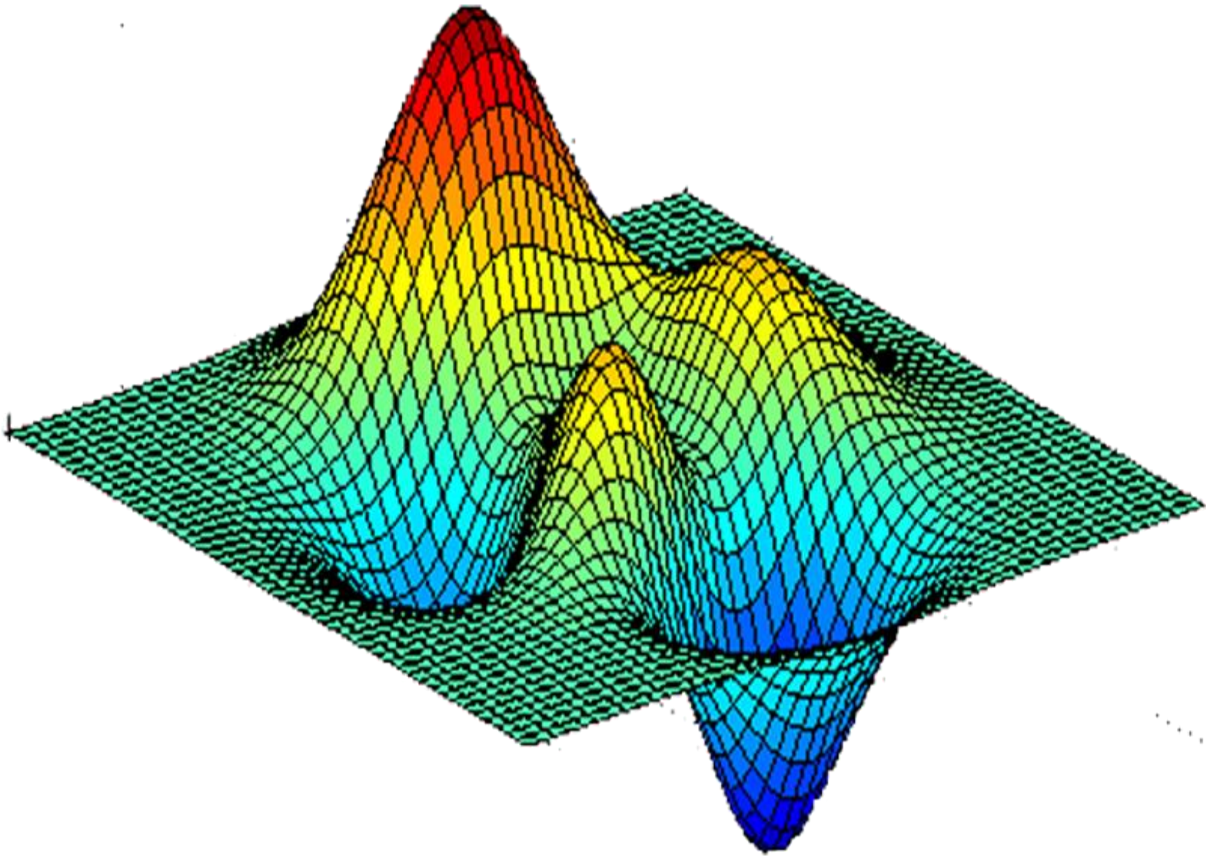
Numerical Computing

Secant Method and Regula Falsi Method



QASID AHMED ALEEM

COURSE TEACHER: DR. SHAHID QURESHI
BSC(HONOURS)
H-2 (MORNING)
INSTITUTE OF SPACE AND PLANETARY ASTROPHYSICS
UNIVERSITY OF KARACHI
Date:13/2/18

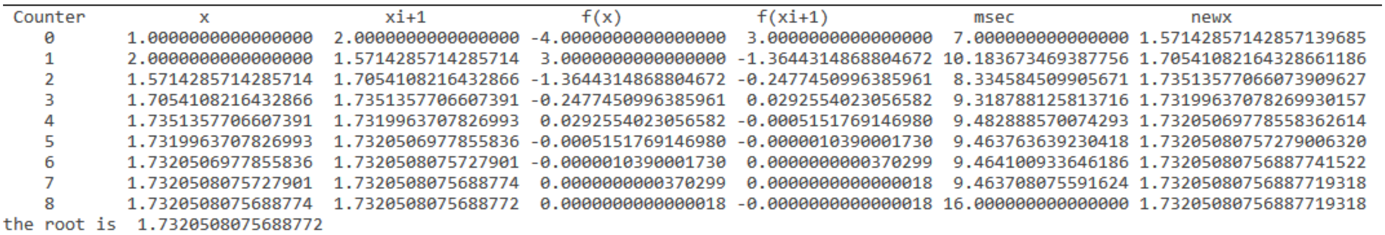
Tasks

Task 1: Secant Method

Python Code:

```
#Qasid Ahmed Aleem
#Secant Method
def func(f):                                    #given function
        return(f**3)+(f**2)-(3*f)-3

def msec(xi,xi1):                               #secant function
        return (func(xi1)-func(xi))/(xi1-xi)

def newx(xi,xi1):                               #new x function
        return xi-(func(xi)/msec(xi,xi1))

xi =1.0 #a
xi1=2.0
print("{:^10s} {:^18s} {:^18s} {:^19s}{:^18s} {:^19s} {:^21s}".format(
"Counter","x","xi+1","f(x)","f(xi+1)","msec","newx"))
for x in range(0,30):

        if xi-xi1==0:
                print("the root is ",xi)
                break
        print("{:^10d} {: 18.16f} {: 18.16f} {: 18.16f} {: 18.16f}\
{: 19.15f} {:21.20f}".format(x,xi,
        xi1,func(xi),func(xi1),msec(xi,xi1),newx(xi,xi1)))


        xn=xi
        xi=xi1

        xi1=newx(xn,xi)
```

Output:



```
 Counter           x                 xi+1              f(x)              f(xi+1)             msec                 newx
    0       1.0000000000000000  2.0000000000000000 -4.0000000000000000  3.0000000000000000  7.000000000000000 1.57142857142857139685
    1       2.0000000000000000  1.5714285714285714  3.0000000000000000 -1.3644314868804672 10.183673469387756 1.70541082164328661186
    2       1.5714285714285714  1.7054108216432866 -1.3644314868804672 -0.2477450996385961  8.334584509905671 1.73513577066073909627
    3       1.7054108216432866  1.7351357706607391 -0.2477450996385961  0.0292554023056582  9.318788125813716 1.73199637078269930157
    4       1.7351357706607391  1.7319963707826993  0.0292554023056582 -0.0005151769146980  9.482888570074293 1.73205069778558362614
    5       1.7319963707826993  1.7320506977855836 -0.0005151769146980 -0.0000010390001730  9.463763639230418 1.73205080757279006320
    6       1.7320506977855836  1.7320508075727901 -0.0000010390001730  0.0000000000370299  9.464100933646186 1.73205080756887741522
    7       1.7320508075727901  1.7320508075688774  0.0000000000370299  0.0000000000000018  9.463708075591624 1.73205080756887719318
    8       1.7320508075688774  1.7320508075688772  0.0000000000000018 -0.0000000000000018 16.000000000000000 1.73205080756887719318
the root is  1.7320508075688772
```

Excel Table:

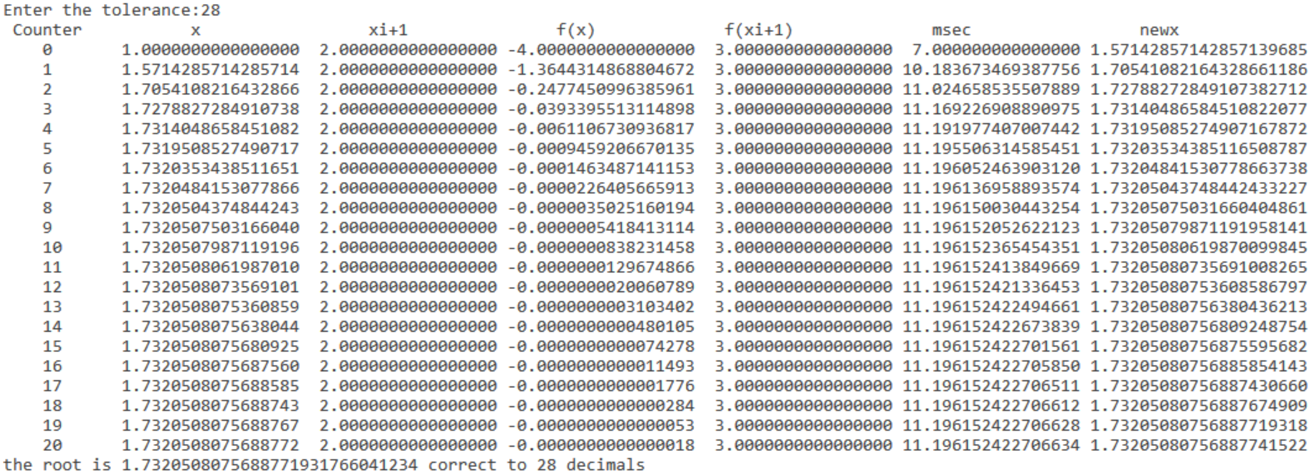
counter	xi	xi+1	f(x)	f(xi+1)	msec=f(xi+1)-f(xi)/(xi+1)-xi	xi+2=xi-f(xi)/msec
1	1	2	-4	3	7	1.571428571
2	2	1.571428571	3	-1.36443149	10.18367347	1.705410822
3	1.571428571429	1.705410822	-1.36443149	-0.2477451	8.33458451	1.735135771
4	1.705410821643	1.735135771	-0.2477451	0.029255402	9.318788126	1.731996371
5	1.735135770661	1.731996371	0.029255402	-0.00051518	9.48288857	1.732050698
6	1.731996370783	1.732050698	-0.00051518	-1.039E-06	9.463763639	1.732050808
7	1.732050697786	1.732050808	-1.039E-06	3.70299E-11	9.464100934	1.732050808
8	1.732050807573	1.732050808	3.70299E-11	0	9.464162079	1.732050808
9	1.732050807569	1.732050808	0	0	#DIV/0!	#DIV/0!

Task 2:Regula Falsi Method

Python Code:

```
#Qasid Ahmed Aleem
#Regula Falsi Method

def func(f):                                    #given function
        return(f**3)+(f**2)-(3*f)-3

def msec(xi,xi1):                               #secant function
        return (func(xi1)-func(xi))/(xi1-xi)

def newx(xi,xi1):                               #new x function
        return xi-(func(xi)/msec(xi,xi1))

xi =1.0 #a
xi1=2.0

i=int(input("Enter the tolerance:"))
print("{:^10s} {:^18s} {:^18s} {:^19s}{:^18s} {:^19s} {:^21s}".format(
"Counter","x","xi+1","f(x)","f(xi+1)","msec","newx"))
for x in range(0,1000):

        if (newx(xi,xi1)-xi)<1*(10**-i):
                print("the root is {:30.28f} correct to {:d} decimals
".format(\
                newx(xi,xi1),i))
                break
        print("{:^10d} {: 18.16f} {: 18.16f} {: 18.16f} {: 18.16f}\
{: 19.15f} {:21.20f}".format(x,xi,
        xi1,func(xi),func(xi1),msec(xi,xi1),newx(xi,xi1)))


        if (func(xi)*func(newx(xi,xi1)))<0:
                xi1=newx(xi,xi1)
        else:
                xi=newx(xi,xi1)
```

Output:



```
Enter the tolerance:28
 Counter             x                  xi+1                 f(x)                f(xi+1)               msec                  newx
    0       1.0000000000000000  2.0000000000000000 -4.0000000000000000  3.0000000000000000  7.000000000000000 1.57142857142857139685
    1       1.5714285714285714  2.0000000000000000 -1.3644314868804672  3.0000000000000000 10.183673469387756 1.70541082164328661186
    2       1.7054108216432866  2.0000000000000000 -0.2477450996385961  3.0000000000000000 11.024658535507889 1.72788272849107382712
    3       1.7278827284910738  2.0000000000000000 -0.0393395513114898  3.0000000000000000 11.169226908890975 1.73140486584510822077
    4       1.7314048658451082  2.0000000000000000 -0.0061106730936817  3.0000000000000000 11.191977407007442 1.73195085274907167872
    5       1.7319508527490717  2.0000000000000000 -0.0009459206670135  3.0000000000000000 11.195506314585451 1.73203534385116508787
    6       1.7320353438511651  2.0000000000000000 -0.0001463487141153  3.0000000000000000 11.196052463903120 1.73204841530778663738
    7       1.7320484153077866  2.0000000000000000 -0.0000226405665913  3.0000000000000000 11.196136958893574 1.73205043748442433227
    8       1.7320504374844243  2.0000000000000000 -0.0000035025160194  3.0000000000000000 11.196150030443254 1.73205075031660404861
    9       1.7320507503166040  2.0000000000000000 -0.0000005418413114  3.0000000000000000 11.196152052622123 1.73205079871191958141
    10      1.7320507987119196  2.0000000000000000 -0.0000000838231458  3.0000000000000000 11.196152365454351 1.73205080619870099845
    11      1.7320508061987010  2.0000000000000000 -0.0000000129674866  3.0000000000000000 11.196152413849669 1.73205080735691008265
    12      1.7320508073569101  2.0000000000000000 -0.0000000020060789  3.0000000000000000 11.196152421336453 1.73205080753608586797
    13      1.7320508075360859  2.0000000000000000 -0.0000000003103402  3.0000000000000000 11.196152422494661 1.73205080756380436213
    14      1.7320508075638044  2.0000000000000000 -0.0000000000480105  3.0000000000000000 11.196152422673839 1.73205080756809248754
    15      1.7320508075680925  2.0000000000000000 -0.0000000000074278  3.0000000000000000 11.196152422701561 1.73205080756875595682
    16      1.7320508075687560  2.0000000000000000 -0.0000000000011493  3.0000000000000000 11.196152422705850 1.73205080756885854143
    17      1.7320508075688585  2.0000000000000000 -0.0000000000001776  3.0000000000000000 11.196152422706511 1.73205080756887430660
    18      1.7320508075688743  2.0000000000000000 -0.0000000000000284  3.0000000000000000 11.196152422706612 1.73205080756887674909
    19      1.7320508075688767  2.0000000000000000 -0.0000000000000053  3.0000000000000000 11.196152422706628 1.73205080756887719318
    20      1.7320508075688772  2.0000000000000000 -0.0000000000000018  3.0000000000000000 11.196152422706634 1.73205080756887741522
the root is 1.7320508075688771931766041234 correct to 28 decimals
```

Excel Table:

counter	xi	xi+1	f(x)	f(xi+1)	msec=f(xi+1)-f(xi)/(xi+1)-xi	xi+2=xi-f(xi)/msec	f(xi+2)
1	1.00000000000	2.0000000	-4.0000000	3.0000000	7.0000000	1.57142857143	-1.3644315
2.0	1.57142857143	2.0000000	-1.3644315	3.0000000	10.1836735	1.70541082164	-0.2477451
3.0	1.70541082164	2.0000000	-0.2477451	3.0000000	11.0246585	1.72788272849	-0.0393396
4.0	1.72788272849	2.0000000	-0.0393396	3.0000000	11.1692269	1.73140486585	-0.0061107
5.0	1.73140486585	2.0000000	-0.0061107	3.0000000	11.1919774	1.73195085275	-0.0009459
6.0	1.73195085275	2.0000000	-0.0009459	3.0000000	11.1955063	1.73203534385	-0.0001463
7.0	1.73203534385	2.0000000	-0.0001463	3.0000000	11.1960525	1.73204841531	-0.0000226
8.0	1.73204841531	2.0000000	-0.0000226	3.0000000	11.1961370	1.73205043748	-0.0000035
9.0	1.73205043748	2.0000000	-0.0000035	3.0000000	11.1961500	1.73205075032	-0.0000005
10.0	1.73205075032	2.0000000	-0.0000005	3.0000000	11.1961521	1.73205079871	-0.0000001
11.0	1.73205079871	2.0000000	-0.0000001	3.0000000	11.1961524	1.73205080620	0.0000000
12.0	1.73205080620	2.0000000	0.0000000	3.0000000	11.1961524	1.73205080736	0.0000000
13.0	1.73205080736	2.0000000	0.0000000	3.0000000	11.1961524	1.73205080754	0.0000000
14.0	1.73205080754	2.0000000	0.0000000	3.0000000	11.1961524	1.73205080756	0.0000000
15.0	1.73205080756	2.0000000	0.0000000	3.0000000	11.1961524	1.73205080757	0.0000000
16.0	1.73205080757	2.0000000	0.0000000	3.0000000	11.1961524	1.73205080757	0.0000000