

Assignment 1 — Roots with Multiplicity

2019862

Semester 1, 2015–2016

The purpose of this assignment is to devise and implement a modified version of the Newton-Raphson method for finding roots with multiplicity. In order to proceed, let us first give a precise definition of a root of multiplicity m .

Definition 1. *The number a is called a **root (or zero) of multiplicity m** of the function $f(x)$ if*

$$f(a) = f'(a) = \dots = f^{(m-1)}(a) = 0, \quad \text{but} \quad f^{(m)}(a) \neq 0. \quad (1)$$

The assignment consists of 4 major questions with several sub-parts. The solutions are given in the order of the questions. For ease of the reader, the questions themselves are included before each solution is presented.

Question 1. *Let a be the root of multiplicity m of $f(x)$. Use Taylor's Theorem to show that $f(x)$ can be represented in the form $f(x) = (x - a)^m q(x)$, where $q(a) \neq 0$.*

Solution. Using Taylor's theorem to expand around the point a ,

$$\begin{aligned} f(x) &= f((x - a) + a) = f(a) + f'(a)(x - a) + \frac{1}{2!}f''(a)(x - a)^2 + \dots \\ &\quad \dots + \frac{1}{m!}f^{(m)}(a)(x - a)^m + \frac{1}{(m+1)!}f^{(m+1)}(\xi)(x - a)^{m+1}, \end{aligned}$$

where $\xi \in (x, a)$. But since $x = a$ is a root of multiplicity m of the function $f(x)$, utilizing **Definition 1** all derivatives up to the m -th one vanish, i.e.

$$f(a) = f'(a) = \dots = f^{(m-1)}(a) = 0.$$

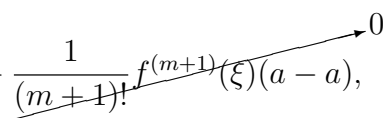
Taking this into account, the Taylor expansion of $f(x)$ around a becomes

$$\begin{aligned} f(x) &= \frac{1}{m!}f^{(m)}(a)(x - a)^m + \frac{1}{(m+1)!}f^{(m+1)}(\xi)(x - a)^{m+1} \\ &= (x - a)^m \left[\frac{1}{m!}f^{(m)}(a) + \frac{1}{(m+1)!}f^{(m+1)}(\xi)(x - a) \right]. \end{aligned} \quad (2)$$

Notice that Equation 2 is of the form

$$f(x) = (x - a)^m q(x), \quad \text{where} \quad q(x) = \left[\frac{1}{m!}f^{(m)}(a) + \frac{1}{(m+1)!}f^{(m+1)}(\xi)(x - a) \right].$$

Now check that $q(a) \neq 0$. Evaluate $q(x)$ at $x = a$, which yields

$$q(a) = \frac{1}{m!}f^{(m)}(a) + \frac{1}{(m+1)!}f^{(m+1)}(\xi)(a-a),$$


as $a - a = 0$, and by **Definition 1** the m -th derivative is non-zero, so indeed

$$q(a) = \frac{1}{m!}f^{(m)}(a) \neq 0,$$

as required. □

Question 2. Consider the function $f(x) = \frac{(x-1)^2}{x}$ on the interval $x \in (0, \infty)$.

(a) Show that the equation $f(x) = 0$ has a root $a = 1$ of multiplicity 2. Make a plot of $f(x)$ to confirm this fact visually.

Solution. Differentiate once in order to obtain the first derivative of the function $f(x)$.

$$f'(x) = \frac{(2x+2)x - x^2 + 2x - 1}{x^2} = \frac{x^2 - \cancel{2x} + \cancel{2x} - 1}{x^2} = \frac{x^2 - 1}{x^2}.$$

Now differentiate once again to obtain the second derivative of the function $f(x)$.

$$f''(x) = \frac{(2x)x^2 - (x^2 - 1)2x}{x^4} = \frac{\cancel{2x^3} - \cancel{2x^3} + 2x}{x^4} = \frac{2}{x^3}.$$

Now evaluate both derivatives at the given point $a = 1$ to obtain

$$\begin{aligned} f'(1) &= \frac{1-1}{1} = 0, \\ f''(1) &= \frac{2}{1} = 2 \neq 0. \end{aligned}$$

Hence, when evaluated at $a = 1$ the first non-zero derivative of the function $g(x)$ is the second derivative, and by **Definition 1**, the root a is of multiplicity 2.

The figure below is a plot of the function $f(x)$, which shows the root $a = 1$ is of multiplicity 2. □

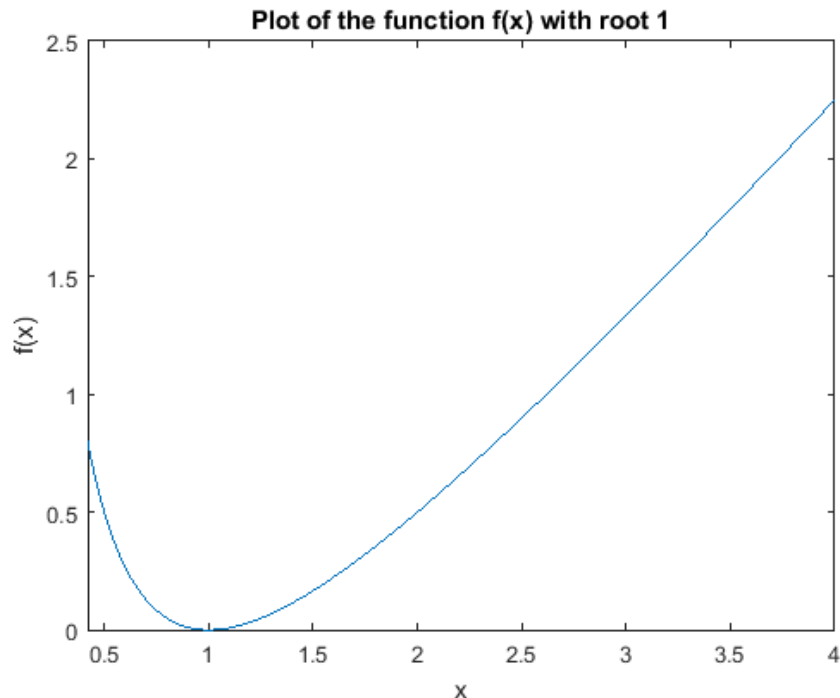


Figure 1: Plot of $f(x)$.

(b) Show analytically that the Newton-Raphson method with initial guess $x_0 = \frac{1}{2}$ will converge to $a = 1$.

Solution. By **Remark 1.34** from the Lecture Notes, the Newton-Raphson method is defined as the map

$$g(x_n) = x_n - \frac{f(x_n)}{f'(x_n)}. \quad (3)$$

Using the given function, the Newton-Raphson method is

$$\begin{aligned} g(x_n) &= x_n - \frac{(x_n - 1)^2}{\frac{x_n}{(x_n^2 - 1)}} = x_n - \frac{\cancel{(x_n - 1)}(x_n - 1)x_n^2}{x_n \cancel{(x_n - 1)}(x_n + 1)} \\ &= x_n - \frac{(x_n - 1)x_n}{(x_n + 1)} = \frac{\cancel{x_n^2} + x_n - \cancel{x_n^2} + x_n}{x_n + 1} \\ &= \frac{2x_n}{x_n + 1}. \end{aligned}$$

In order for the map to converge, it must be a contraction mapping. By **Definition 1.18** from the Lecture Notes, the map $g(x)$ is a contraction map in a region D if it satisfies a Lipschitz condition in D with a Lipschitz constant $L < 1$, that is if

$$|g(x) - g(y)| < |x - y|.$$

Rearranging the inequality we get

$$\frac{|g(x) - g(y)|}{|x - y|} < 1. \quad (4)$$

Now, the left hand side of Inequality 4 is the precise definition of the derivative of the function $g(x)$, which was covered in the *3H: Analysis of Integration and Differentiation* course. Hence, the condition for $g(x)$ to be a contraction is

$$|g'(x_n)| < 1. \quad (5)$$

Differentiate the expression for the map once to obtain its first derivative, that is

$$g'(x_n) = \frac{2(x_n + 1) - 2x_n}{(x_n + 1)^2} = \frac{\cancel{2x_n} + 2 - \cancel{2x_n}}{(x_n + 1)^2} = \frac{2}{(x_n + 1)^2}.$$

Substitute the derived expression in Inequality 5

$$|g'(x_n)| = \left| \frac{2}{(x_n + 1)^2} \right| < 1 \quad \therefore \quad -1 < \frac{2}{(x_n + 1)^2} < 1.$$

Solving for the LHS inequality,

$$-(x_n + 1)^2 - 2 < 0 \quad \therefore \quad x_n \in (-\infty, -1 - i\sqrt{2}) \cup (-1 + i\sqrt{2}, \infty)$$

we obtain complex solutions, which are of no interest in this case. On the other hand, solving the RHS inequality

$$\frac{2}{(x_n + 1)^2} < 1.$$

Since the term in the denominator, namely $(x_n + 1)^2 > 0$ is strictly greater than zero, we can rearrange the RHS inequality to obtain

$$\begin{aligned} 2 &< (x_n + 1)^2 \\ 2 &< x_n^2 + 2x_n + 1 \quad \therefore \quad x_n^2 + 2x_n - 1 > 0. \end{aligned}$$

By solving the inequality we obtain two separate intervals

$$x_n \in (-\infty, -\sqrt{2} - 1) \cup (\sqrt{2} - 1, \infty).$$

However, since $x \in (0, \infty)$ as stated in the question description, we can neglect the negative solutions of the inequality and should only consider the values of $x_n \in (\sqrt{2} - 1, \infty)$. According to MATLAB, $\sqrt{2} - 1 \approx 0.4142$. Hence, the map $g(x)$ is a contraction map for any $x > \sqrt{2} - 1$. Now, the given initial guess $x_0 = \frac{1}{2}$ belongs to the interval, i.e.

$$x_0 = 1/2 \in (\sqrt{2} - 1, \infty),$$

so by the **Banach Fixed-Point Theorem**, that is **Claim 1.19** from the Lecture notes, $g(x)$ with initial guess $x_0 = \frac{1}{2}$ will converge to the root $a = 1$. □

- (c) Show analytically that the Newton-Raphson method has a linear order of convergence in this case.

Solution. Consider the following two different ways of showing the required linear order of convergence.

1. By **Definition 1.29** from the Lecture Notes, the order of convergence α of a method is defined as

$$\lim_{n \rightarrow \infty} \left| \frac{e_{n+1}}{e_n^\alpha} \right| = C \neq 0, \quad (6)$$

where $e_n = x_n - x^*$ is the absolute error, and x^* is the fixed point of the map, in this case $x^* = a = 1$. Now consider the absolute error on the $(n+1)$ -th step, that is

$$e_{n+1} = x_{n+1} - x^* = \frac{2x_n}{x_n + 1} - 1 = \frac{2x_n - x_n - 1}{x_n + 1} = \frac{x_n - 1}{x_n + 1}.$$

Further, the error on the n -th step is

$$e_n = x_n - x^* = x_n - 1.$$

Using substituting the two expressions in Equation 6 yields

$$\lim_{n \rightarrow \infty} \left| \frac{e_{n+1}}{e_n^\alpha} \right| = \lim_{n \rightarrow \infty} \left| \frac{\frac{(x_n-1)}{(x_n+1)}}{(x_n-1)^\alpha} \right| = \lim_{n \rightarrow \infty} \left| \frac{(x_n-1)}{(x_n+1)(x_n-1)^\alpha} \right| = K \neq 0,$$

where K is some non zero constant. Now, consider two separate cases:

- $\alpha = 1$. Then

$$\lim_{n \rightarrow \infty} \left| \frac{\cancel{(x_n-1)}}{(x_n+1)\cancel{(x_n-1)}} \right| = \lim_{n \rightarrow \infty} \left| \frac{1}{x_n+1} \right|$$

and since $x_n \rightarrow 1$ as $n \rightarrow \infty$,

$$\lim_{n \rightarrow \infty} \left| \frac{1}{x_n+1} \right| = \frac{1}{2} = k \neq 0,$$

is satisfied, i.e. $\alpha = 1$ is the order of convergence.

- $\alpha \geq 2$. Then

$$\left| \frac{(x_n-1)}{(x_n+1)(x_n-1)^\alpha} \right| \rightarrow \infty,$$

as $x_n \rightarrow 1$ and the denominator becomes zero, so $\alpha \not\geq 2$, i.e. $\alpha = 1$.

2. Assuming that the first $\alpha - 1$ derivatives of $g(x)$ are 0, and using Taylor expansion,

$$\begin{aligned} g(x) &= g((x-1)+1) = g(1) + (x-1)g'(1) + \frac{(x-1)^2}{2}g''(1) + \dots \\ &\quad \dots + \frac{(x-1)^\alpha}{\alpha!}g^{(\alpha)}(1) + \frac{(x-1)^{\alpha+1}}{(\alpha+1)!}g^{(\alpha+1)}(\xi) \end{aligned}$$

where $\xi \in (x, 1)$. That is,

$$g(x) = 1 + \frac{1}{\alpha!}(x-1)^\alpha g^{(\alpha)}(1) + \frac{1}{(\alpha+1)!}(x-1)^{\alpha+1} g^{(\alpha+1)}(\xi).$$

Evaluate at $x = x_n$ and use the definition of the absolute error $e_n = x_n - 1$ to get

$$e_{n+1} = \frac{1}{\alpha!} e_n^\alpha g^{(\alpha)}(1) + \frac{1}{(\alpha+1)!} e_n^{\alpha+1} g^{(\alpha+1)}(\xi).$$

Now divide by e_n^α , yielding

$$\frac{e_{n+1}}{e_n^\alpha} = \frac{1}{\alpha!} g^{(\alpha)}(1) + \frac{1}{(\alpha+1)!} e_n g^{(\alpha+1)}(\xi).$$

As $n \rightarrow \infty$, the second term becomes negligibly small, i.e. tends to 0, so evaluating the limit

$$\lim_{n \rightarrow \infty} \left| \frac{e_{n+1}}{e_n^\alpha} \right| = \frac{1}{\alpha!} g^{(\alpha)}(1) \neq 0.$$

So given the expression for $g(x)$, we obtain the first and second derivatives, to make sure we derive the desired result. That is

$$g'(x_n) = \frac{2}{(x_n + 1)^2}, \quad g''(x_n) = -\frac{4}{(x_n + 1)^3}.$$

Evaluate at $x_n \rightarrow 1$ to get

$$g'(1) = \frac{2}{(1 + 1)^2} = \frac{1}{2},$$

which in turn is the first non-zero derivative of $g(x)$, so it is unnecessary to evaluate the second derivative. Therefore, by **Claim 1.30** from the Lecture Notes, $g(x)$ is an order 1 method, that is, it is linear.

□

- (d) Use the Newton-Raphson code provided below to solve $f(x) = 0$ with $x_0 = \frac{1}{2}$. Count the actual number of iterations needed to achieve an accuracy of 15 decimal places. Plot the sequence produced by the Newton-Raphson method x_n .

Solution. The first three MATLAB scripts show how the function $f(x)$, its first, and its second derivative have been defined as functions.

Listing 1: The function $f(x)$ defined in the MATLAB file `funcOne.m`

```

1 %% 2019862s
2
3 %%This is the given function f(x)
4
5 function f = funcOne(x)
6     f = (x-1).^2./x;
7 end

```

Listing 2: The derivative of the function - $f'(x)$ defined in the MATLAB file `funcPrime.m`

```

1 %% 2019862s
2
3 %%This is the first derivative of the given function f(x)
4
5 function df = funcPrime(x)
6     df = (x.^2-1)./x.^2;
7 end

```

Listing 3: The second derivative of the function - $f''(x)$ defined in the MATLAB file `funcSec.m`

```

1 %% 2019862s
2
3 %%This is the second derivative of the given function f(x)
4
5 function ddf = funcSec(x)
6     ddf = 2./(x.^3);
7 end

```

The next file, called `newton.m` is used to actually implement the Newton-Raphson method, with float variables the initial guess and the specified tolerance.

Listing 4: The Newton-Raphson method used to count the number of iterations.

```

1 %% 2019862s
2
3 %% Function, which implements Newton-Raphson
4 %% method. Depends on a given function - f,

```



```

5  %% its derivative - df, an initial guess - x0,
6  %% and specified tolerance - tol.
7
8  function newtonMethod = newton (f , df , x0 , tol )
9
10 iter = 0;
11 err = Inf;
12
13 disp('iter ..... x ..... |x(n+1)-x(n)| ')
14
15 fprintf('%2.0f %+65.55e %+15.6e \n', iter, x0, err)
16 while (err>tol)
17
18     x1=x0-f(x0)./df(x0);
19     err=abs(x1-x0);
20     x0=x1;
21     iter=iter+1;
22     newtonMethod(1,iter)=x0;
23     fprintf('%2.0f %+65.55e %+15.6e \n', iter, x0, err)
24 end

```

The next MATLAB script is the main file, which was used to run the method and plot the result.

Listing 5: Main file.

```

1  %% 2019862s
2
3  %% Main file, used to run the implemented
4  %% code for the Newton-Raphson method, to plot
5  %% the resulting sequence, to determine the order of convergence,
6  %% and to plot the order of convergence.
7
8  newtonMethod = newton( @funcOne , @funcPrime , 1/2, 5*10^(-16));
9  oc = orderConv( @funcOne , @funcPrime , 1/2, 5*10^(-16));
10
11 %% This figure plots the sequence produced
12 %% by the original Newton-Raphson method
13 figure
14 scatter(1:1:length(newtonMethod), newtonMethod, [], 'filled');
15 title('Sequence produced by Newton-Raphson')
16 xlabel('Number of iterations')
17 ylabel('x_{n}')
18
19 %% This figure plots the order of convergence
20 %% of the original Newton-Raphson method
21 figure
22 scatter(3:1:length(oc)+2,oc, [], 'filled');
23 title('Order of convergence of Newton-Raphson')

```

```

24 xlabel('Number of iterations')
25 ylabel('Order')

```

The number of iterations needed to achieve the desired accuracy of 15 decimal places was obtained to be **51**. The screenshot below shows the size of a , which was used to store the iterated values, and the last couple of iterations.

```

47 +9.999999999999999289000000000000000000000000000000000000000000000000e-01 +7.105427e-15
48 +9.999999999999999645000000000000000000000000000000000000000000000000e-01 +3.552714e-15
49 +9.999999999999999822000000000000000000000000000000000000000000000000e-01 +1.776357e-15
50 +9.999999999999999911000000000000000000000000000000000000000000000000e-01 +8.881784e-16
51 +9.999999999999999956000000000000000000000000000000000000000000000000e-01 +4.440892e-16
>> size(a)

ans =

    1    51
fx >>

```

Figure 2: Last several iterations, and total number of iterations.

And finally, below is a scatter plot of the sequence produced by the Newton-Raphson method. The reason why `scatter` was used instead of `plot` as it is easier for the reader to see the specific points, rather than a continuous function, which interpolated through all points.

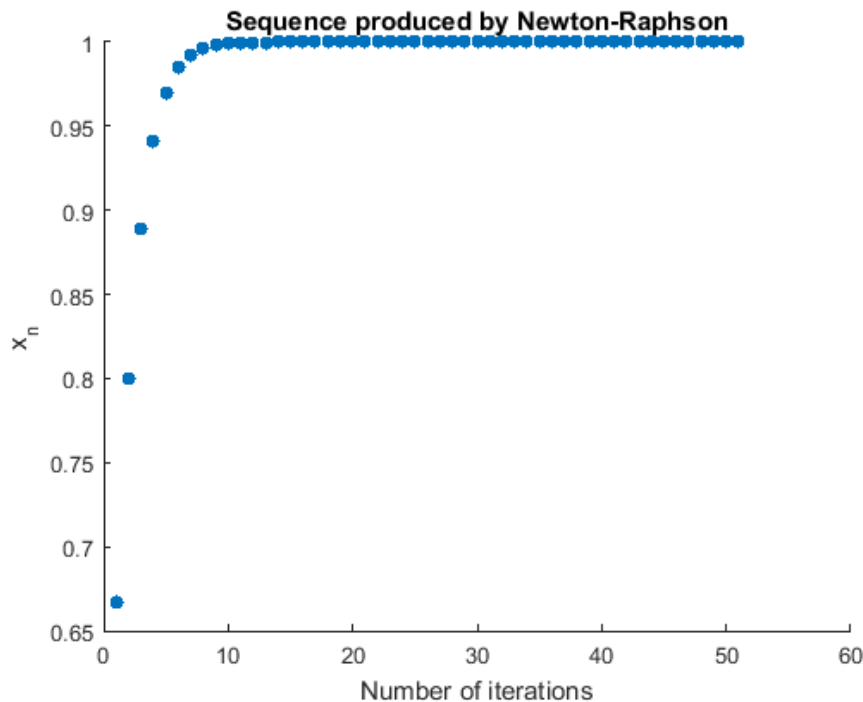


Figure 3: A scatter-plot of the sequence produced by the Newton-Raphson method.

□

(e) Use Claim (1.25) from the Lecture Notes to estimate analytically the number of iterations needed to achieve the given accuracy. How does it compare to the actual number of iterations counted? Can you improve your estimate?

Solution. By **Claim 1.17** from the Lecture Notes, since $g(x)$ is differentiable in $x \in (0, \infty)$, $g(x)$ is Lipschitz with Lipschitz constant

$$L = \sup_{\eta \in D} |g'(\eta)|.$$

The expression for the first derivative of $g(x)$ is

$$g'(\eta) = \frac{2}{(\eta + 1)^2}.$$

As $g(x)$ is convergent for $\forall x > \sqrt{2} - 1$, evaluate $g'(\eta)$ at $\eta = \sqrt{2} - 1$.

$$g'(\sqrt{2} - 1) = \frac{2}{(\sqrt{2} - 1 + 1)^2} = \frac{2}{2} = 1 = L.$$

But we require $L < 1$, so we need to choose another value of η . Evaluate $g'(x)$ at $x_0 = \frac{1}{2}$.

$$g'\left(\frac{1}{2}\right) = \frac{2}{\frac{9}{4}} = \frac{8}{9} = L.$$

By **Claim 1.25**, the error bound on the n -th iteration is given by

$$|x_n - x^*| \leq \frac{L^n}{1 - L} |g(x_0) - x_0|.$$

We know $x^* = 1$, $L = \frac{8}{9}$, $x_0 = \frac{1}{2}$ and

$$g(x_0) = \left(\frac{2x_0}{x_0 + 1}\right) = \frac{2 \times \frac{1}{2}}{\frac{1}{2} + 1} = \frac{1}{\frac{3}{2}} = \frac{2}{3}.$$

Further, $x_n - x^* = e_n$, so we require

$$|e_n| \leq \frac{\left(\frac{8}{9}\right)^n}{\frac{1}{9}} \left|\frac{2}{3} - \frac{1}{2}\right| \leq 5 \times 10^{-16} \quad \therefore \quad \left(\frac{8}{9}\right)^n \times \frac{3}{2} \leq 5 \times 10^{-16}.$$

Solving for n using MATLAB, we obtain

$$n \geq 302,56809 \dots,$$

hence, we require at least **n=303** iterations in order to obtain the given accuracy. The number obtained is ≈ 6 times larger than the 51 iterations obtained in MATLAB. The estimate can be improved by choosing an initial guess, which is closer to the root, e.g. $x_0 = \frac{99}{100}$, which in turn would produce a different Lipschitz constant, by making an iterative refinement of the map, or simply by modifying the model, which we will do in the next question. \square

(f) Pretend that the root $a = 1$ is not known beforehand. Define $e_n = x_n - x_{n-1}$ as a proxy for the absolute error. Use Definition (1.29) from the Lecture Notes to derive that the order of convergence of the method α is approximated by

$$\alpha = \lim_{n \rightarrow \infty} \frac{\log \left| \frac{e_n}{e_{n-1}} \right|}{\log \left| \frac{e_{n-1}}{e_{n-2}} \right|}.$$

Modify the Newton-Raphson code to compute this quantity at each iteration step n and so establish the order of convergence “experimentally”. Plot the resulting curve α_n . How does the result compare to the “theoretical” order of convergence derived in part (c)?

Solution. By **Definition 1.29**, the order of convergence of a sequence (produced by an iterative method) is defined by the integer α such that

$$\lim_{n \rightarrow \infty} \left| \frac{e_{n+1}}{e_n^\alpha} \right| = C \neq 0. \quad (7)$$

Hence,

$$\lim_{n \rightarrow \infty} |e_n| = C \lim_{n \rightarrow \infty} |e_{n-1}^\alpha|.$$

Taking logarithms gives

$$\lim_{n \rightarrow \infty} \log |e_n| = \log C + \alpha \lim_{n \rightarrow \infty} \log |e_{n-1}|. \quad (8)$$

Reindexing $n \rightarrow m - 1$ has no effect on the order of convergence yields

$$\lim_{m \rightarrow \infty} |e_{m-1}| = C \lim_{m \rightarrow \infty} |e_{m-2}^\alpha|,$$

so we can write

$$\lim_{n \rightarrow \infty} |e_{n-1}| = C \lim_{n \rightarrow \infty} |e_{n-2}^\alpha|,$$

and

$$\lim_{n \rightarrow \infty} \log |e_{n-1}| = \log C + \alpha \lim_{n \rightarrow \infty} \log |e_{n-2}|. \quad (9)$$

Subtracting Equation 9 from Equation 8 and using properties of limits and logarithms,

we obtain

$$\begin{aligned}
\lim_{n \rightarrow \infty} \log|e_n| - \lim_{n \rightarrow \infty} \log|e_{n-1}| &= \cancel{\log C} + \alpha \lim_{n \rightarrow \infty} \log|e_{n-1}| - \cancel{\log C} - \alpha \lim_{n \rightarrow \infty} \log|e_{n-2}| \\
\lim_{n \rightarrow \infty} \left(\log|e_n| - \log|e_{n-1}| \right) &= \alpha \lim_{n \rightarrow \infty} \left(\log|e_{n-1}| - \log|e_{n-2}| \right) \\
\lim_{n \rightarrow \infty} \log \left| \frac{e_n}{e_{n-1}} \right| &= \alpha \lim_{n \rightarrow \infty} \log \left| \frac{e_{n-1}}{e_{n-2}} \right| \\
\alpha &= \frac{\lim_{n \rightarrow \infty} \log \left| \frac{e_n}{e_{n-1}} \right|}{\lim_{n \rightarrow \infty} \log \left| \frac{e_{n-1}}{e_{n-2}} \right|} \\
\therefore \alpha &= \lim_{n \rightarrow \infty} \frac{\log \left| \frac{e_n}{e_{n-1}} \right|}{\log \left| \frac{e_{n-1}}{e_{n-2}} \right|},
\end{aligned}$$

as required.

Now, let us examine whether this holds for our specific map.

Recall that the map is given by

$$x_{n+1} = \frac{2x_n}{1+x_n}.$$

Now, we shall derive the recursive relation for x_{n-1} , that is

$$x_{n-1} = \frac{x_n}{2-x_n}.$$

As stated in the formulation of the problem, we use a proxy for the errors, i.e.

$$e_n = x_n - x_{n-1}, \quad e_{n-1} = x_{n-1} - x_{n-2}.$$

Substituting the obtained expressions in the definition of order of convergence, we obtain

$$\begin{aligned}
\log \lim_{n \rightarrow \infty} \left| \frac{e_n}{e_{n-1}} \right| &= \log \lim_{n \rightarrow \infty} \left| \frac{x_n - x_{n-1}}{x_{n-1} - x_{n-2}} \right| \\
&= \log \lim_{n \rightarrow \infty} \left| \frac{\frac{2x_{n-1}}{1+x_{n-1}} - x_{n-1}}{x_{n-1} - \frac{x_{n-1}}{2-x_{n-1}}} \right| \\
&= \log \lim_{n \rightarrow \infty} \left| \frac{(2x_{n-1} - \cancel{x_{n-1} - x_{n-1}^2})(2-x_{n-1})}{(1+x_{n-1})(2x_{n-1} - \cancel{x_{n-1} - x_{n-1}^2})} \right| \\
&= \log \lim_{n \rightarrow \infty} \left| \frac{2-x_{n-1}}{1+x_{n-1}} \right|,
\end{aligned}$$

where as $n \rightarrow \infty$, $x_i \rightarrow x^*$, $\forall i$, where x^* is the root of $f(x)$, hence

$$\begin{aligned} \log \lim_{n \rightarrow \infty} \left| \frac{e_n}{e_{n-1}} \right| &= \log \lim_{n \rightarrow \infty} \left| \frac{2 - x_{n-1}}{1 + x_{n-1}} \right| \\ &= \log \left| \frac{2 - x^*}{1 + x^*} \right|. \end{aligned}$$

Apply the same to the ratio of the error on the $(n-1)$ -th step to the error on the $(n-2)$ -th step, to obtain

$$\begin{aligned} \log \lim_{n \rightarrow \infty} \left| \frac{e_{n-1}}{e_{n-2}} \right| &= \log \lim_{n \rightarrow \infty} \left| \frac{x_{n-1} - x_{n-2}}{x_{n-2} - x_{n-3}} \right| \\ &= \log \lim_{n \rightarrow \infty} \left| \frac{\frac{2x_{n-2}}{1 + x_{n-2}} - x_{n-2}}{x_{n-2} - \frac{x_{n-2}}{2 - x_{n-2}}} \right| \\ &= \log \lim_{n \rightarrow \infty} \left| \frac{(x_{n-2} - x_{n-2}^2)(2 - x_{n-2})}{(1 + x_{n-2})(x_{n-2} - x_{n-2}^2)} \right| \\ &= \log \lim_{n \rightarrow \infty} \left| \frac{2 - x_{n-2}}{1 + x_{n-2}} \right|, \end{aligned}$$

where as $n \rightarrow \infty$, $x_i \rightarrow x^*$, $\forall i$, where x^* is the root of $f(x)$, hence

$$\begin{aligned} \log \lim_{n \rightarrow \infty} \left| \frac{e_{n-1}}{e_{n-2}} \right| &= \log \lim_{n \rightarrow \infty} \left| \frac{2 - x_{n-2}}{1 + x_{n-2}} \right| \\ &= \log \left| \frac{2 - x^*}{1 + x^*} \right|. \end{aligned}$$

Now evaluate the given expression for α to obtain

$$\lim_{n \rightarrow \infty} \frac{\log \left| \frac{2 - x^*}{1 + x^*} \right|}{\log \left| \frac{2 - x^*}{1 + x^*} \right|} = \lim_{n \rightarrow \infty} 1 = \alpha,$$

i.e. $\alpha = 1$. Now,

$$\frac{1}{1!} g^{(1)}(x_n) = \frac{2}{(1 + x_n)^2} \neq 0,$$

hence, using **Claim 1.30** from the Lecture notes, the order of convergence is indeed 1, irrespective of the root.

The following MATLAB script implements the formula for deriving the order of convergence.

Listing 6: Function used to experimentally calculate the order of convergence.

```
1  %% 2019862s
2
3  %% Function, which determines the order of
4  %% convergence of the Newton-Raphson
5  %% method. Depends on a given function - f,
6  %% its derivative - df, an initial guess - x0,
7  %% and specified tolerance - tol.
8
9  function orderOfConv=orderConv (f , df , x0 , tol )
10
11  iter = 0;
12  err = Inf;
13
14  disp('iter ..... Order of convergence ')
15
16  while (err>tol)
17      x1=x0-f(x0)./df(x0);
18      err=abs(x1-x0);
19      x0=x1;
20      iter=iter+1;
21      error(iter) = err;
22      if iter > 2 && err>tol
23          orderOfConv(1,iter-2) = ...
24              log(abs(error(iter)/error(iter-1)))/...
25              log(abs(error(iter-1)/error(iter-2)));
26          fprintf('%2.0f %23.6e\n', iter , orderOfConv(1,iter-2));
27      end
28  end
```

The figure below illustrates the resulting curve from the experimental computation of the order of convergence. Even though it starts at ≈ 1.8 , with several iterations it confirms the established analytically linear order of convergence as can be seen in the figure below. \square

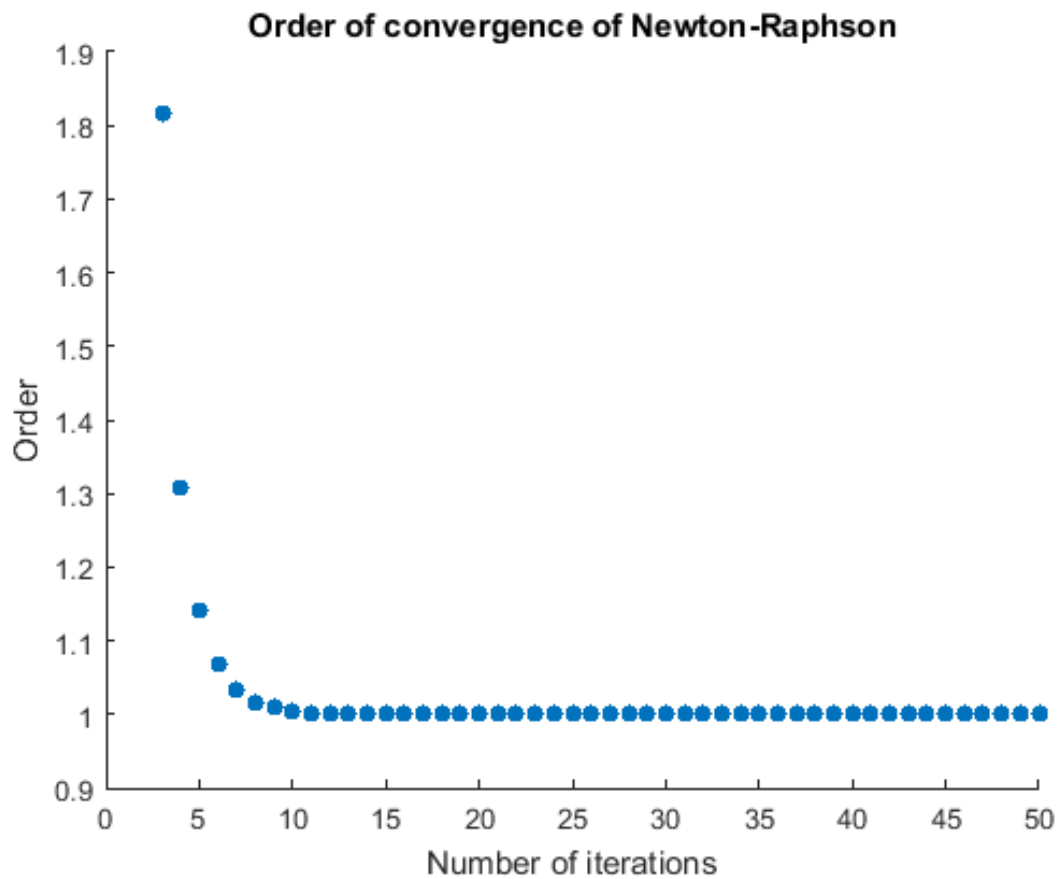


Figure 4: Order of convergence of the Newton-Raphson method.

Question 3. Consider the function $\mu(x) = \frac{f(x)}{f'(x)}$.

(a) Show that if $x = a$ is a root of multiplicity m of $f(x)$, then $x = a$ is a simple root (root of multiplicity 1) of $\mu(x)$.

Solution. The function we are interested in is defined as $\mu(x) = \frac{f(x)}{f'(x)}$. Since it involves a lot of terms, we shall work component wise, that is, first describe the function in the numerator, then the one in the denominator, and finally combine the two to form the required function $\mu(x)$.

$$f(x) = \frac{(x-a)^m f^{(m)}(a)}{m!} + \frac{(x-a)^{m+1} f^{(m+1)}(\xi)}{(m+1)!},$$

$$f'(x) = \frac{(x-a)^{m-1} f^{(m)}(a)}{(m-1)!} + \frac{(x-a)^m f^{(m+1)}(\xi)}{m!}.$$

Now combining the two expressions and taking into consideration the fact that $(m+1)! = (m+1)m!$ and $m! = (m-1)!m$, we obtain

$$\mu(x) = \frac{\left[\frac{(x-a)^m f^{(m)}(a)}{m!} + \frac{(x-a)^{m+1} f^{(m+1)}(\xi)}{(m+1)!} \right]}{\left[\frac{(x-a)^{m-1} f^{(m)}(a)}{(m-1)!} + \frac{(x-a)^m f^{(m+1)}(\xi)}{m!} \right]}. \quad (10)$$

Hence, we proceed to manipulating the expression in order to be able to derive any conclusions about the multiplicity of the root a .

$$\begin{aligned} \mu(x) &= \frac{\left[\frac{(x-a)^m f^{(m)}(a)}{m!} + \frac{(x-a)^{m+1} f^{(m+1)}(\xi)}{(m+1)!} \right]}{\left[\frac{(x-a)^{m-1} f^{(m)}(a)}{(m-1)!} + \frac{(x-a)^m f^{(m+1)}(\xi)}{m!} \right]} \\ &= \frac{(x-a)^m \left[\frac{f^{(m)}(a)}{m!} + \frac{(x-a) f^{(m+1)}(\xi)}{(m+1)!} \right]}{(x-a)^{m-1} \left[\frac{f^{(m)}(a)}{(m-1)!} + \frac{(x-a) f^{(m+1)}(\xi)}{m!} \right]} \\ &= \frac{(x-a)^m \cancel{m!} \left[(m+1) f^{(m)}(a) + (x-a) f^{(m+1)}(\xi) \right]}{(x-a)^{m-1} \cancel{m!} (m+1) \left[m f^{(m)}(a) + (x-a) f^{(m+1)}(\xi) \right]} \end{aligned}$$

$$\begin{aligned}
\mu(x) &= \frac{(x-a)}{(m+1)} \left[\frac{\cancel{mf^{(m)}(a)} + \cancel{(x-a)f^{(m+1)}(\xi)}}{\cancel{mf^{(m)}(a)} + \cancel{(x-a)f^{(m+1)}(\xi)}} + \frac{f^{(m)}(a) + (x-a)f^{(m+1)}(\xi)}{mf^{(m)}(a) + (x-a)f^{(m+1)}(\xi)} \right] \\
&= \frac{(x-a)}{(m+1)} \left[1 + \frac{f^{(m)}(a) + (x-a)f^{(m+1)}(\xi)}{mf^{(m)}(a) + (x-a)f^{(m+1)}(\xi)} \right] \\
&= (x-a) \times \underbrace{\frac{1}{(m+1)} \left[\frac{mf^{(m)}(a) + (x-a)f^{(m+1)}(\xi) + f^{(m)}(a) + (x-a)f^{(m+1)}(\xi)}{mf^{(m)}(a) + (x-a)f^{(m+1)}(\xi)} \right]}_{Q(x)}
\end{aligned}$$

$$\therefore \mu(x) = (x-a)Q(x). \quad (11)$$

Now, we have shown that the function $\mu(x)$ can be represented in the form $\mu(x) = (x-a)^m Q(x)$, as shown in Question 1. Evaluate the function $Q(x)$ at the root $x = a$ to obtain

$$\begin{aligned}
Q(a) &= \frac{1}{(m+1)} \left[\frac{\cancel{mf^{(m)}(a)} + \cancel{(a-a)f^{(m+1)}(\xi)} + \overset{0}{f^{(m)}(a)} + \overset{0}{(a-a)f^{(m+1)}(\xi)}}{\cancel{mf^{(m)}(a)} + \cancel{(a-a)f^{(m+1)}(\xi)}} \right] \\
&= \frac{1}{(m+1)} \left[\frac{\overset{0}{(m+1)f^{(m)}(a)}}{\cancel{mf^{(m)}(a)}} \right] \\
&= \frac{1}{\cancel{(m+1)}} \frac{\overset{0}{(m+1)}}{m} \\
&= \frac{1}{m} \neq 0 \quad \forall m. \quad (12)
\end{aligned}$$

By Equation 11 and Equation 12, we have shown that $\mu(x) = (x-a)^m Q(x)$, where $Q(a) \neq 0$, and $m = 1$, so indeed, if $x = a$ is a root of multiplicity m of $f(x)$, then $x = a$ is a simple root, i.e. $m = 1$, of $\mu(x)$, as required. \square

(b) Construct a Newton-Raphson formula based on $\mu(x)$ for the solution of $f(x) = 0$.

Solution. In order to construct a Newton-Raphson formula based on $\mu(x)$, we first need to recall the map, which defines the method, that is

$$g(x_n) = x_n - \frac{\mu(x)}{\mu'(x)}.$$

Again, for the ease of the reader, we shall discuss the numerator and denominator separately. Using the given definition of $\mu(x)$ and then applying the Chain Rule, we obtain

$$\begin{aligned}\mu(x_n) &= \frac{f(x_n)}{f'(x_n)}, \\ \mu'(x_n) &= \frac{[f'(x_n)]^2 - f(x_n)f''(x_n)}{[f'(x_n)]^2}.\end{aligned}$$

Plugging the obtained expressions in the formula for the map of the Newton-Raphson method we obtain

$$\begin{aligned}g(x_n) &= x_n - \frac{\frac{f(x_n)}{f'(x_n)}}{\frac{[f'(x_n)]^2 - f(x_n)f''(x_n)}{[f'(x_n)]^2}} \\ &= x_n - \frac{f(x_n)[f'(x_n)]^{\cancel{2}}^1}{\cancel{f'(x_n)}[f'(x_n)]^2 - f(x_n)f''(x_n)} \\ \therefore g(x_n) &= x_n - \frac{f(x_n)[f'(x_n)]}{[f'(x_n)]^2 - f(x_n)f''(x_n)}.\end{aligned}\tag{13}$$

As required, the Newton-Raphson formula based on $\mu(x)$ for the solution of $f(x) = 0$ is given by Equation 13. \square

- (c) The formula you derived in the previous part is known as the “modified Newton-Raphson method”. What order of convergence do you expect this method to have in general and why?

Solution. In general, the Newton-Raphson method with a simple root has a quadratic order of convergence. And since our function $\mu(x)$ has a simple root, we can expect the method to converge to its root quadratically. But in order to be precise, let us examine this carefully and analytically. Given the modified Newton-Raphson method in Equation 13, that is,

$$\frac{f(x_n)[f'(x_n)]^{\overset{1}{\cancel{2}}}}{\cancel{f'(x_n)}[f'(x_n)]^2 - f(x_n)f''(x_n)},$$

we need to compute its derivatives in order to be able to analyze the behaviour of the map. Hence, we proceed with differentiating $g(x)$ once and twice to obtain

$$\begin{aligned} g'(x_n) &= 1 - \left[\frac{\mu(x_n)}{\mu'(x_n)} \right]' \\ &= 1 - \frac{[\mu(x_n)]^2 - \mu(x_n)\mu''(x_n)}{[\mu(x_n)]^2} \\ &= 1 - \frac{\cancel{[\mu(x_n)]^2}}{\cancel{[\mu(x_n)]^2}} + \frac{\mu(x_n)\mu''(x_n)}{[\mu(x_n)]^2} \\ &= \frac{\mu(x_n)\mu''(x_n)}{[\mu(x_n)]^2}, \end{aligned}$$

but we know that a is a root of $\mu(x)$, so $\mu(a) = 0$, hence,

$$\therefore g'(x_n) = 0,$$

which shows that the modified Newton-Raphson method has order of converge greater than 1, that is, the order of convergence is at least 2. Now, compute the second derivative of the map.

$$g''(x_n) = \frac{\overset{0}{\cancel{-2\mu(x_n)[\mu''(x_n)]^2}} + \overset{0}{\cancel{\mu(x_n)\mu''(x_n)\mu'(x_n)}} + [\mu'(x_n)]^2\mu''(x_n)}{[\mu'''(x_n)]^3},$$

as $\mu(x_n) = 0$, and so we obtain

$$g''(x_n) = \frac{[\mu'(x_n)]^2\mu''(x_n)}{[\mu'''(x_n)]^3} \neq 0,$$

which is definitely not equal to zero as both $\mu'(x_n) \neq 0$ and $\mu''(x_n) \neq 0$. Hence, the modified Newton-Raphson method has quadratic order of convergence as supposed earlier. \square

(d) Implement the modified Newton-Raphson method to solve $f(x) = 0$. Include the code in your report.

Solution. The following MATLAB script is the implementation of the modified Newton-Raphson model, derived in Question 3b.

Listing 7: Implementation of the modified Newton-Raphson.

```

1  %% 2019862s
2
3  %% Function, which implements modified Newton-Raphson
4  %% method. Depends on a given function - f,
5  %% its derivative - df, its second derivative - ddf,
6  %% an initial guess - x0,
7  %% and specified tolerance - tol.
8
9  function newtonModified = newtonMod (f , df , ddf, x0 , tol )
10
11  iter = 0;
12  err = Inf;
13
14  disp('iter ..... x ..... |x(n+1)-x(n)| ')
15
16  fprintf('%2.0f %+65.55e %+15.6e \n', iter, x0, err)
17
18  while (err>tol)
19
20      x1=x0-(f(x0)*df(x0))./((df(x0)).^2-f(x0)*ddf(x0));
21      err=abs(x1-x0);
22      if err>tol
23          x0=x1;
24          iter=iter+1;
25          newtonModified(1,iter)=x0;
26          fprintf('%2.0f %+65.55e %+15.6e \n', iter, x0, err)
27      end
28  end

```

□

Question 4. Consider the function $f(x) = \frac{(x-1)^2}{x}$.

- (a) Count the actual number of iterations needed to achieve accuracy of 15 decimal places in the solution of $f(x) = 0$ using the modified Newton-Raphson method with $x_0 = \frac{1}{2}$. Plot the sequence produced by the modified Newton-Raphson method x_n .

Solution. The actual number of iterations for the modified method is extremely lower compared to the original Newton-Raphson. This implementation requires only **6** iterations to achieve the desired accuracy. The figure below shows the sequence produced by the modified Newton-Raphson method.

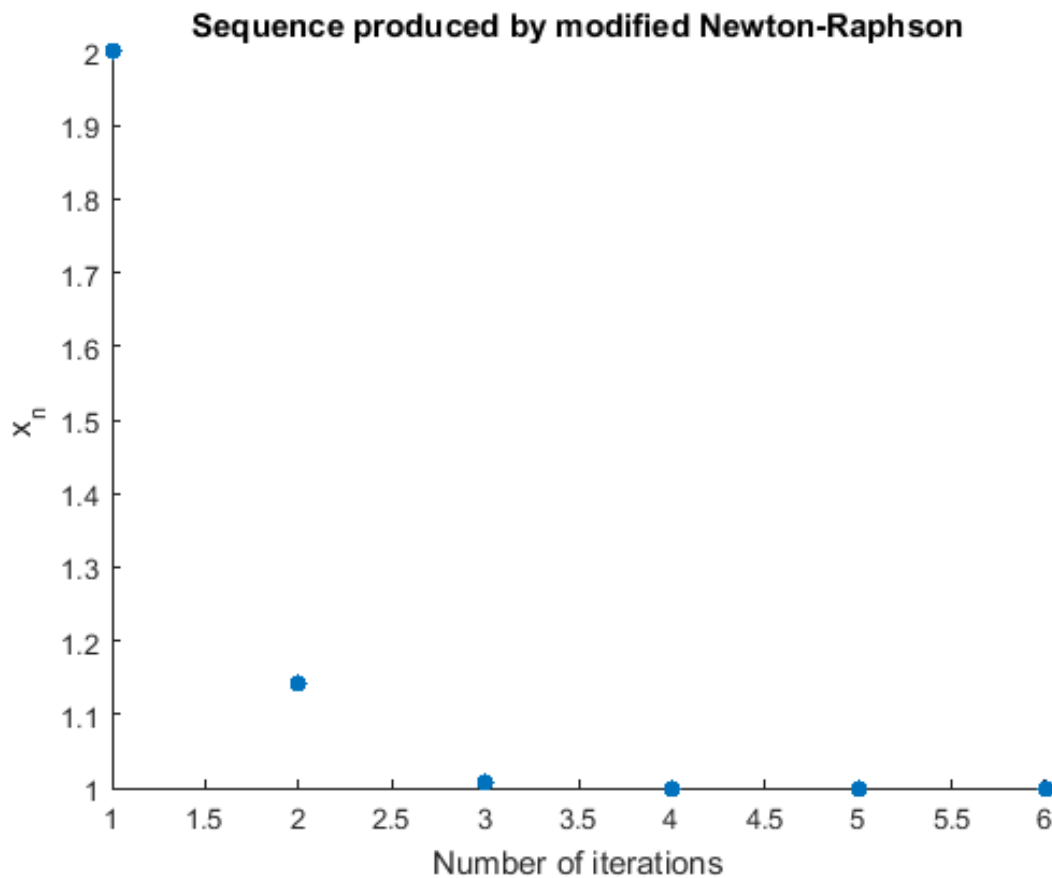


Figure 5: Sequence produced by modified Newton-Raphson.

□

- (b) Determine the order of convergence α of the modified Newton-Raphson method experimentally and produce a plot analogous to that in Question 2(f).

Solution. The MATLAB script below illustrates the function used to determine the order of convergence of the modified Newton-Raphson method.

Listing 8: Function used to calculate the order of convergence of the modified method.

```

1  %% 2019862s
2
3  %% Function, which determines the order of
4  %% convergence of the modified Newton-Raphson
5  %% method. Depends on a given function - f,
6  %% its derivative - df, its second derivative - ddf,
7  %% an initial guess - x0,
8  %% and specified tolerance - tol.
9
10 function orderOfConvMod=orderConvMod (f, df, ddf, x0 , tol )
11
12 iter = 0;
13 err = Inf;
14
15 disp('iter ..... Order of convergence ')
16
17 while (err>tol)
18     x1=x0-(f(x0)*df(x0))/((df(x0)).^2-f(x0)*ddf(x0));
19     err=abs(x1-x0);
20     x0=x1;
21     iter=iter+1;
22     error(iter) = err;
23     if iter > 2 && err>tol
24         orderOfConvMod(1,iter-2) = ...
25             log(abs(error(iter)/error(iter-1)))/...
26             log(abs(error(iter-1)/error(iter-2)));
27         fprintf('%2.0f %+.3e\n', iter, orderOfConvMod(1,iter-2));
28     end
29 end

```

The following main file was used to run the above script with the specified conditions.

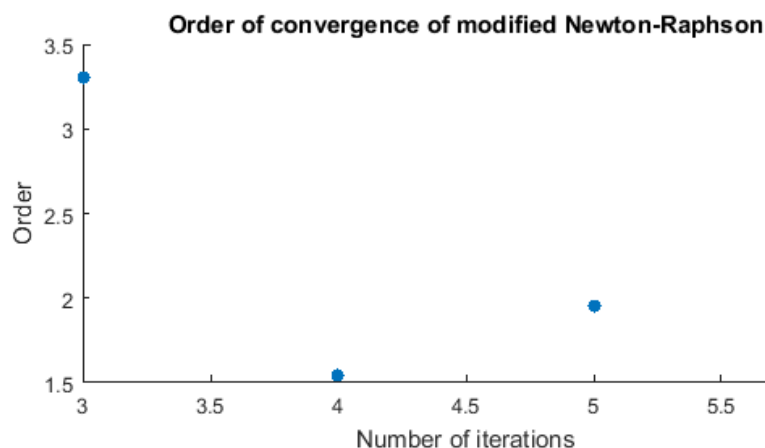
Listing 9: Main file for modified Newton-Raphson.

```

1  %% 2019862s
2
3  %% Main file, used to run the implemented
4  %% code for the modified Newton-Raphson method, to plot
5  %% the resulting sequence, to determine the order of convergence,
6  %% and to plot the order of convergence.
7
8  newtonMod=newtonMod(@funcOne,@funcPrime,@funcSec,1/2,5*10^(-16));
9  ocm=orderConvMod(@funcOne,@funcPrime,@funcSec,1/2,5*10^(-16));
10
11
12  %% This figure plots the sequence produced
13  %% by the modified Newton-Raphson method
14  figure
15  scatter(1:1:length(newtonMod), newtonMod, [], 'filled');
16  title('Sequence produced by modified Newton-Raphson')
17  xlabel('Number of iterations')
18  ylabel('x_{n}')
19
20
21  %% This figure plots the order of convergence
22  %% of the modified Newton-Raphson method
23  figure
24  scatter(3:1:length(ocm)+2,ocm, [], 'filled');
25  title('Order of convergence of modified Newton-Raphson')
26  xlabel('Number of iterations')
27  ylabel('Order')

```

The figure below is the plot, produced by implementing the MATLAB script. Indeed, it confirms the quadratic order of convergence, determined in Question 3. Even though it starts at ≈ 3.3 , it later on converges to 2 as it was predicted.



□