

# EXPERIMENT-2

## Q1

2. The equation  $f(x) = x^3 + 4x^2 - 10 = 0$  has a unique root in  $[1,2]$ . There are many ways to change the equation to the fixed-point form  $x = g(x)$  using simple algebraic manipulation. Let  $g_1, g_2, g_3, g_4$  and  $g_5$  are iteration functions obtained by the given function, then check which of the following iteration functions will converge to the fixed point? (Tolerance  $\epsilon = 10^{-3}$ )

(a)  $g_1(x) = x - x^3 - 4x^2 + 10$

(b)  $g_2(x) = \sqrt{\frac{10}{x}} - 4x$

(c)  $g_3(x) = 0.5\sqrt{10 - x^3}$

(d)  $g_4(x) = \sqrt{\frac{10}{4+x}}$

(e)  $g_5(x) = x - \frac{x^3 + 4x^2 - 10}{3x^2 + 8x}$

## Ans1(a)

The screenshot shows a MATLAB script in the Editor and its execution in the Command Window and Workspace.

**Script Code:**

```

1 clc
2 clear
3 f=@(x)x^3+4*x^2-10;
4 g1=@(x)x-x^3-4*x^2+10;
5 tol=0.001;
6
7 N=input('Enter number of iterations \n');
8 x0=input('Enter initial approximation \n');
9
10 i=1;
11 while i<=N
12     x1= g1(x0);
13     if abs(x1-x0)<=tol || abs(x1-x0)/abs(x1)<=tol
14         break
15     else
16         x0=x1;
17     end
18     i=i+1;
19 end
20 if i>N
21     fprintf('doesnt coverge to a fixed point\n');
22 else
23     fprintf('successfull\n');
24     fprintf('The root is %f',x1);
25     fprintf('\nNumber of iterations taken is %f\n',i);
26 end
27

```

**Command Window Output:**

```

Enter number of iterations
100
Enter initial approximation
1.5
doesnt coverge to a fixed point
fx >>

```

**Workspace Variables:**

| Name | Value              |
|------|--------------------|
| f    | @(x)x^3+4*x^2-10   |
| g1   | @(x)x-x^3-4*x^2+10 |
| i    | 101                |
| N    | 100                |
| tol  | 1.0000e-03         |
| x0   | NaN                |
| x1   | NaN                |

## CODE:

```
clc
```

```
clear
```

```
f=@(x)x^3 +4*x^2 - 10;
```

```
g1=@(x)x - x^3 - 4*x^2 + 10;
```

```
tol=0.001;
```

```
N=input('Enter number of iterations \n');
```

```
x0=input('Enter initial approximation \n');
```

```
i=1;
```

```
while i<=N
```

```
    x1= g1(x0);
```

```
    if abs(x1-x0)<=tol || abs(x1-x0)/abs(x1)<=tol
```

**break**

**else**

**x0=x1;**

**end**

**i=i+1;**

**end**

**if i>N**

**fprintf('doesnt coverge to a fixed point\n');**

**else**

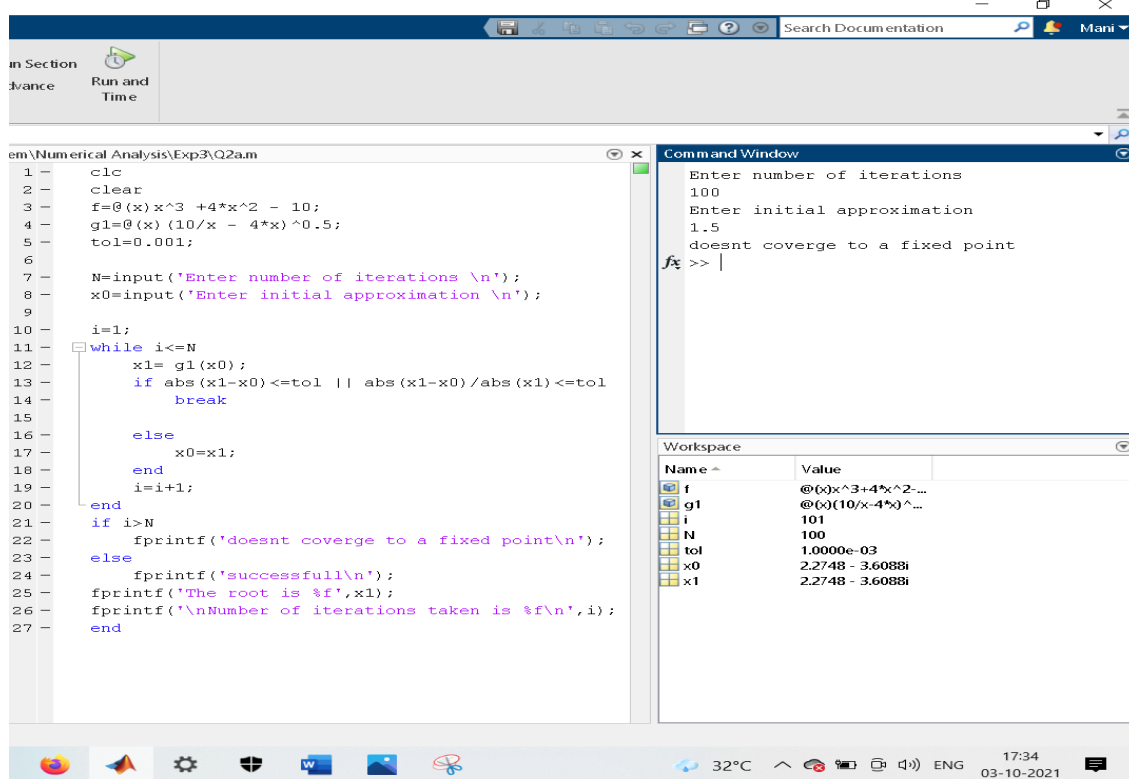
**fprintf('successfull\n');**

**fprintf('The root is %f',x1);**

**fprintf('\nNumber of iterations taken is  
%f\n',i);**

**end**

## Ans1(b)



```
1 clc
2 clear
3 f=@(x)x^3 +4*x^2 - 10;
4 g1=@(x)(10/x - 4*x)^0.5;
5 tol=0.001;
6
7 N=input('Enter number of iterations \n');
8 x0=input('Enter initial approximation \n');
9
10 i=1;
11 while i<=N
12     x1= g1(x0);
13     if abs(x1-x0)<=tol || abs(x1-x0)/abs(x1)<=tol
14         break
15     else
16         x0=x1;
17     end
18     i=i+1;
19 end
20 if i>N
21     fprintf('doesnt coverge to a fixed point\n');
22 else
23     fprintf('successfull\n');
24     fprintf('The root is %f',x1);
25     fprintf('\nNumber of iterations taken is %f\n',i);
26 end
27
```

Command Window

```
Enter number of iterations
100
Enter initial approximation
1.5
doesnt coverge to a fixed point
fx >> |
```

Workspace

| Name | Value              |
|------|--------------------|
| f    | @(x)x^3+4*x^2-10   |
| g1   | @(x)(10/x-4*x)^0.5 |
| i    | 101                |
| N    | 100                |
| tol  | 1.0000e-03         |
| x0   | 2.2748 - 3.6088i   |
| x1   | 2.2748 - 3.6088i   |

## CODE:

**clc**

**clear**

**f=@(x)x^3 +4\*x^2 - 10;**

**g1=@(x)(10/x - 4\*x)^0.5;**

**tol=0.001;**

**N=input('Enter number of iterations \n');**

```
x0=input('Enter initial approximation \n');

i=1;
while i<=N
    x1= g1(x0);
    if abs(x1-x0)<=tol || abs(x1-x0)/abs(x1)<=tol
        break
    else
        x0=x1;
    end
    i=i+1;
end
if i>N
    fprintf('doesnt coverge to a fixed point\n');
else
    fprintf('successfull\n');
```

```
fprintf('The root is %f',x1);
```

```
fprintf('\nNumber of iterations taken is  
%f\n',i);
```

```
end
```

Ans1(c)

The screenshot shows a MATLAB environment with a script editor on the left and a Command Window on the right. The script defines a function `g1` and uses a `while` loop to find the root of the equation  $x^3 + 4x^2 - 10 = 0$ . The Command Window shows the user inputting 100 for iterations and 1.5 for the initial approximation. The script outputs 'successfull', 'The root is 1.364878', and 'Number of iterations taken is 9.000000'. The Workspace window shows the values of variables `f`, `g1`, `i`, `N`, `tol`, `x0`, and `x1`.

```
1 clc
2 clear
3 f=@(x)x^3 +4*x^2 - 10;
4 g1=@(x)0.5 * (10 - x^3)^(0.5);
5 tol=0.001;
6
7 N=input('Enter number of iterations \n');
8 x0=input('Enter initial approximation \n');
9
10 i=1;
11 while i<=N
12     x1= g1(x0);
13     if abs(x1-x0)<=tol || abs(x1-x0)/abs(x1)<=tol
14         break
15     else
16         x0=x1;
17     end
18     i=i+1;
19 end
20 if i>N
21     fprintf('doesnt coverge to a fixed point\n');
22 else
23     fprintf('successfull\n');
24     fprintf('The root is %f',x1);
25     fprintf('\nNumber of iterations taken is %f\n',i);
26 end
27
```

Command Window

```
Enter number of iterations
100
Enter initial approximation
1.5
successfull
The root is 1.364878
Number of iterations taken is 9.000000
>>
```

Workspace

| Name | Value                |
|------|----------------------|
| f    | @(x)x^3+4*x^2-10     |
| g1   | @(x)0.5*(10-x^3)^0.5 |
| i    | 9                    |
| N    | 100                  |
| tol  | 1.0000e-03           |
| x0   | 1.3659               |
| x1   | 1.3649               |

CODE:

```
clc
```

```
clear
```

```
f=@(x)x^3 +4*x^2 - 10;
```

```
g1=@(x)0.5 * (10 - x^3)^(0.5);
```

```
tol=0.001;
```

```
N=input('Enter number of iterations \n');
```

```
x0=input('Enter initial approximation \n');
```

```
i=1;
```

```
while i<=N
```

```
    x1= g1(x0);
```

```
    if abs(x1-x0)<=tol || abs(x1-x0)/abs(x1)<=tol
```

```
        break
```

```
    else
```

```
        x0=x1;
```

```
    end
```

```
    i=i+1;
```

```
end
```

if i>N

fprintf('doesnt coverge to a fixed point\n');

else

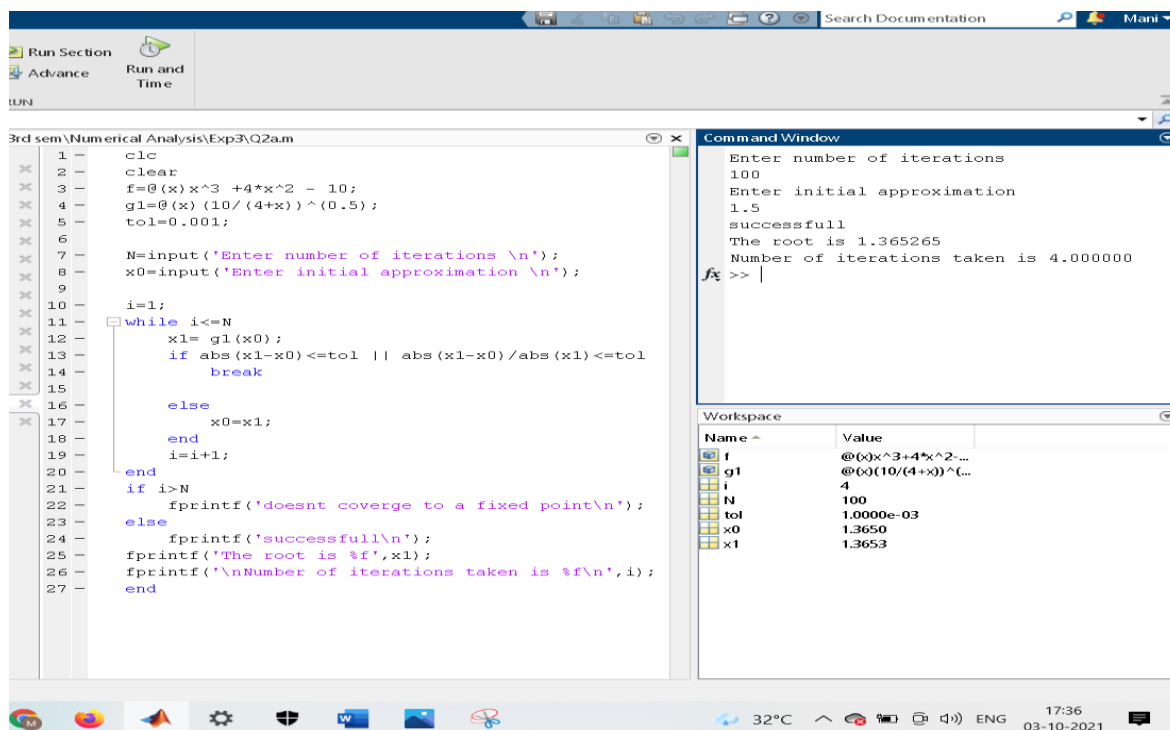
fprintf('successfull\n');

fprintf('The root is %f',x1);

fprintf('\nNumber of iterations taken is  
%f\n',i);

end

Ans1(d)



The screenshot displays a MATLAB environment with a script editor on the left and a Command Window on the right. The script, located at '3rd sem\Numerical Analysis\Exp3\Q2a.m', defines a function `g1` and a loop to find the root of the equation  $x^3 + 4x^2 - 10 = 0$ . The loop iterates up to `N=100` times, with a tolerance `tol=0.001`. The Command Window shows the user inputting 100 for iterations and 1.5 for the initial approximation. The output indicates success, showing the root as 1.365265 and 4.000000 iterations taken. The Workspace window on the right lists the variables: `f` (function handle), `g1` (function handle), `i` (4), `N` (100), `tol` (1.0000e-03), `x0` (1.3650), and `x1` (1.3653).

```
1 - clc
2 - clear
3 - f=@(x)x^3 +4*x^2 - 10;
4 - g1=@(x) (10/(4+x))^(0.5);
5 - tol=0.001;
6 -
7 - N=input('Enter number of iterations \n');
8 - x0=input('Enter initial approximation \n');
9 -
10 - i=1;
11 - while i<=N
12 -     x1= g1(x0);
13 -     if abs(x1-x0)<=tol || abs(x1-x0)/abs(x1)<=tol
14 -         break
15 -     else
16 -         x0=x1;
17 -     end
18 -     i=i+1;
19 - end
20 -
21 - if i>N
22 -     fprintf('doesnt coverge to a fixed point\n');
23 - else
24 -     fprintf('successfull\n');
25 -     fprintf('The root is %f',x1);
26 -     fprintf('\nNumber of iterations taken is %f\n',i);
27 - end
```

Command Window:

```
Enter number of iterations
100
Enter initial approximation
1.5
successfull
The root is 1.365265
Number of iterations taken is 4.000000
fx >>
```

Workspace:

| Name | Value                |
|------|----------------------|
| f    | @(x)x^3+4*x^2-10     |
| g1   | @(x)(10/(4+x))^(0.5) |
| i    | 4                    |
| N    | 100                  |
| tol  | 1.0000e-03           |
| x0   | 1.3650               |
| x1   | 1.3653               |



## CODE:

```
clc
```

```
clear
```

```
f=@(x)x^3 +4*x^2 - 10;
```

```
g1=@(x)(10/(4+x))^(0.5);
```

```
tol=0.001;
```

```
N=input('Enter number of iterations \n');
```

```
x0=input('Enter initial approximation \n');
```

```
i=1;
```

```
while i<=N
```

```
    x1= g1(x0);
```

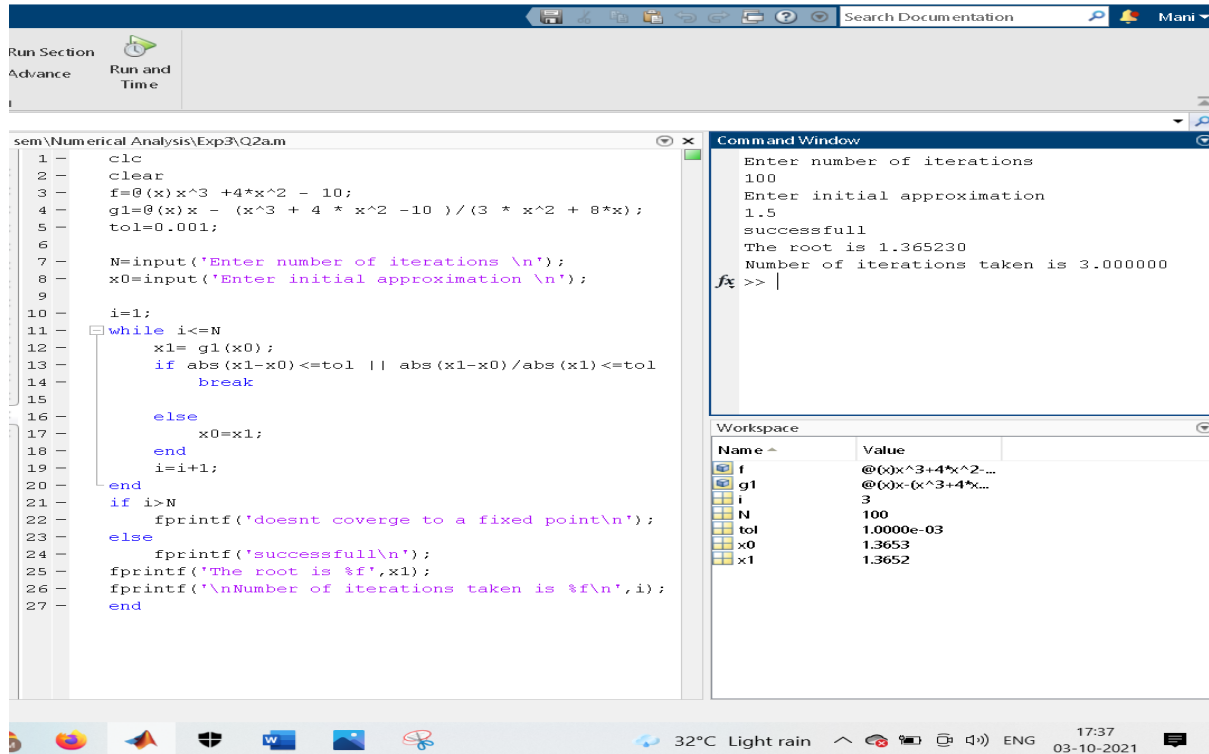
```
    if abs(x1-x0)<=tol || abs(x1-x0)/abs(x1)<=tol
```

```
        break
```

```
    else
```

```
        x0=x1;
    end
    i=i+1;
end
if i>N
    fprintf('doesnt coverge to a fixed point\n');
else
    fprintf('successfull\n');
    fprintf('The root is %f',x1);
    fprintf('\nNumber of iterations taken is
    %f\n',i);
end
```

**Ans1(e)**



**CODE:**

**clc**

**clear**

**f=@(x)x^3 +4\*x^2 - 10;**

**g1=@(x)x - (x^3 + 4 \* x^2 -10 )/(3 \* x^2 + 8\*x);**

**tol=0.001;**

**N=input('Enter number of iterations \n');**

**x0=input('Enter initial approximation \n');**

```
i=1;
while i<=N
    x1= g1(x0);
    if abs(x1-x0)<=tol || abs(x1-x0)/abs(x1)<=tol
        break
    else
        x0=x1;
    end
    i=i+1;
end
if i>N
    fprintf('doesnt coverge to a fixed point\n');
else
    fprintf('successfull\n');
    fprintf('The root is %f',x1);
```

```
fprintf('\nNumber of iterations taken is
%f\n',i);
```

```
end
```

## Q2

- Find the smallest and second smallest positive roots of the equation  $\tan(x) = 4x$ , with an accuracy of  $10^{-3}$  using fixed-point iterations.

## Ans2

The image shows a MATLAB script in the Editor and the Command Window. The script defines a function `g1` and uses a while loop to find the roots of the equation  $\tan(x) = 4x$  using fixed-point iterations. The Command Window shows the execution results for two different initial approximations.

```

1  clc
2  clear
3  f=@(x) 4*x-tan(x);
4  g1=@(x) x-((4*x-tan(x))/(4-(sec(x))^2));
5  tol=0.001;
6  h=0.1;
7  for i=0:h:2
8      if (f(i)*f(i+h)<0)
9          a=i;
10         b=i+h;
11         c=(a+b)/2;
12         break;
13     end
14 end
15 for i=2:h:5
16     if (f(i)*f(i+h)<0)
17         a=i;
18         b=i+h;
19         d=(a+b)/2;
20         break;
21     end
22 end
23 x0=c;
24 for k=1:1:2
25
26     N=input('Enter number of iterations \n');
27     i=1;
28     fprintf('Initial Approximation(via IVT): %f\n',x0);
29     while i<=N
30         x1= g1(x0);
31         if abs(x1-x0)<=tol || abs(x1-x0)/abs(x1)<=tol
32             break

```

Command Window Output:

```

Enter number of iterations
100
Initial Approximation(via IVT): 1.350000
successfull
The 1 smallest root is 1.393256
Number of iterations taken is 3.000000

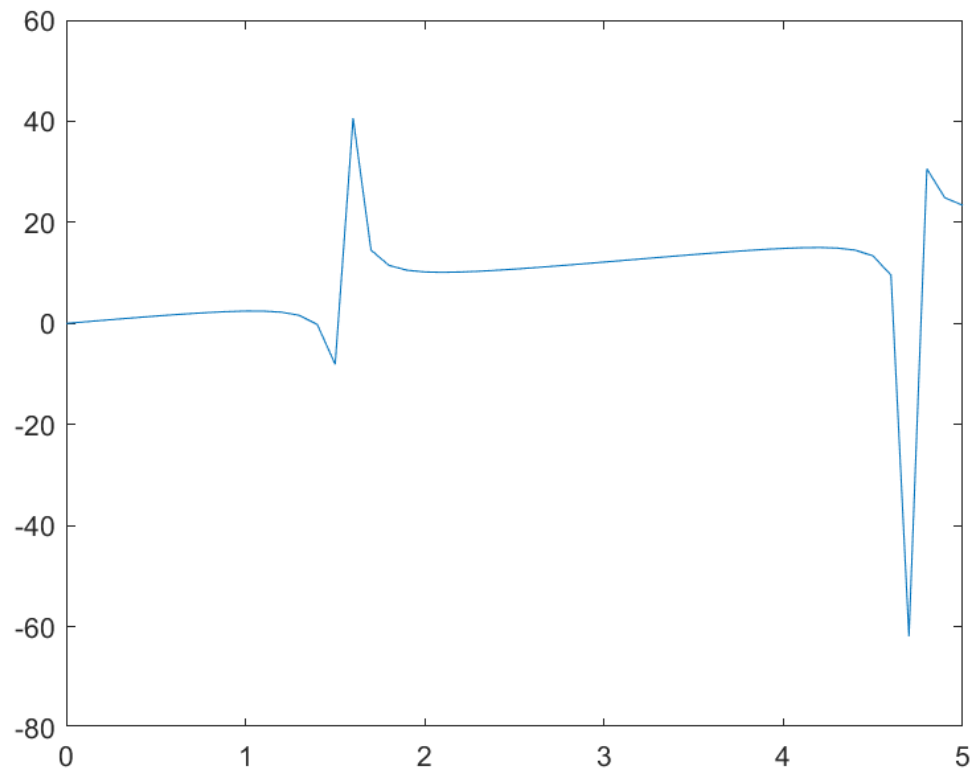
Enter number of iterations
100
Initial Approximation(via IVT): 4.650000
successfull
The 2 smallest root is 4.658818
Number of iterations taken is 2.000000

```

Workspace:

| Name | Value               |
|------|---------------------|
| a    | 4.6000              |
| b    | 4.7000              |
| c    | 1.3500              |
| d    | 4.6500              |
| f    | @(x)4*x-tan(x)      |
| g1   | @(x)x-((4*x-tan(... |
| h    | 0.1000              |
| i    | 2                   |
| k    | 2                   |
| N    | 100                 |
| tol  | 1.0000e-03          |
| x0   | 4.6500              |
| x1   | 4.6588              |

## GRAPH:



## CODE:

```
clc
```

```
clear
```

```
f=@(x)4*x-tan(x);
```

```
g1=@(x)x-((4*x-tan(x))/(4-(sec(x))^2));
```

```
tol=0.001;
```

```
h=0.1;
```

```
for i=0:h:2
```

**if(f(i)\*f(i+h)<0)**

**a=i;**

**b=i+h;**

**c=(a+b)/2;**

**break;**

**end**

**end**

**for i=2:h:5**

**if(f(i)\*f(i+h)<0)**

**a=i;**

**b=i+h;**

**d=(a+b)/2;**

**break;**

**end**

**end**

**x0=c;**

**for k=1:1:2**

**N=input('Enter number of iterations \n');**

**i=1;**

**fprintf('Initial Approximation(via IVT):  
%f\n',x0);**

**while i<=N**

**x1= g1(x0);**

**if abs(x1-x0)<=tol || abs(x1-x0)/abs(x1)<=tol**

**break**

**else**

**x0=x1;**

**end**

**i=i+1;**

**end**

**if i>N**



```
fprintf('doesnt coverge to a fixed point\n');  
else  
    fprintf('successfull\n');  
    fprintf('The %d smallest root is %f',k,x1);  
    fprintf('\nNumber of iterations taken is  
    %f\n\n',i);  
end  
  
x0=d;  
end
```

## Q3

4. Use a fixed-point iteration method to determine a solution accurate to within  $10^{-2}$  for  $2\sin\pi x + x = 0$  on  $[1,2]$ . Use initial guess  $x_0 = 1$ .

## Ans3

The image shows a MATLAB script in the Editor and the Command Window. The script implements a fixed-point iteration method to solve the equation  $2\sin(\pi x) + x = 0$  on the interval  $[1, 2]$  with an initial guess  $x_0 = 1$ . The script defines the function  $f(x) = 2\sin(\pi x) + x$  and the iteration function  $g_1(x) = x - (2\sin(\pi x) + x) / (2\pi\cos(\pi x) + 1)$ . It then iterates until the absolute error is less than  $10^{-2}$  or the maximum number of iterations (100) is reached.

```
1 - clc
2 - clear
3 - f=@(x) 2*sin(pi*x)+x;
4 - g1=@(x) x- ((2*sin(pi*x)+x)/(2*pi*cos(pi*x)+1));
5 - tol=0.01;
6
7 - N=input('Enter number of iterations \n');
8 - x0=input('Enter initial approximation \n');
9 - i=1;
10 - while i<=N
11 -     x1= g1(x0);
12 -     if abs(x1-x0)<=tol || abs(x1-x0)/abs(x1)<=tol
13 -         break
14 -     else
15 -         x0=x1;
16 -     end
17 -     i=i+1;
18 - end
19 - if i>N
20 -     fprintf('doesnt coverge to a fixed point\n');
21 - else
22 -     fprintf('successfull\n');
23 -     fprintf('The root is %f',x1);
24 -     fprintf('\nNumber of iterations taken is %f\n',i);
25 - end
26 -
```

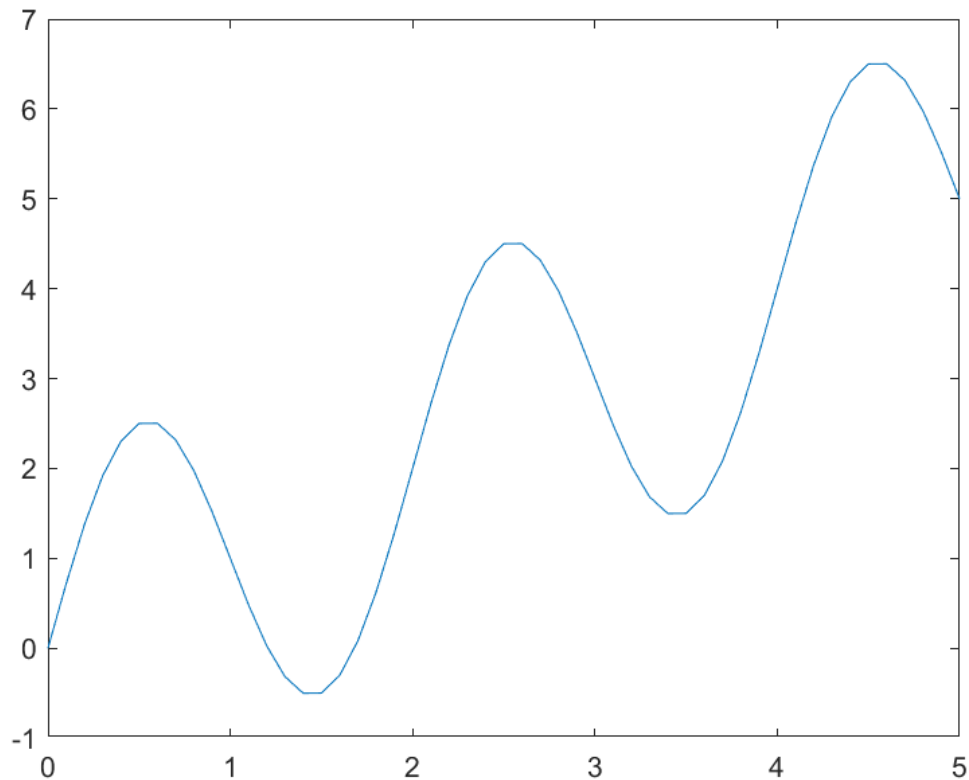
The Command Window shows the following output:

```
Enter number of iterations
100
Enter initial approximation
1
successfull
The root is 1.206035
Number of iterations taken is 3.000000
fx >>
```

The Workspace window shows the following variables:

| Name | Value                                      |
|------|--|
| f    | @(x)2*sin(pi*x)+x                          |
| g1   | @(x)x-((2*sin(pi*x)+x)/(2*pi*cos(pi*x)+1)) |
| i    | 3  |
| N    | 100  |
| tol  | 0.0100                                     |
| x0   | 1.2057                                     |
| x1   | 1.2060                                     |

## GRAPH:



**CODE:**

**clc**

**clear**

**f=@(x)2\*sin(pi\*x)+x;**

**g1=@(x)x-((2\*sin(pi\*x)+x)/(2\*pi\*cos(pi\*x)+1));**

**tol=0.01;**

**N=input('Enter number of iterations \n');**

**x0=input('Enter initial approximation \n');**

```
i=1;
while i<=N
    x1= g1(x0);
    if abs(x1-x0)<=tol || abs(x1-x0)/abs(x1)<=tol
        break
    else
        x0=x1;
    end
    i=i+1;
end
if i>N
    fprintf('doesnt coverge to a fixed point\n');
else
    fprintf('successfull\n');
    fprintf('The root is %f',x1);
```

```
fprintf('\nNumber of iterations taken is  
%f\n',i);  
end
```