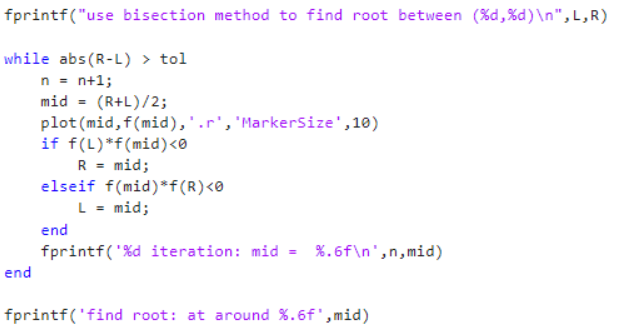
110550126 曾家祐

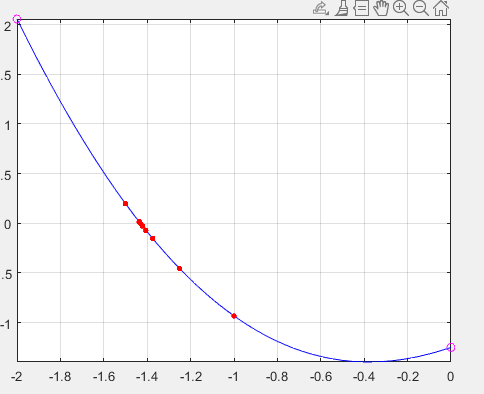
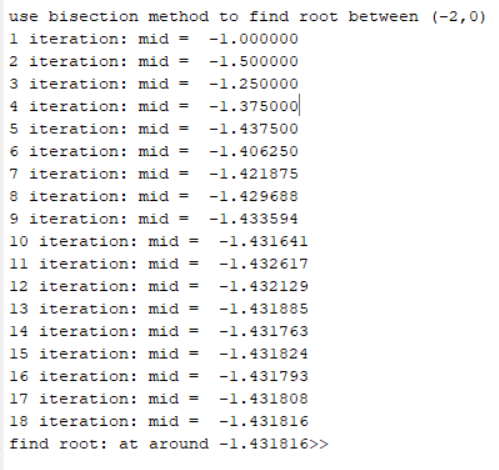
Numerical methods Assignment 1

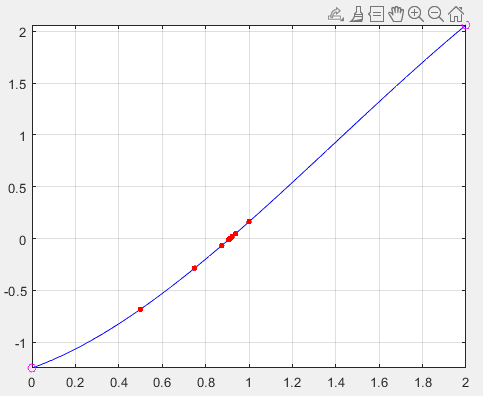
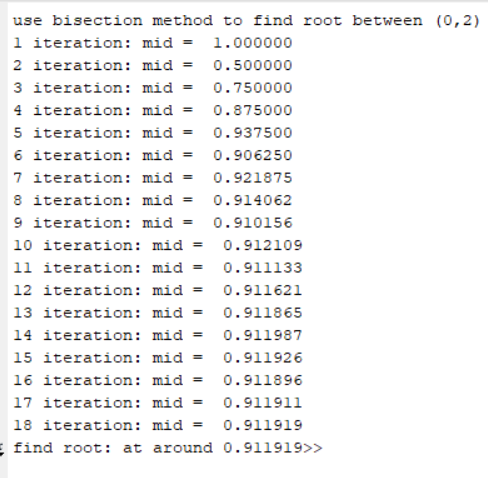
1. Find two roots for f(x)
2. Bisection method

we use a while loop base on interval of L and R to decide how many iteration to run, every time we see the mid point and use new mid point to replace the old point and we can get the two roots is -1.431816 and 0.911919





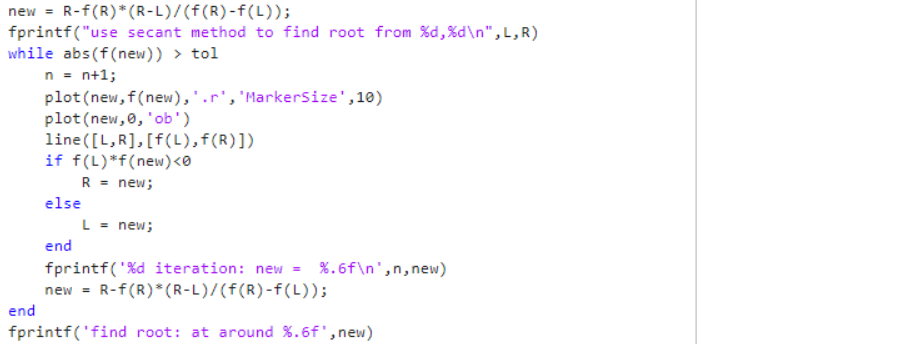


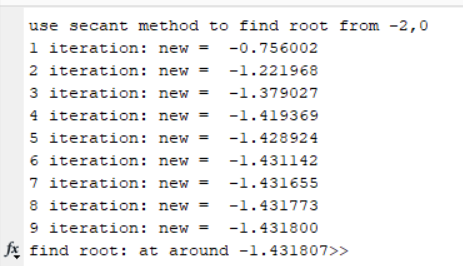
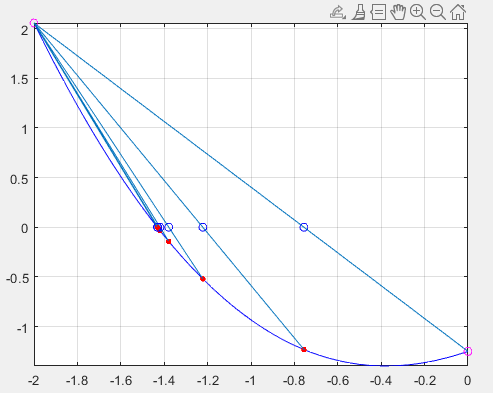


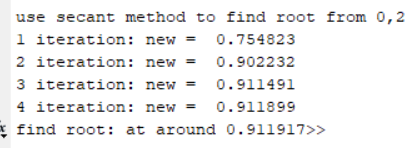
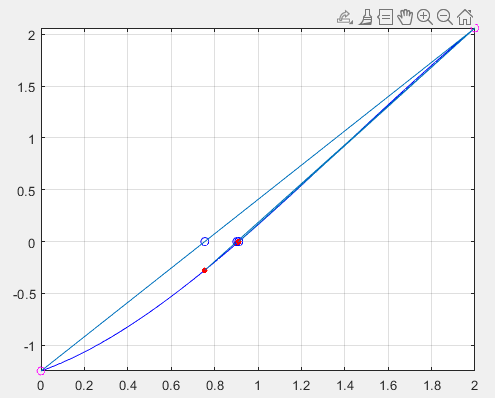
1. Secant method

we use a while loop base on f(new) to decide how many iteration to run, every time we get new point where f(new) = 0 and use new point to replace old point and we can get the two roots is -1.431807 and 0.911917



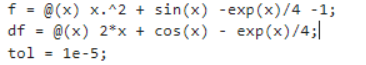


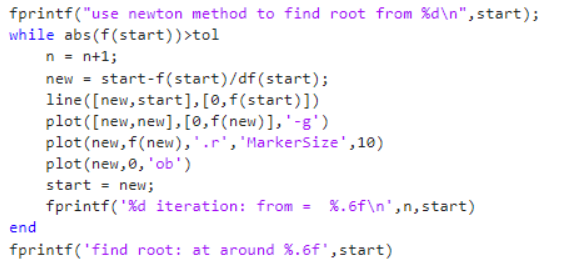
 

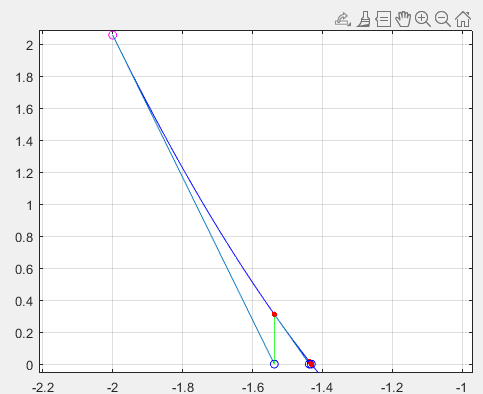
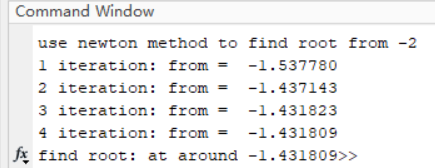
 

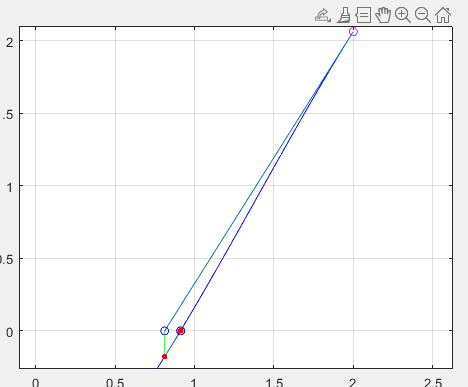
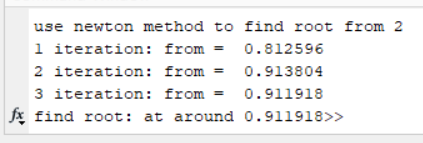
1. Newton’s method

we use a while loop base on f(start) to decide how many iteration to run, every time we get new point where by the tangent line of old point and use new point to replace old point and we can get the two roots is -1.431809 and 0.911918





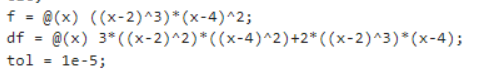


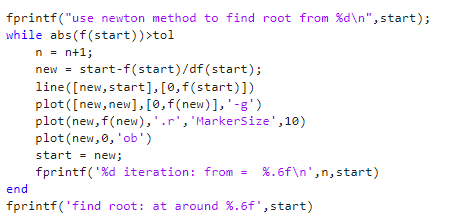


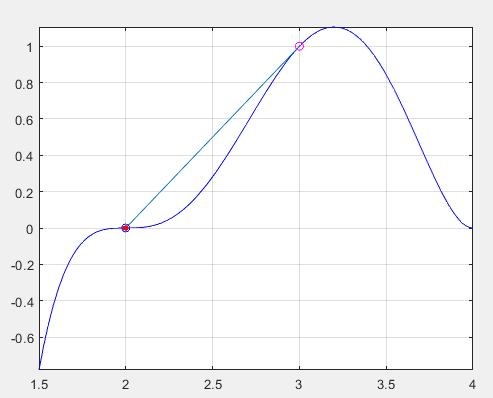
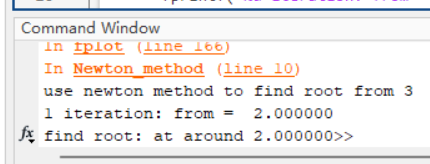
1. Use Newton’s method on the polynomial P(x), with X0 =3. I

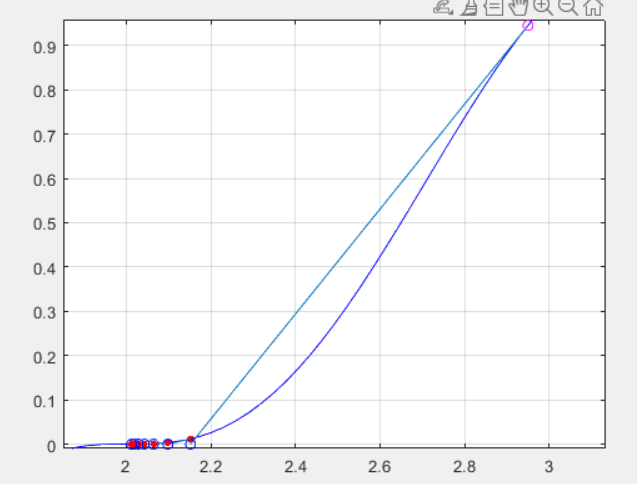
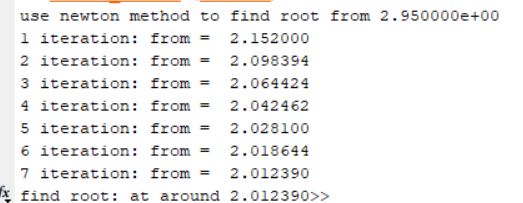
It will coverage to root at x = 2.

It is not convergence quadratic. If we start from x0 = 2.95, it takes 7 iterations to converge to R = 2. Since R = 2 is not simple root, it is triple root. f’(2) = 0 which cause f(R)f’’(R)/f’(R)^2 can’t be omitted make the convergence only linearly not quadratic.

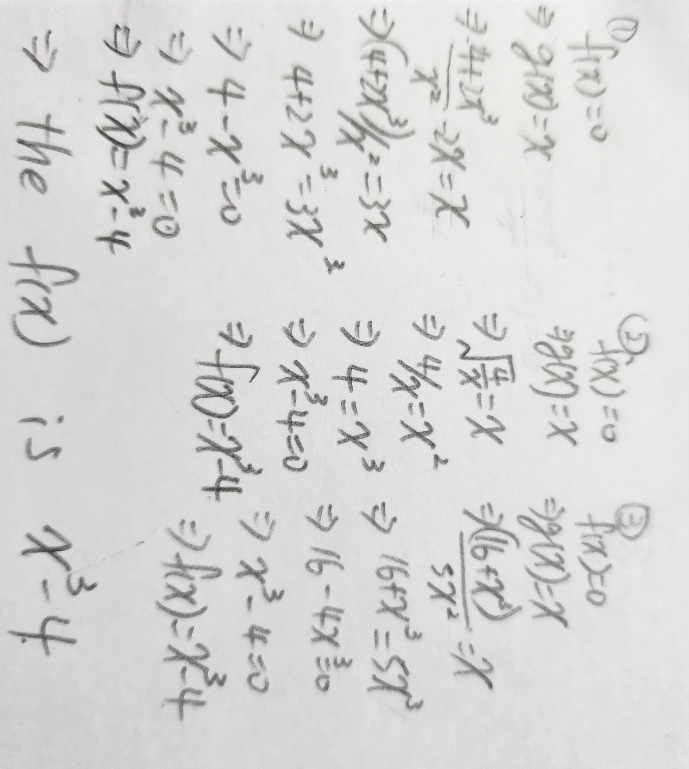


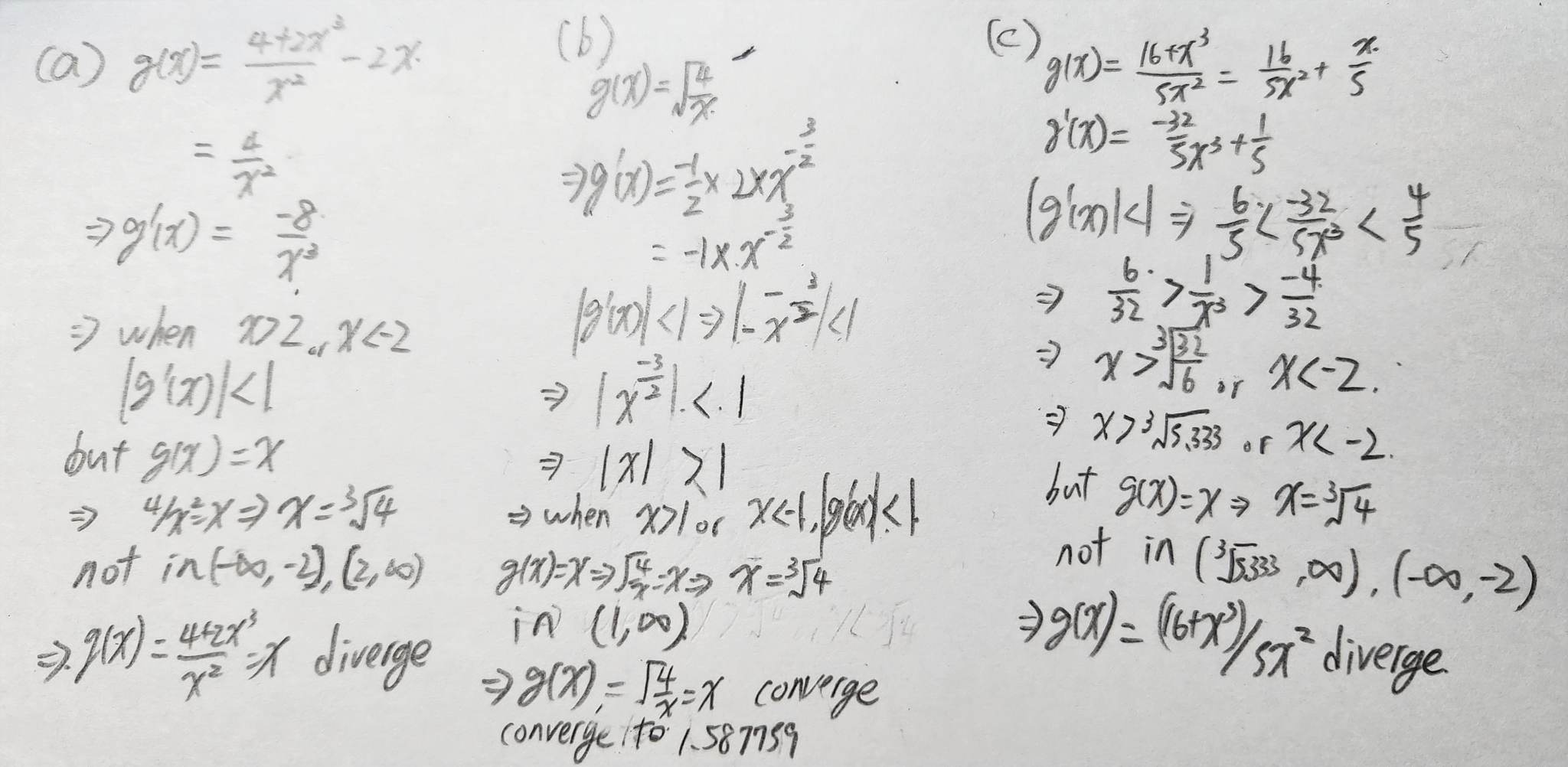


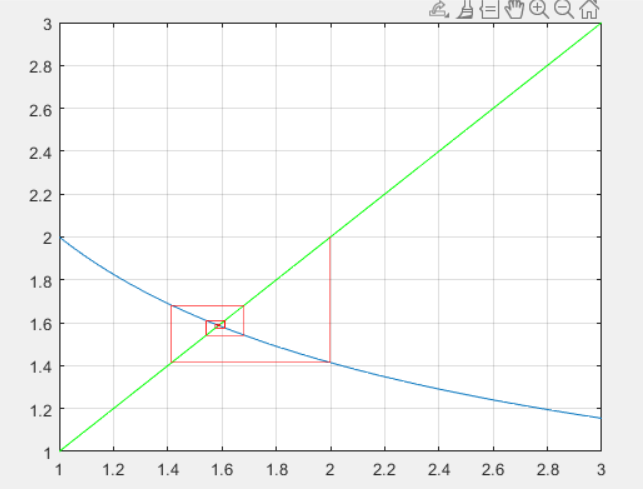
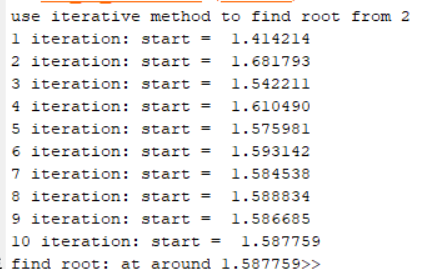


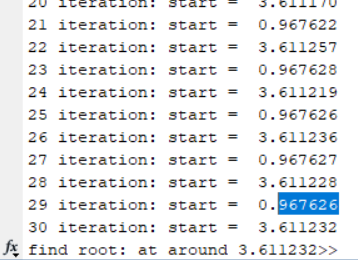
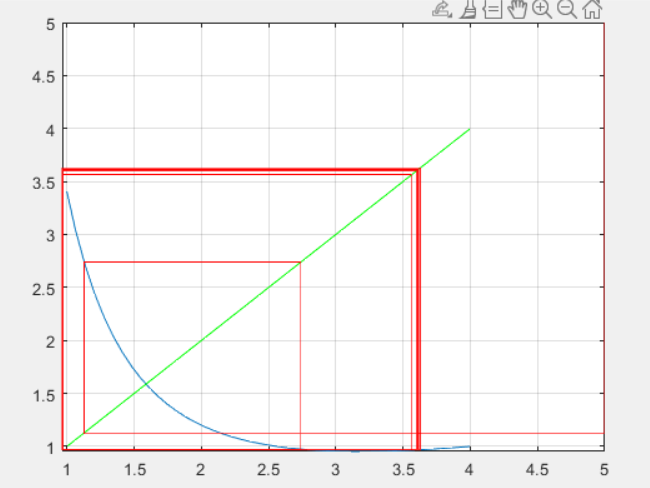


1. Below are three different g(x) functions. All are rearrangements of the same f(x). What is f(x)?





b)

c)  

we can see 0.967 and 3.611 reflect to each other, so it is also diverge.

1. Solve the following system of nonlinear equations using Newton’s method or fixed-point method.

I write a program to solve system of nonlinear equations using Newton’s method. First, I write down the matrix of system of nonlinear equations(3x1) and derived jacobian matrix j(x)(3x3).

For every iteration we calculate the s (j\*s = -f), and update the

new start point = old start point +s. Use while loop to iterate until the norm of s (the value need to update) less than tolerant error.

From 6 starting point, we get 6 solutions (4 reals,2virtual solution)

(1, 1, 1)-> (1.111408, 0.988210, 1.070878)

(1.3, 0.9, -1.2)->( 1.353748, 0.925431, -1.255968)

(100, 100, 100)->( 32.884631, -4434.086620, 115.490885)

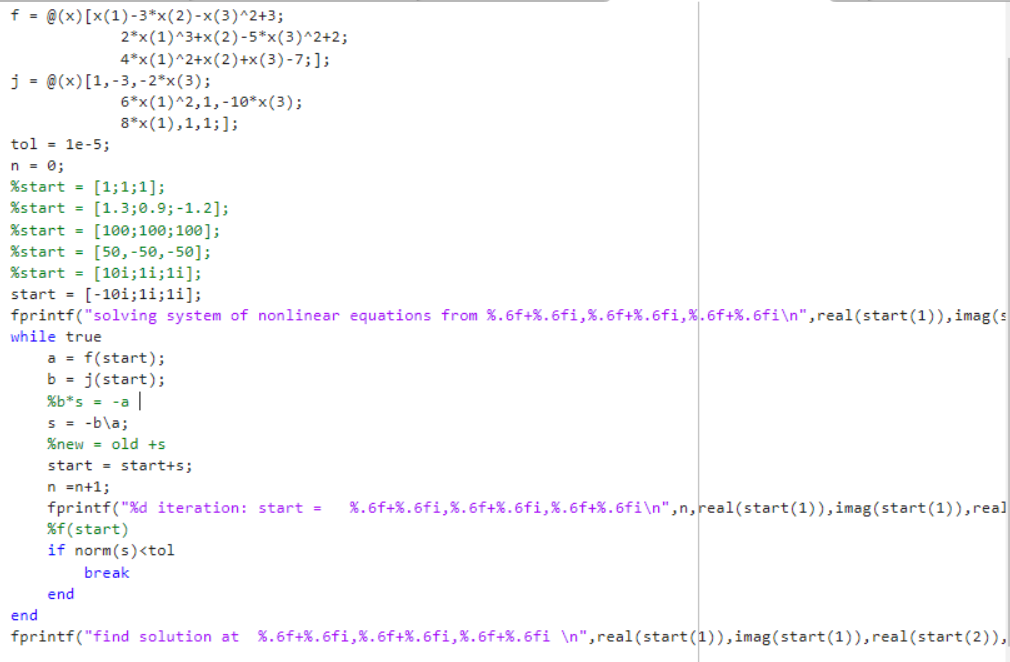
(50, -50, 50)->( 31.151405, -3768.157126, -106.482969)

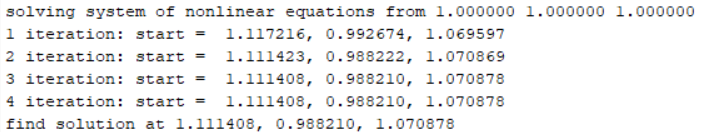
(0+10i, 0+1i, 0+1i)->

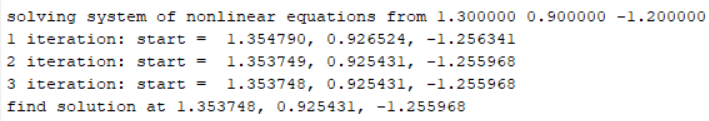
(-1.250596+0.048995i,0.665053-0.013410i,0.088588+0.503590i)

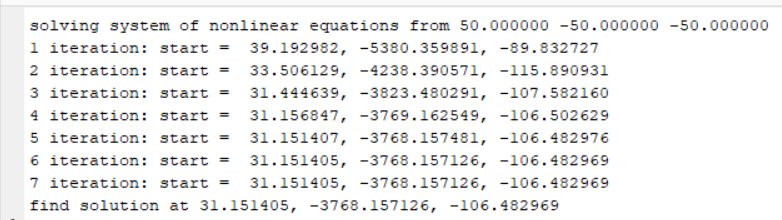
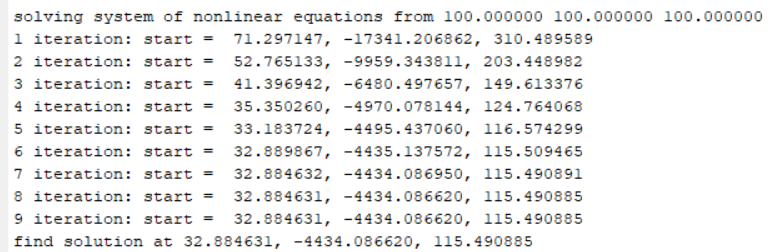
(0-10i, 1+0i, 1+0i)->

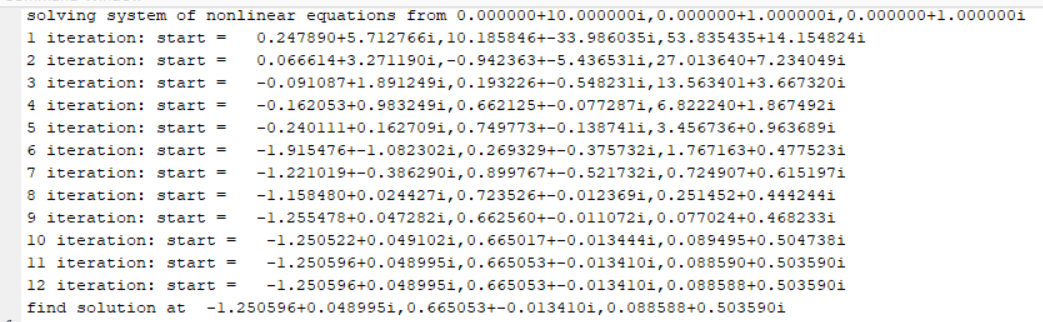
( -1.250596-0.048995i,0.665053+0.013410i,0.088588-0.503590i)

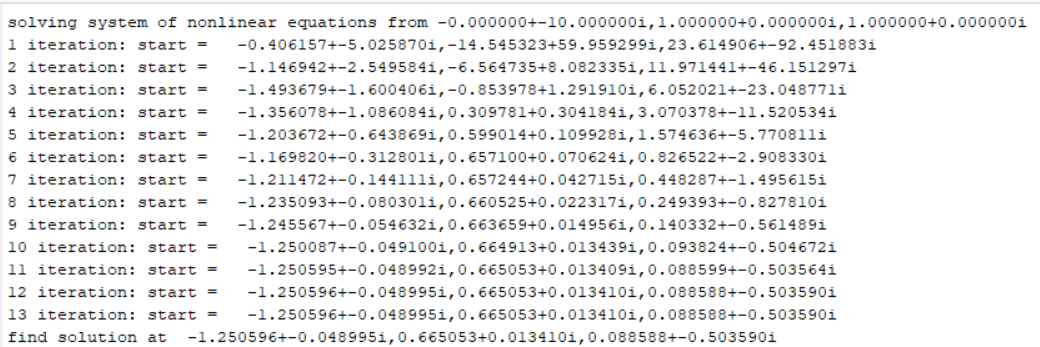












(if +- please regard as - )