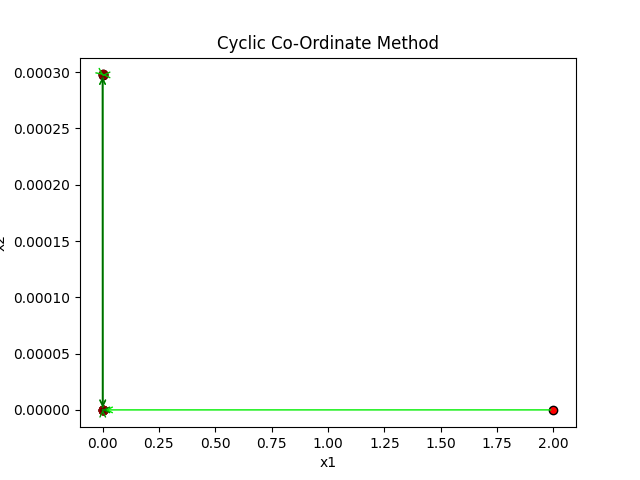
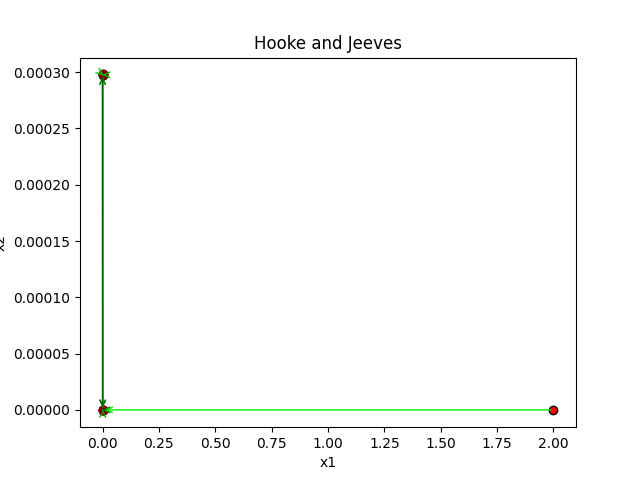
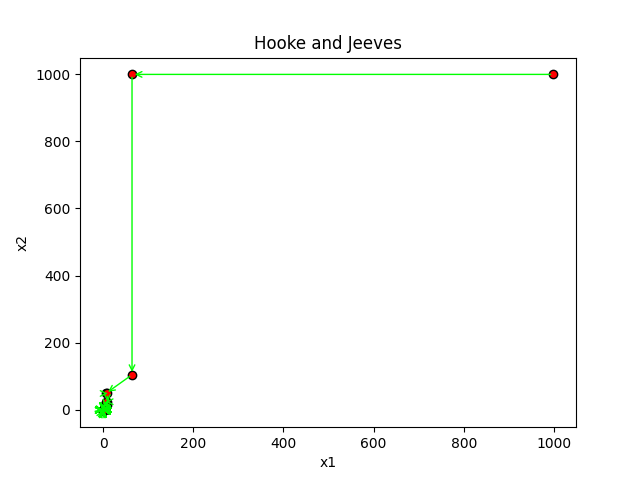
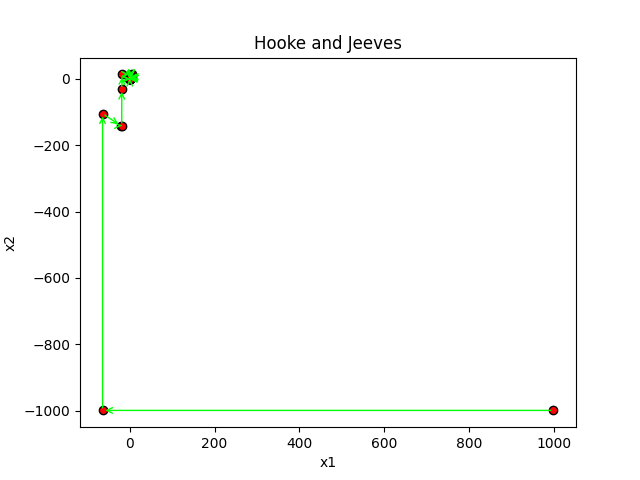
Please Note that stopping criterias:-  
1) Stopping Criteria of functional Value refers to **||∇fk||<=0.001** or **|(fk+1-fk)/fk|<=0.001** as per the differentiability of the function  
2) Stopping Criteria as the individual Dimensional values refers **to |xk+1-xk|<=0.01**

The solution images for each method is attached for the initial starting solution of (2,0)

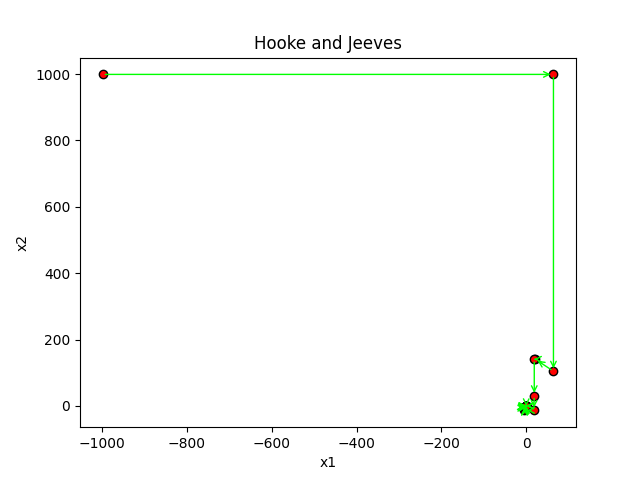




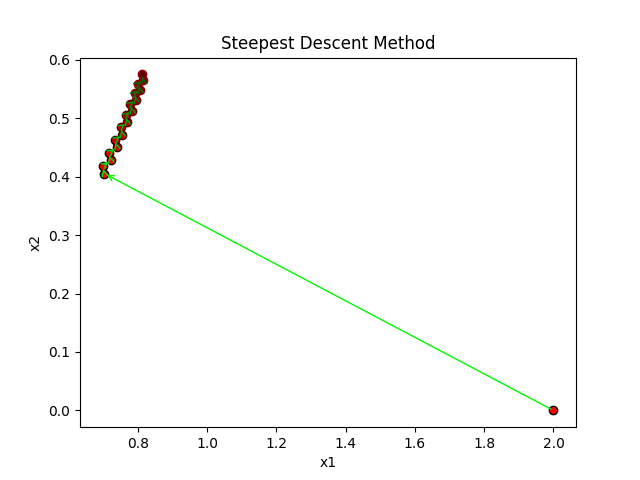
When using the starting value of (999,999) using the stopping criteria of the functional values, Hooke and Jeeves takes 680 iterations as shown below.

When using Hooke and Jeeves with Stopping Criteria as the Dimensional Value, we obtain the following results:-  
1) Starting Value (999,-999) in 9 iterations:-  


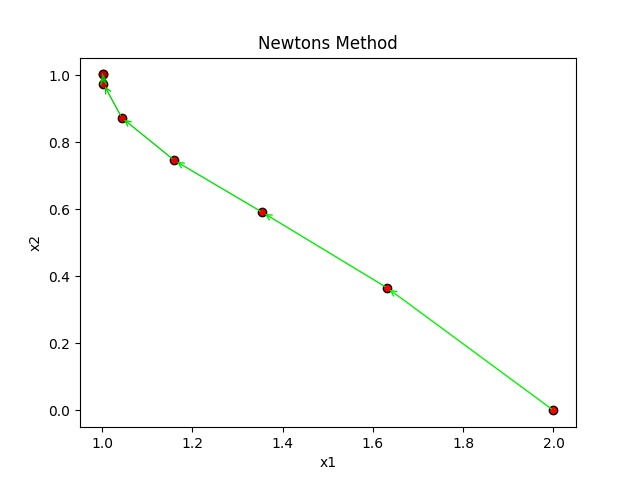
1. Starting Value (-999,999) in 9 iterations:-



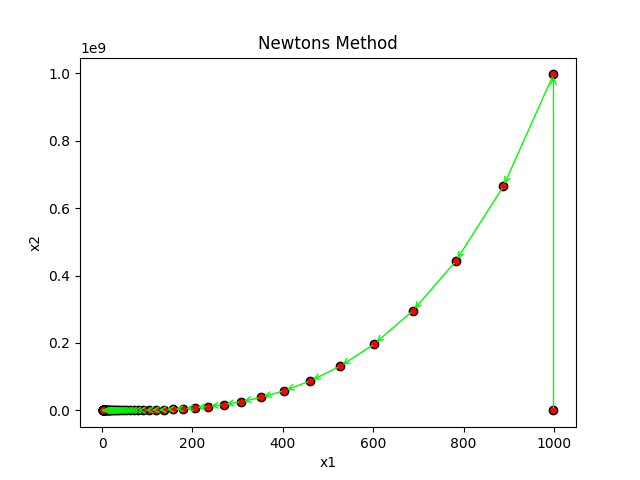
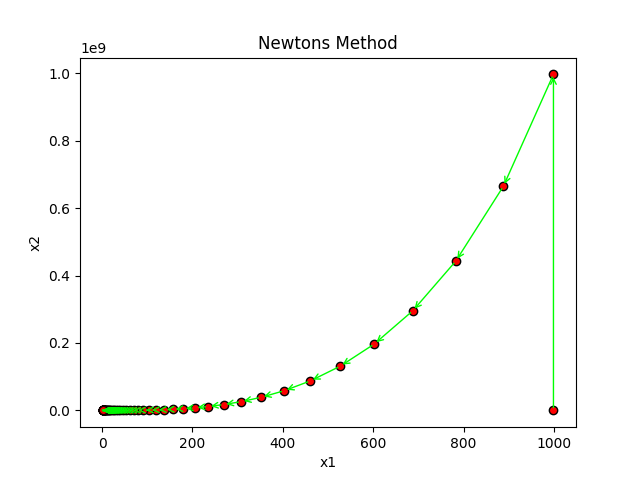
The Steepest Descent Method, with stopping criteria as the individual dimensional values, is shown below (18 iterations) :-



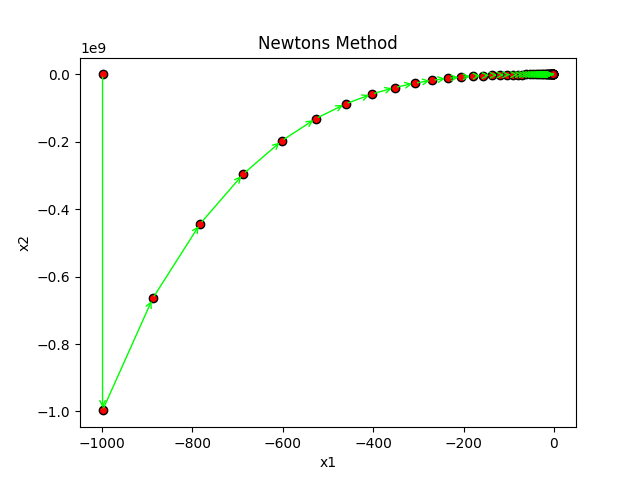
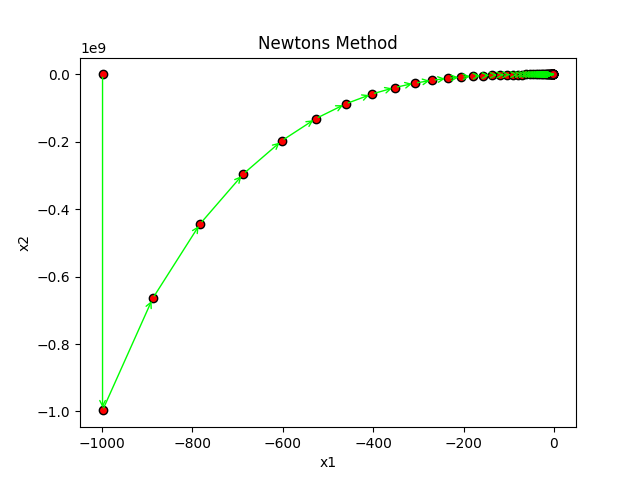
Newton’s Method when solved from (2,0), using individual dimensional stopping criteria, takes 7 iterations as shown below.



Newton’s Method when used from (999,999) gives the solution as below:

1. Stopping criteria as individual dimensions in 63 iterations:- 
2. Stopping Criteria as functional value in 62 iterations:-

Newton’s Method when used for the same problem, and starting from (-999,-999) yields the below solution:-

1. With stopping criteria as the functional value (62 iterations):-
2. With stopping criteria as the individual dimensional values (63 iterations):-

Conclusion:-

1. Steepest Descent Method is not Recommended as in the cases shown the value of Lambda seems to be very large as therefore the program doesn’t show the result
2. A new single Dimensional search is also included which does only a single computation at each step (it computes the middle value and requires the start, end and any random point in between the start and end as the initial points).

# SanBan Search

import random

def SanBan\_Method(expression,starting\_value=-999,another\_value=0,ending\_value=999,stopping\_interval\_length=0.0001):

    print("SanBan Search starts here")

    print("For SanBan search, Interval\_Start\_Value, Interval\_End\_Value and any random value between the start and end is required")

    #starting\_value=float(input("Enter the starting value of the interval: "))

    #ending\_value=float(input("Enter the ending value of the interval: "))

    if ending\_value<starting\_value:

        starting\_value=starting\_value-ending\_value

        ending\_value=starting\_value+ending\_value

        starting\_value=ending\_value-starting\_value

        print("Since the Ending\_Value should be greater than the Starting\_Value, they have been interchanged")

    if another\_value>ending\_value or another\_value<starting\_value:

        another\_value=random.random()

        another\_value=another\_value\*starting\_value+(1-another\_value)\*ending\_value

        print("The random value between Start and End has been modified")

    middle\_value=(ending\_value+starting\_value)/2

    while (ending\_value-starting\_value)>stopping\_criteria:

        if middle\_value>another\_value:

            Lambda=another\_value

            Mu=middle\_value

        else:

            Lambda=middle\_value

            Mu=another\_value

        expression\_at\_Lambda = expression.replace('x','('+str(Lambda)+')')

        expression\_at\_Mu = expression.replace('x','('+str(Mu)+')')

        val\_fn\_at\_Lambda = eval(expression\_at\_Lambda)

        val\_fn\_at\_Mu = eval(expression\_at\_Mu)

        if(val\_fn\_at\_Lambda>val\_fn\_at\_Mu):

            return SanBan\_Method(expression,starting\_value=Lambda,another\_value=Mu,ending\_value=ending\_value,stopping\_interval\_length=0.0001)

        else:

            return SanBan\_Method(expression,starting\_value=starting\_value,another\_value=Lambda,ending\_value=Mu,stopping\_interval\_length=0.0001)

    return middle\_value

1. In the case of Cyclic Co-Ordinate Search Method using Acceleration Step, new Acceleration Steps are suggested as well. For this new acceleration steps, every possible combination of +ve and -ve direction is being accounted for.

Example:- If there are 3 dimensions of x1,x2,x3; then the possible acceleration dimensions are:-  
[1,1,1] ; [1,1,-1] ; [1,-1,1] ; [-1,1,1] ; [1,-1,-1] ; [-1,1,-1] ; [-1,-1,1] ; [-1,-1,-1]