Numerical Methods
Assignment 02

Georgios Christos Chouliaras (2592496)

March 12, 2017

Exercise 1

The function

The function `Chouliaras_assignment2_exercise1(f,p1,p2,N,tol)` approximates the zero of a function `f` using the Secant method with two starting points `p1, p2`.
The input consists of a function of a real variable `f`, the two initial guesses `p1, p2` and optionally the maximum number of iteration steps `N` and the desired precision `tol`. If the user does not specify the optional inputs, then the default values are `N = 20` and `tol = 0.0001`.
The output of the function is the approximated zero `x` as well as a vector `v` which contains the approximated values in the iterations. If the function does not converge to the desired precision within the specified iterations, a message informs the user about this.
As an example, in order to approximate the zero of the function $f(x) = x^3 - 9$ with starting points $x_1 = -3$ and $x_2 = 5$, for $N = 15$ iterations and desired precision $tol = 0.001$ one should type:

```
[x,v] = Chouliaras_assignment2_exercise1(@(x) x^3 -9,-3,5,15,0.001)

x =

    2.0801


v =

  Columns 1 through 9

   -1.1053   -0.6051    3.4813    0.2834    0.9642    7.2798    1.0971    1.2206    3.0014

  Columns 10 through 14

    1.7277    1.9514    2.1058    2.0784    2.0801
```

It can be seen that the algorithm converges after 14 iterations and hence the vector `v` has length 14. Of course the number of iterations depends on the selection of the starting points x_1, x_2.

The method

The Secant method is a modification of Newton's method in which the derivative of the function is approximated by:

$$f'(p_{n-1}) \approx \frac{f(p_{n-1})(p_{n-1} - p_{n-2})}{f(p_{n-1}) - f(p_{n-2})}$$

By plugging this approximation of the derivative into Newton's formula, we get the iterative scheme for the Secant method as follows:

$$p_n = p_{n-1} - \frac{f(p_{n-1})(p_{n-1} - p_{n-2})}{f(p_{n-1}) - f(p_{n-2})}$$

The Secant method works in the following way: We start with two initial approximations p_1 and p_2. Then the next approximation is the x-intercept of the line joining $(p_1, f(p_1))$ and $(p_2, f(p_2))$. This is repeated until the approximated root is within the desired precision, or until the number of iterations reaches the specified maximum N. It needs to be noted that as a measure for the desired precision, the f value at the root was chosen. In other words, the algorithm stops when $|f(x)| < tol$, where x is the approximated root and tol is the desired precision (by default it is 10^{-4}).

Discussion

In order to check the accuracy of the function, the error can be estimated. For example, the real root of the function $f(x) = x^3 - 9$ is $x_{real} = 3^{2/3}$ hence the error in this case is:

$$error = |x_{approx} - x_{real}| = 2.012779160009970 \cdot 10^{-5}$$

In this case the precision was $tol = 0.001$. It should be noted that the error is always less than or equal to the desired precision, if the algorithm converge smoothly.

In order to estimate the order of convergence, we can use the following formula:

$$p \approx \frac{ln|\frac{x_{n+1} - \alpha}{x_n - \alpha}|}{ln|\frac{x_n - \alpha}{x_{n-1} - \alpha}|} \tag{1}$$

where p is the order of convergence and α is the real root. For the function $f(x) = x^3 - 9$ the order of convergence equals $p = 1.6007$ which is approximately equal to the golder ratio $\frac{1+\sqrt{5}}{2}$. In other words the convergence is superlinear, but not quite quadratic. It should be noted that the same order was also calculated for most of the functions that were tested. Having in mind that the order of convergence for the Newton method is $p = 2$ we can conclude that the convergence for the Secant method is slower. To see this in practice, the root of the function $f(x) = x^3 - 9$ was also approximated using the Newton's method (with $x_0 = 5$) which converged in 6 (instead of 14) iterations. However, since the Secant method requires only one function evaluation per step, while Newton requires 2, sometimes the convergence of the Secant method can be occasionally faster in practice. As an example, assuming that the evaluation of f takes the same time as the evaluation of the derivative (without taking into account other costs) two steps can be done by Secant method at the same time as one of Newton's method and in this sense the Secant method is faster. However, Newton's can be considered faster in time if there is parallel evaluation of the function and its derivative, even if the number of steps is larger.

In order to conduct a comparison between the Secant and the Newton method, the advantages and disadvantages of the Secant method compared to Newton are provided below.

Advantages of the Secant method:

1. Since the Secant method uses the approximation of the derivative, there is no need to know the derivative of the function in order to approximate its root. In Newton's method the derivative needs to be known, something that is very difficult in many cases.
2. In the Secant method, only one function evaluation is needed per step. In other words, in every step only $f(x)$ needs to be evaluated, while in Newton's method both $f(x)$ and the derivative $f'(x)$ need to be evaluated.

Disadvantages of the Secant method:

1. The order of convergence for the Secant method is slightly lower than the Newton's order, hence the convergence of the Secant method is slower.
2. The Secant method requires two initial points, and in the case that they are not appropriate the function might not converge. This is more likely to happen in the Secant method than in Newton's method where only one initial point needs to be specified.

Exercise 2

The function

The function `Chouliaras_assignment2_exercise2(f,df,x0,tol,nmax)` determines the zero of a function using the Modified Newton's method.
The inputs are the function `f`, its derivative `df`, an initial point `x0` and optionally the desired precision `tol` and the maximum number of iterations `nmax`. If `tol` and `nmax` are not specified, the function assigns default values 0.0001 and 1000 respectively.
The outputs are the approximated zero `x`, the estimate of the order of the zero `m` and a vector with the approximated zeros `iter`.
If the derivative of the function is 0, the function stops and a message informs the user. Moreover, the function stops when the desired precision is achieved, or the number of iterations reaches the maximum (`nmax`). If the algorithm stops without achieving the desired precision, a message informs the user about this.
As an example, in order to approximate the zero of the function $f(x) = (x-3)^4$, for which $df = 4(x-3)^3$ with starting point $x_0 = 10$, tolerance and iterations equal to the default, one should type:

```
>> [x, m , iter] = Chouliaras_assignment2_exercise2(@(x) (x-3)^4, @(x) 4*(x-3)^3, 6)
```

```
x =

     3


m =

     4


iter =

    6.0000    5.2500    4.6875    3.0000
```

As it can be seen, the algorithm converged in 4 iterations to the correct zero and the correct order of the zero. The multiplicity of a zero x is the least positive integer m such that $f^{(m)}(x) \neq 0$ In this case $f^{(4)}(3) \neq 0$ while all the smaller derivatives are zero, hence $m = 4$ is the order of the zero and the function estimated it accurately.

The method

Since the order of the zero is not given as an input by the user, it needs to be estimated. We know that close to the real root α it holds $f(x) \approx C(x-\alpha)^m$ and $f'(x) \approx Cm(x-\alpha)^{m-1}$ and hence by dividing them we have:

$$\frac{f(x_n)}{f'(x_n)} \approx \frac{(x_n-\alpha)^m}{m(x_n-\alpha)^{m-1}} = \frac{x_n-\alpha}{m} \tag{2}$$

Furthermore, the modified Newton method is given by:

$$x_{n+1} = x_n - M\frac{f(x_n)}{f'(x_n)} \tag{3}$$

Now applying (2) in (3) we have:

$$x_{n+1} = x_n - \frac{M}{m}(x_n-\alpha) \tag{4}$$

Similarly, for x_n holds:

$$x_n = x_{n-1} - \frac{M}{m}(x_{n-1}-\alpha) \tag{5}$$

By subtracting (5) from (4) we have:

$$x_{n+1} - x_n = x_n - x_{n-1} - \frac{M}{m}(x_n - \alpha) + \frac{M}{m}(x_{n-1} - \alpha) \Leftrightarrow$$
$$x_{n+1} - 2x_n + x_{n-1} = \frac{M}{m}(x_{n-1} - x_n) \Leftrightarrow$$
$$m = \frac{M(x_{n-1} - x_n)}{x_{n+1} - 2x_n + x_{n-1}} \tag{6}$$

Next, we assign an initial value to M ($M = 1$) and we use the modified Newton's method until we have 3 estimates of the root. Then, using equation (6) we estimate $M = m$ and we use this new M in the iterative scheme of the modified Newton. The procedure continues until the maximum number of iterations is reached, or the algorithm achieves the required precision.

Discussion

It needs to be noted, that if the initial point is very close to the real zero, the function results in a false estimation of the order m, since the algorithm does not repeats the necessary number of iterations for m to be approximated correctly. Moreover, if the initial value x_0 is chosen very far from the real zero, there is also the possibility for the function not to converge.

In order to investigate the order of convergence for modified Newton's method many functions were tested and the order was calculated using (1). An example can be seen below for $f(x) = (x-2)^2(x+1)^3$ for which $m = 3$.

```
>> [x, m , iter] = Chouliaras_assignment2_exercise2(f,df,65,0.00001,1000)

x =

   -0.9999


m =

    3.0354


iter =

  Columns 1 through 9

   65.0000   52.0467   41.6858    0.2800   -3.9512    0.1634   -3.5727   -1.6600   -1.0871

  Columns 10 through 12

   -0.9684   -1.0126   -0.9999
```

For this case the order of convergence was calculated $p = 0.9078$ using formula (1). A plot of the estimations of m from the initial value until convergence, can be seen in figure (1). It should be noted that in the figure the iterations in the x-axis, are not the same as the number of iterations for the root, since for each estimation of m, 3 approximations of the zero are needed as it can be seen in equation (6).

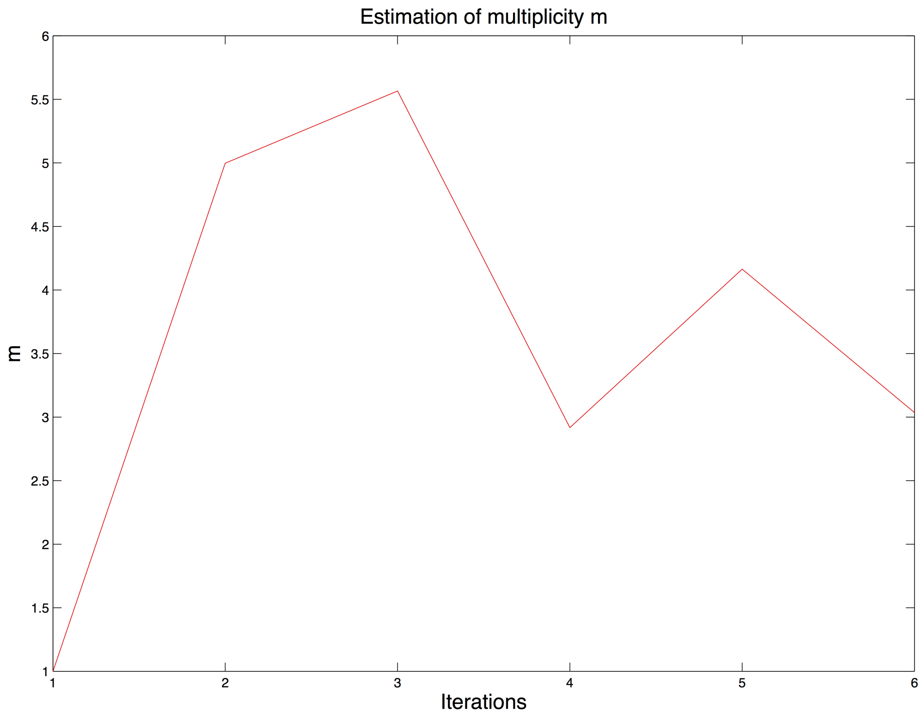

Figure 1: Convergence of the order of the zero m for $f(x) = (x-2)^2(x+1)^3$ after 6 estimations.

From the functions that were tested it was found that the order of convergence for modified Newton's is not constant but varies. More specifically, most of the functions had order of convergence between 1 and 2.

In order to compare the Modified Newton's method with the classical Newton, we attempt to approximate the same function, with both methods. The function to be approximated is again $f(x) = (x-3)^4$ with starting point $x_0 = 10$. The results of both methods are given below.

```
>> [zero,res,niter]=newton(@(x) (x-3)^4, @(x) 4*(x-3)^3,10,0.0001,1000)

zero =

    3.0003


res =

   7.7441e-15
```

```
niter =

    35


>>  [x, m , iter] = Chouliaras_assignment2_exercise2(@(x) (x-3)^4, @(x) 4*(x-3)^3,10)

x =

     3


m =

     4


iter =

   10.0000    8.2500    6.9375    3.0000
```

As it can be seen, the classical Newton method converged after 35 iterations, while the Modified Newton only in 4. This is a significant difference in the algorithm's speed. However, this is not always the case. In some cases, the classical Newton's method converges faster. As an example consider the function $f(x) = x^2 - 531$, with $df(x) = 2x$ and initial point $x_0 = 10$ for which the multiplicity of the zero is $m = 1$. The results of both methods are given below.

```
>> [x, m , iter] = Chouliaras_assignment2_exercise2(@(x)x^2-531,@(x) 2*x,10)

x =

   23.0434


m =

    1.0017


iter =

   10.0000   31.5500   24.1902   23.3556   23.1245   23.0411   23.0435   23.0434

>> [zero,res,niter]=newton(@(x)x^2-531,@(x) 2*x,10,0.0001,1000)

zero =

   23.0434
```

```
res =

   2.5648e-10


niter =

     5
```

As it can be seen, in this case the classical Newton method converged in the correct zero after 5 iterations, while the modified Newton after 8 iterations, however the difference is not that significant in this case. At this point it needs to be noted, that several other functions of multiplicity 1 were tested, and in every case the Newton method converged in less iterations. However, for functions with multiplicity larger than 1, the Modified Newton's converged faster.

Exercise 3

The function

The function `Chouliaras_assignment2_exercise3(V, discr, nsteps)` applies Newton's method to the complex function $f(z) = z^3 - 1$ and produces a graph of a rectangle in the complex plane, in which each point has a different color depending on which of the three zeros of the function it is converging to.

The input is a vector of two complex numbers `V`, the number of discretization steps in both directions `discr` and optionally, the number of iteration steps in the Newton's method `nsteps`. If this value is not given, the function assigns a default `nsteps = 1000`.

Furthermore the function returns error in case that the input is invalid. This can happen if the user inputs vectors with length different than 2, or if both numbers are real. It also returns an error if the two numbers are the same, since in this case a rectangle cannot be formed.

As an example, in order to produce a picture of the rectangle which is formed by `V = [-3-3i, 3+ 3i]` and have 600 discretization steps in each direction on should type:

```
>> Chouliaras_assignment2_exercise3([(-3-3i),(3+3i)], [600,600])
```

Illustration

The function creates a grid with points z_0 which constitute the initial values for the Newton's method. Next, this matrix of initial values is plugged in into Newton's method and the zeros are calculated. For the Newton's method a precision $tol = 10^{-5}$ is used. Then, each z_0 is plotted with a different color, depending on which of the three zeros it converges. The points z_0 which converge to the root $z = 1$ are plotted in red, those that converge to $z = -\frac{1}{2} - \frac{\sqrt{3}}{2}$ are plotted in green and those that converge to $z = -\frac{1}{2} + \frac{\sqrt{3}}{2}$ are plotted in blue. In figure (2) can be seen the picture created for given vector `V = [-3-3i, 3 + 3i]` and 600 discretization steps in both directions.

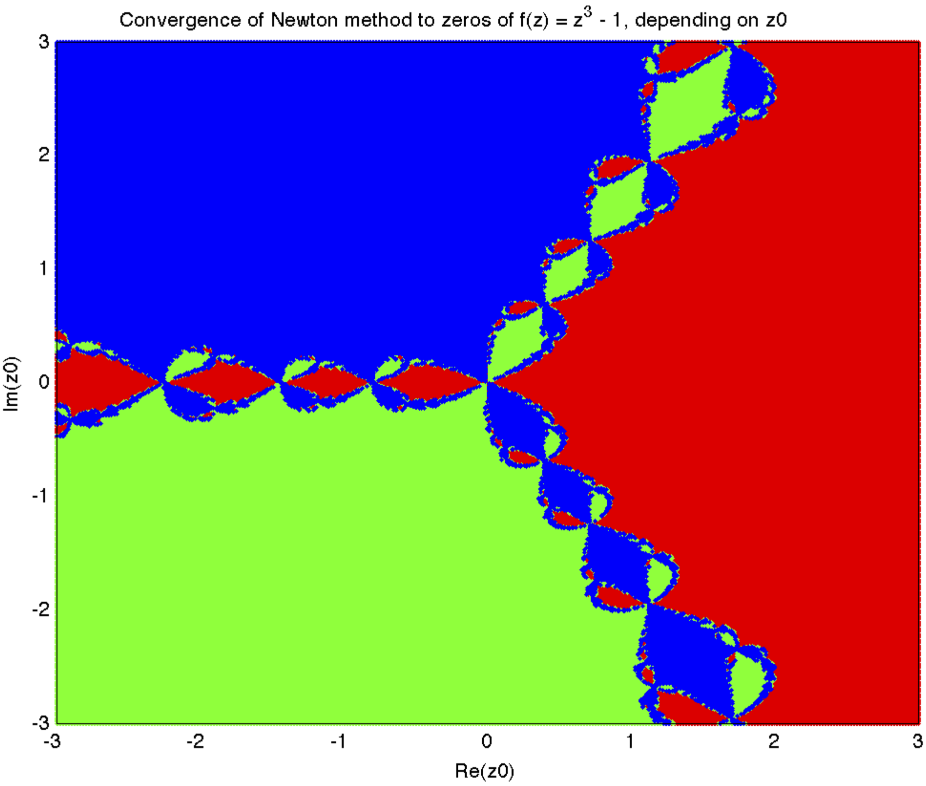

Figure 2: Convergence of Newton's method to zeros of $f(z) = z^3 - 1$, depending on z_0. Red is for $z = 1$, green is for $z = -\frac{1}{2} - \frac{\sqrt{3}}{2}$ and blue for $z = -\frac{1}{2} + \frac{\sqrt{3}}{2}$. The input vector is $V = [-3 - 3i, 3 + 3i]$ and the number of discretization steps is $discr = [600, 600]$

In the picture we can observe that there is a repeated pattern for the convergence of the points. It needs to be noted that in order to get this form of the fractals in the picture , both the real and the imaginary parts of the input numbers must have opposite signs. In the case that the real parts and/or the imaginary parts have the same sign, the fractals are still visible in some cases but not in the same form as in figure 2.