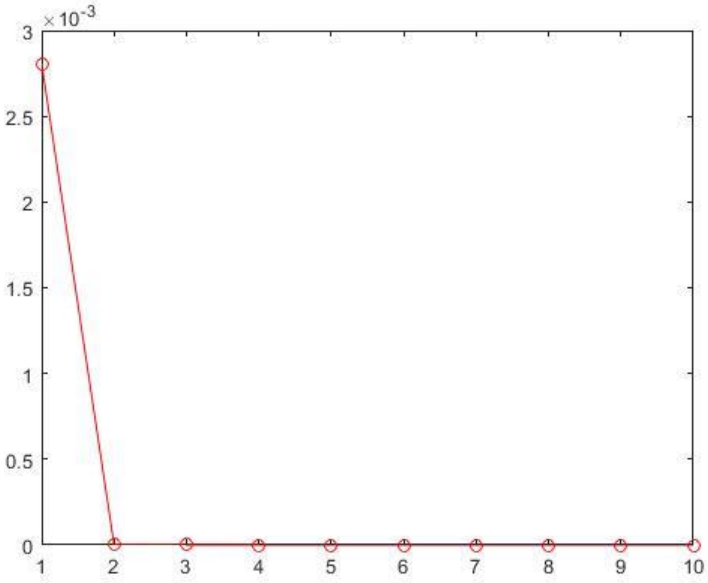
h = 10^-4

b. Trapezoidal Method

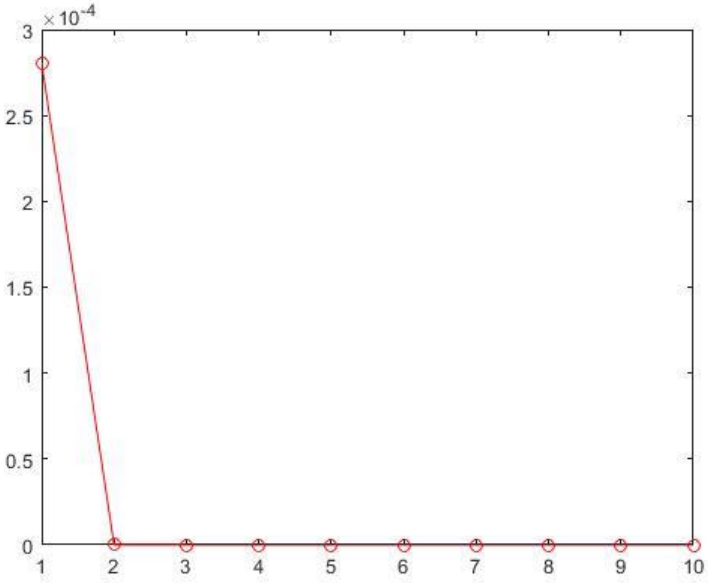
Code

```
clc
clearvars
h=input('\nInterval ''h'': ');
for k=1:10
%Input Iteration number
it=k;
x=0:h:5;
%Defining zero row of order n for y
[n,n]=size(x);
ye=zeros(1,n);
ye(1)=2;

%Forward Euler Formula for initial values
for i=1:n-1
   ye(i+1)=ye(i)+h*(log(x(i)+ye(i)));
end
%For simplicity y(converging) is given variable 'yc'
%and y(iteration) is given variable 'yy'
yc=ye;
yy=ye;


for i=1:n-1

   %Trapezoidal formula
   %case 1: Full convergence
   %Initial definition of error to pass while loop
   error=1;
   %dum stores the previous iteration value.
   %Y(i+1)(j+1)=F(y(i),y(i+1)(j))
   %here j represents iteration number
   %i represents index of y
   dum=yc(i+1);
   while error > 10^(-12)
      %Trapezoidal formula
      yc(i+1)=yc(i)+0.5*h*(log(x(i+1)+dum)+log(x(i)+yc(i)));
      error=abs(yc(i+1)-dum);
      dum=yc(i+1);
   end

   if it==0
      yy=yc;

  %else loop is written for predictor corrector scheme
  %Loop performs 10 iterartions for input of 10
  %For 1 iteration simple predictor corrector is obtained
   else
      for j=1:it
     yy(i+1)=yy(i)+0.5*h*(log(x(i+1)+yy(i+1))+log(x(i)+yy(i)));
      end
   end

end
```

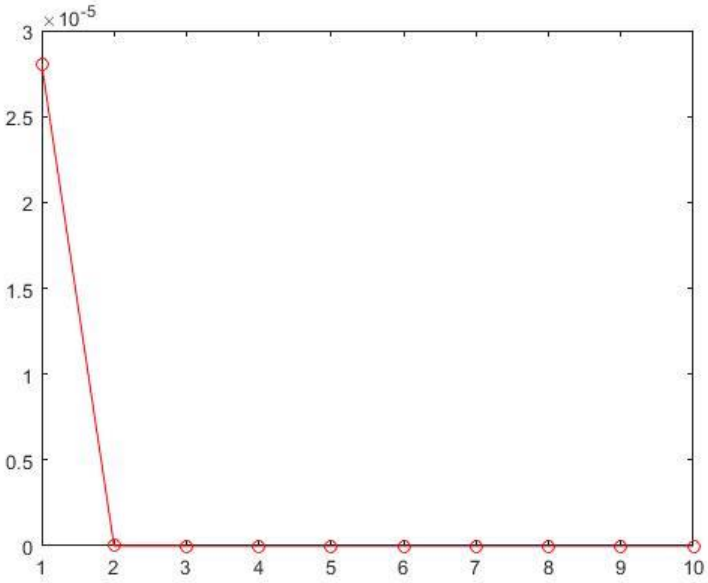
```
%Residue gives diffrence in converged value & iterated value
residue=abs(yy-yc);
residue_max(k)=max(residue);
end
plot([1:it],residue_max,'-ro')
```



h = 10^-2



h = 10^-3



h = 10^-4

➢ **Comparision of Results**

Max Error Table for all methods and different intervals

Backward Euler	10^-2	10^-3	10^-4
N = 0	0	0	0
N = 1	0.0099	9.9199e-04	9.9203e-05
N = 10	5.8168e-08	2.0793e-07	8.7180e-09

Trapezoidal	10^-2	10^-3	10^-4
N = 0	0	0	0
N = 1	0.0028	2.8048e-04	2.8046e-05
N = 10	4.6185e-14	8.8818e-15	3.5527e-15

➢ **Comparison within Each Method**

1. Backward Euler

a) N = 0
Maximum error is o

b) N = 1
The max error goes on decreasing if the in the value of h decreases
Error in 10^-2 > 10^-3 > 10^-4

c) N = 10
The max error increase and then decrease
Error in 10^-3 > 10^-2 > 10^-4

- For each h the maximum error decreases as the interval increases
  Error in n = 1 > n = 10

2. Trapezoidal

a) N = 0
   Maximum error is o

b) N = 1
   The max error goes on decreasing if the in the value of h decreases
   Error in 10^-2 > 10^-3 > 10^-4

c) N = 10
   The max error decrease and then increase
   Error in 10^-2 > 10^-4 > 10^-3

- For each h the maximum error decreases as the interval increases
  Error in n = 1 > n = 10

➢ **Comparison with Each other**

a) N = 0 for all h
   Maximum error in Backward Euler = maximum error in Trapezoidal

b) N = 1 for all h
   Maximum error in Backward Euler > maximum error in Trapezoidal

c) N = 10 for all h
   Maximum error in Backward Euler > maximum error in Trapezoidal

d) h = 10^-2 for all n
   Maximum error in Backward Euler = maximum error in Trapezoidal

e) h = 10^-3 for all n
   Maximum error in Backward Euler > maximum error in Trapezoidal

f) h = 10^-3 for all n
   Maximum error in Backward Euler > maximum error in Trapezoidal

- Trapezoidal is more accurate in predicting the results