Programming Assignment 4

Jenny Petrova

December 9th, 2024

1 Executive Summary

This report demonstrates and compares the capabilities of the Regula Falsi (false position), Secant, Newton's (Newton-Raphson), and Steffensen's methods for approximating the roots (denoted x^*) of real-valued functions. Through a series of controlled experiments, we investigate the convergence behaviors for each root-finding method. In particular, our first set of tests focuses on higher-order root problems of the form

$$f(x) = (x - \rho)^d, \tag{1}$$

where we observe convergence rate trends in response to changes in the multiplicity (d) of the root ($x^* = \rho$). We then explore the performance of Newton's method on a generic cubic polynomial with three distinct roots ($x^* = 0, \rho, -\rho$):

$$f(x) = x^3 - \rho^2 x, \quad \rho > 0, \tag{2}$$

and investigate the Newton's sensitivity to the scaling of $f(x)$ such that

$$\tilde{f}(x) = \sigma f(x), \quad \sigma > 0. \tag{3}$$

The last set of experiments consider the gradual degradation of convergence rates from quadratic to linear to slower linear. We consider the function

$$f(x) = x(x - \rho_1)(x - \rho_2), \quad \rho_1 > 0, \rho_2 > 0, \tag{4}$$

and vary ρ_1 and ρ_2 to examine how parameter changes affect the convergence rate behaviors of the Newton's and Secant methods. Ultimately, we explore the influence of initial guesses and parameter choices on convergence or divergence for each method. After carefully selecting a small number of illustrative examples and analyzing their outcomes, we discuss when and how each method achieves its expected rate of convergence and under what conditions the performance of each method may degrade.

2 Statement of the Problem

Consider a continuous function $f : \mathbb{R} \mapsto \mathbb{R}$. For nonlinear functions, finding a closed-form solution (x^* such that $f(x^*) = 0$) to the equation is often impractical or even impossible. When an exact analytical solution cannot be determined, we can instead use numerical methods to approximate the exact roots (or "zeros") of f in a finite number of steps. However, it is quite important to note that most root-finding algorithms do not guarantee finding *every* root of f. In particular, if an algorithm fails to locate a root, we cannot necessarily conclude that *no* root exists. In practice, these methods require one or more initial guesses of the root as starting values, and perform iterations that produce increasingly accurate approximations to the root. The iterations stop when the approximation is sufficiently close (within a specified tolerance) to the root, not the exact solution. The next section describes the particular methods we will observe.

3 Description of the Methods, Algorithms and Implementation

We implement the code for the following algorithms in Python. Each method is defined as a separate Python function, with appropriate input parameters and output results (Section 3 will describe each algorithm in detail). For testing and generating visualizations, we make use of the standard NumPy, SciPy, Pandas, and Matplotlib libraries in Python. For each experiment, the corresponding plots and tables are automatically saved as .png and .csv files, respectively. We present the relevant plots and tables in Section 4 of this report.

3.1 Generic Method for Root-Finding Algorithms

The construction of the Regula Falsi, Secant, and Newton's methods follow a generic root-finding algorithm. All three methods can be written in the form

$$x_{k+1} = x_k + \frac{f(x_k)}{q_k}, \tag{5}$$

where

$$q_k = \frac{f(x_k) - f(y_k)}{x_k - y_k} = f[x_k, y_k] \tag{6}$$

is an approximation to the derivative of (or slope) at or near x_k. The choice of the point y_k and the strategy for updating the points after each iteration differentiate the behavior of the three methods. The differences between the methods are made clear in the following subsections.

3.2 Regula Falsi Method and Algorithm

The Regula Falsi method begins with an initial interval value $[x_0, x_1]$ such that $f(x_0)f(x_1) < 0$, ensuring that this interval brackets at least one root. The interval remains bracketed around a root at each iteration, ensuring convergence for Regula Falsi. At each iteration, the method constructs a secant line connecting the points $(x_0, f(x_0))$ and $(x_1, f(x_1))$, and then finds the x-intercept of this line. The procedure is as follows:

1. Compute the slope of the secant line:

$$q = \frac{f(x_1) - f(x_0)}{x_1 - x_0}.$$

2. Estimate the next approximation:

$$x_2 = x_1 - \frac{f(x_1)}{q}.$$

3. If the exact root x^* is known (for testing purposes), compute the error:

$$|e_k| = |x^* - x_2|.$$

4. Determine which sub-interval still contains the root anf set

$$\begin{cases} x_1 \leftarrow x_2 & \text{if } f(x_2)f(x_0) < 0, \\ x_0 \leftarrow x_2 & \text{otherwise.} \end{cases}$$

5. Repeat until the stopping criterion is met:

$$|f(x_2)| < 10^{-6}.$$

We compute the error terms because theoretical analysis suggests that Regula Falsi converges at least linearly. More concretely, there exists a constant C_{RF} such that

$$\frac{|e_{k+1}|}{|e_k|} \leq C_{RF}$$

for all sufficiently large k. Even as the number of iterations increases, the error should decrease proportionally with each iteration.

3.3 Secant Method and Algorithm

The Secant method is similar to Regula Falsi (beginning with an initial interval $[x_0, x_1]$) but does not require the interval to remain bracketed around the root with each iteration. Hence convergence is not always guaranteed for this method. Starting from two initial approximations x_0 and x_1, the Secant method proceeds as follows:

1. Compute the slope using the most recent approximations:

$$q = \frac{f(x_1) - f(x_0)}{x_1 - x_0}.$$

2. Estimate the next approximation:

$$x_2 = x_1 - \frac{f(x_1)}{q}.$$

3. If the exact root x^* is known, compute the error:

$$e_k = x^* - x_2.$$

4. Update the previous approximations:

$$x_0 \leftarrow x_1,$$

$$x_1 \leftarrow x_2.$$

5. Repeat until the stopping criterion is met:

$$|f(x_2)| < 10^{-6}.$$

Theoretical analysis suggests that the Secant method converges superlinearly, meaning there exists a constant C_S

$$\frac{|e_{k+1}|}{|e_k|^p} \approx C_S,$$

for all sufficiently large k, where $p \approx 1.618$ (the "golden ratio").

3.4 Newton's Method and Algorithm

Newton's method uses the derivative of the function to find successively better approximations of the root. We begin with an initial guess x_0 and a value for the parameter m (we will examine the performance of Newton's for different values of m in Section 4). We assume $f'(x)$ exists and is not zero near the root. The method iterates as follows:

1. Compute the derivative at the current approximation:

$$q = f'(x_0).$$

2. Estimate the the next approximation:

$$x_1 = x_0 - m \cdot \frac{f(x_0)}{f'(x_0)}.$$

3. If the exact root x^* is known, evaluate the error:

$$e_k = x^* - x_1.$$

4. Update the previous approximation:

$$x_0 \leftarrow x_1.$$

5. Repeat until the stopping criterion is met, for instance:

$$|f(x_{k+1})| < 10^{-6}.$$

Since Newton's method uses the actual derivative information, it often converges very rapidly when initiated sufficiently close to the root. Theoretically, we should examine quadratic convergence for Newton's method, such that

$$\frac{|e_{k+1}|}{|e_k|^2} \approx C_N,$$

for a constant C_N. Hence the difference between the root and the approximation is squared at each step.

3.5 Steffensen's Method and Algorithm

Steffensen's method is similar to Newton's in that it converges quadratically. However, whereas Newton's method requires the computation of the derivative $f^{'}(x)$, Steffensen's method does not require the use of derivatives. This is an advantage when we cannot easily or efficiently compute the derivative. Consider the difference approximation

$$\theta(x_k) = \frac{f(x_k + f(x_k)) - f(x_k)}{f(x_k)}. \tag{7}$$

Steffensen's method replaces $f^{'}(x_k)$ with the difference approximation given by $\theta(x_k)$. We again begin with an initial guess x_0 and iterate as follows:

1. Compute the difference approximation:

$$\theta(x_k) = \frac{f(x_k + f(x_k)) - f(x_k)}{f(x_k)}.$$

2. Estimate the the next approximation:

$$x_1 = x_0 - \frac{f(x_0)}{\theta(x_k)}.$$

3. If the exact root x^* is known, evaluate the error:

$$e_k = x^* - x_1.$$

4. Update the previous approximation:

$$x_0 \leftarrow x_1.$$

5. Repeat until the stopping criterion is met, for instance:

$$|f(x_{k+1})| < 10^{-6}.$$

We note a slight disadvantage in Steffensen's, exemplified in the algorithm above. Steffensen's method requires two function evaluations per step, whereas the Newton's and Secant methods require only one function evaluation per step. As with Newton's, theoretical results yield quadratic convergence of the error terms to a constant C_S, such that

$$\frac{|e_{k+1}|}{|e_k|^2} \approx C_S.$$

4 Description of Experimental Design and Results

In this section we observe the performance of the root-finding methods. We test three different function types. For each problem, we examine several different parameters, and observe under which conditions the methods do or do not converge. For each series of tests, our algorithm produces a table to show the convergence of $f(x_k) \to f(x^*) = 0$ as $x_k \to x^*$ per iteration. We include a few relevant tables to support our results. We also plot the error terms and error ratios per iteration, to compare the true convergence results to the theoretical convergence results. We mainly focus on the plots to demonstrate the performance of the methods. We detail the different experiments, corresponding results, and relevant plots below.

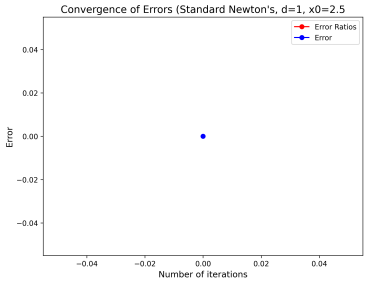
4.1 Higher Order Roots

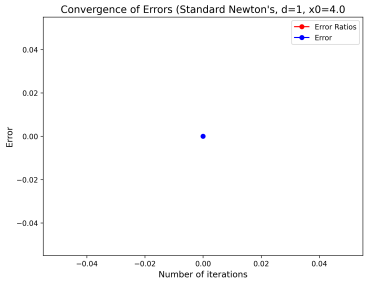
Our first set of tests observe the performance of the methods on a function of the form

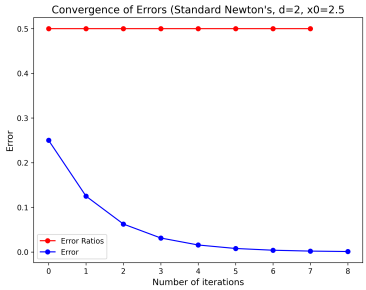
$$f(x) = (x - \rho)^d. \tag{8}$$

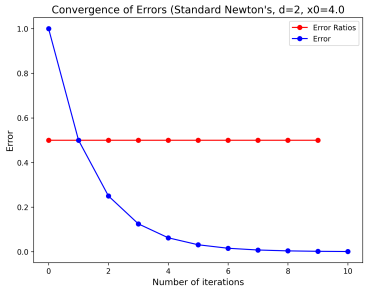
This problem has a root ρ and multiplicity d. For our experiments, we set $\rho = 2$. We will observe the convergence behavior for multiplicities ranging in the set $d = [1, 2, 3, 4, 5]$. For each method, we observe the general behavior for at least 2 initial inputs (initial value x_0 or an initial interval $[x_0, x_1]$, depending on the requirements of the particular method). Examine the following results:

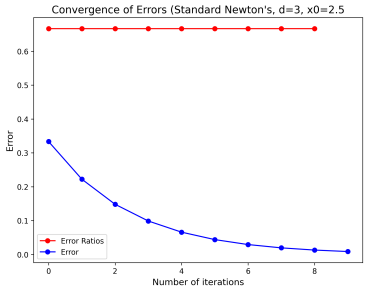
1. **Standard Newton's Method (m=1)**: Test Newton's method with parameter $m = 1$ and initial values $x_0 = \{2.5, 4.0\}$.

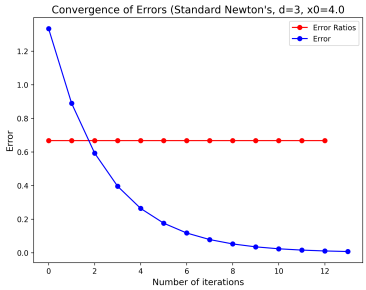


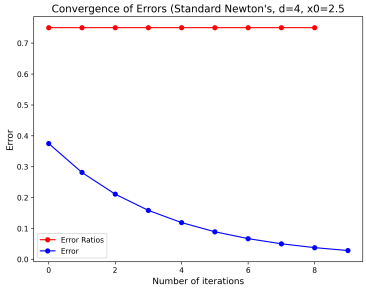


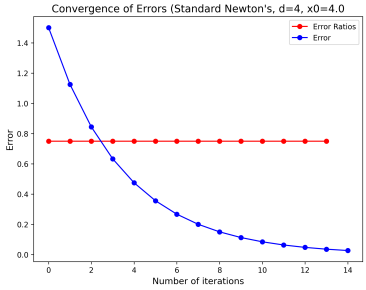


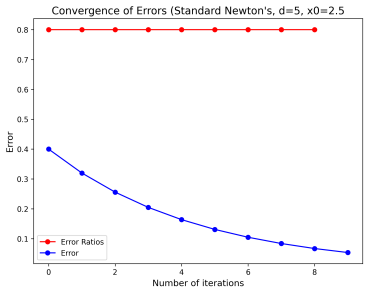











Convergence of Errors (Standard Newton's, d=5, x0=2.5
Error
Number of iterations
Error Ratios
Error

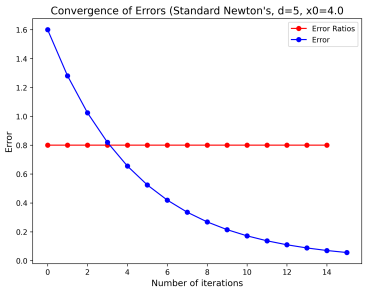
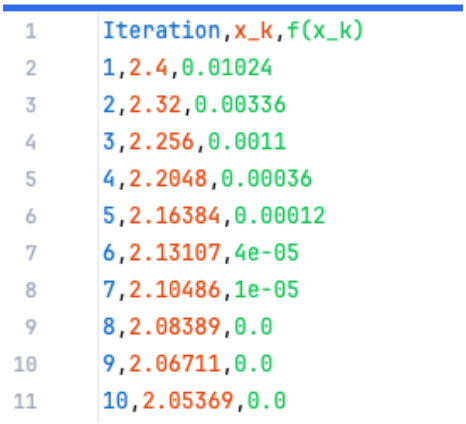

Convergence of Errors (Standard Newton's, d=5, x0=4.0
Error
Number of iterations
Error Ratios
Error

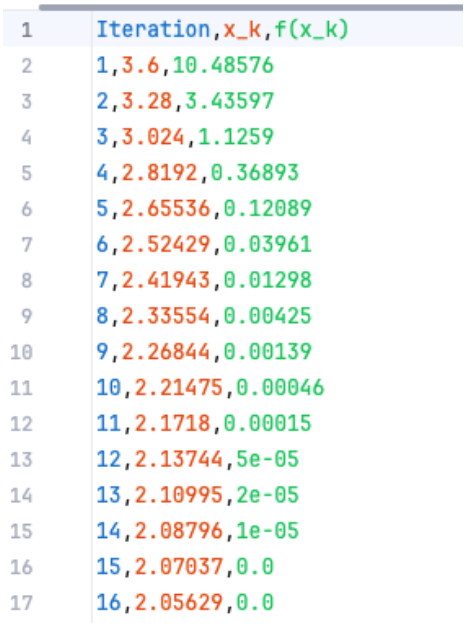
table-Standard Newton's-5-2-1.csv



```
Iteration,x_k,f(x_k)
1,2.4,0.01024
2,2.32,0.00336
3,2.256,0.0011
4,2.2048,0.00036
5,2.16384,0.00012
6,2.13107,4e-05
7,2.10486,1e-05
8,2.08389,0.0
9,2.06711,0.0
10,2.05369,0.0
```

Figure 1: Iteration Results (Standard Newton's, $d = 5$, $x_0 = 2.5$)

table-Standard Newton's-5-4-1.csv

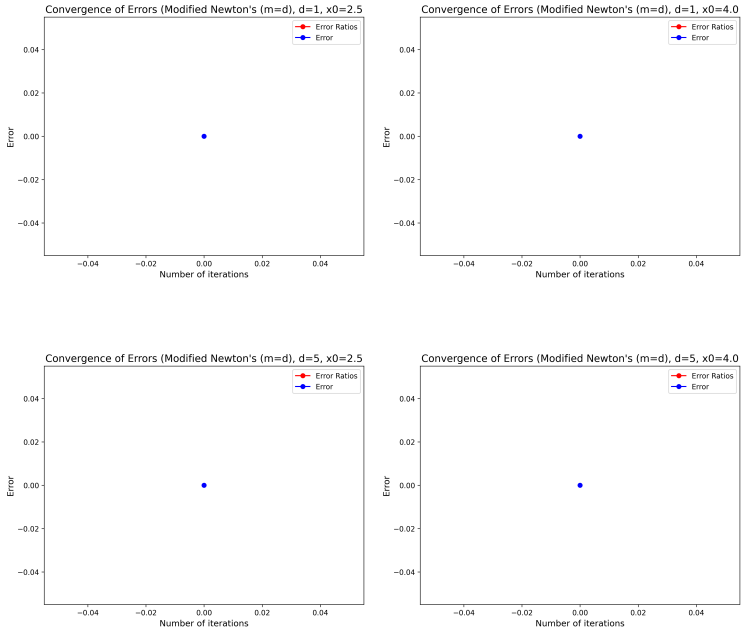


```
Iteration,x_k,f(x_k)
1,3.6,10.48576
2,3.28,3.43597
3,3.024,1.1259
4,2.8192,0.36893
5,2.65536,0.12089
6,2.52429,0.03961
7,2.41943,0.01298
8,2.33554,0.00425
9,2.26844,0.00139
10,2.21475,0.00046
11,2.1718,0.00015
12,2.13744,5e-05
13,2.10995,2e-05
14,2.08796,1e-05
15,2.07037,0.0
16,2.05629,0.0
```

Figure 2: Iteration Results (Standard Newton's, $d = 5$, $x_0 = 4.0$)

Standard Newton's method ($m = 1$) converges for all multiplicities d tested. When $d = 1$, Newton's converges in one step for both x_0 values tested. As d increases, the number of iterations until convergence remains about the same (per initial input value x_0). Newton's method takes more iterations to converge when x_0 is farther from the actual root. We notice that the error plots show linear convergence of the ratios for the error terms. This result is due to our observance of $m = 1$ for Standard Newton's method. When $m = 1$, the ratios of the error terms for Newton's method will converge to a value $1 - \frac{1}{d}$, where d is our value for the multiplicity. Notice this is true in the plots above. For example, when $d = 2$, we examine the error ratios converge linearly to $1 - \frac{1}{2} = 0.5$. When $d = 5$, we examine linear convergence at $1 - \frac{1}{5} = 0.8$.

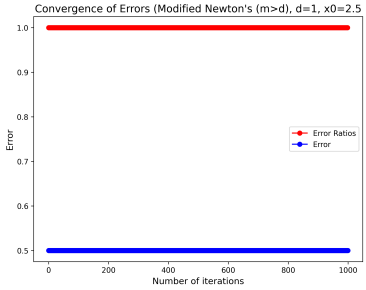
2. **Modified Newton's Method** ($m = d$): Test Newton's method with parameter $m = d$, where d is the multiplicity tested. We again test initial values $x_0 = \{2.5, 4.0\}$.

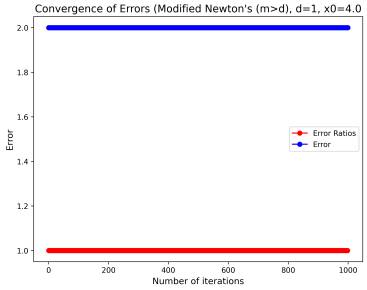


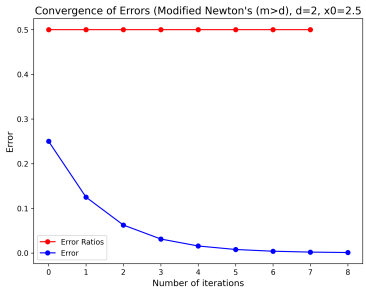
When $m = d$, Modified Newton's method converges directly to $x^* = \rho$ in one step, for any $x_0 \neq \rho$ tested. This result can be directly seen in the iteration formula:

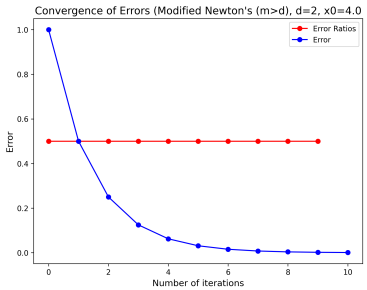
$$x_1 = x_0 - \frac{mf(x_0)}{f'(x_0)} = x_0 - \frac{d(x_0 - \rho)^d}{d(x_0 - \rho)^{d-1}} = x_0 - (x_0 - \rho) = \rho.$$

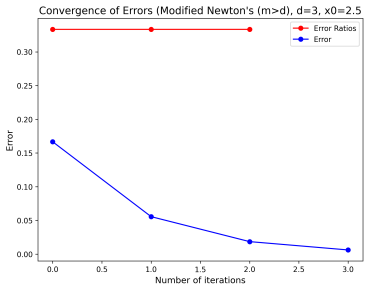
3. **Modified Newton's Method** ($m > d$): Test Newton's method when $m > d$, where d is the multiplicity tested. We set $m = d + 1$ and again test initial values $x_0 = \{2.5, 4.0\}$.

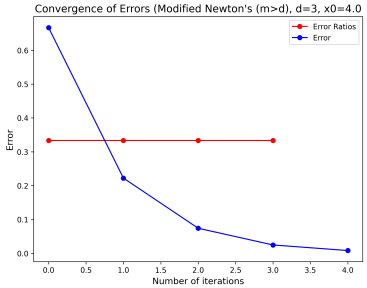


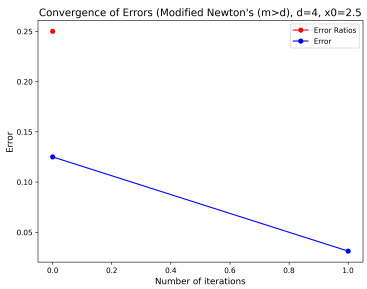


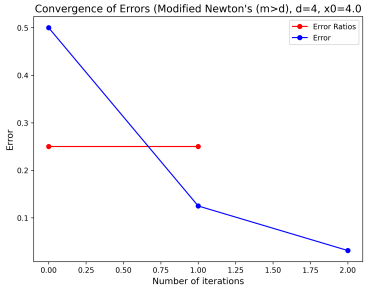












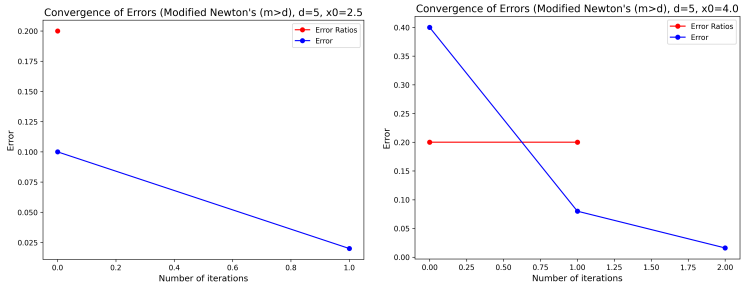


table-Modified Newton's (m>d)-2-4-1.csv

```
Iteration,x_k,f(x_k)
1,1.0,1.0
2,2.5,0.25
3,1.75,0.0625
4,2.125,0.01562
5,1.9375,0.00391
6,2.03125,0.00098
7,1.98438,0.00024
8,2.00781,6e-05
9,1.99609,2e-05
10,2.00195,0.0
11,1.99902,0.0
```

Figure 3: Iteration Results (Modified Newton's (m¿d), $d = 2$, $x_0 = 4.0$)

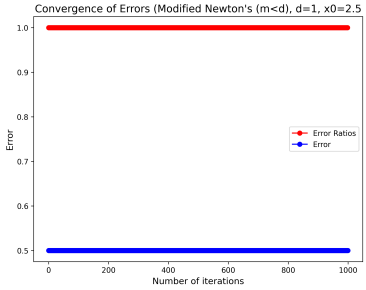
table-Modified Newton's (m>d)-5-4-1.csv

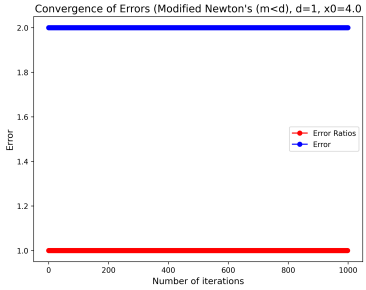
```
Iteration,x_k,f(x_k)
1,1.6,-0.01024
2,2.08,0.0
3,1.984,-0.0
```

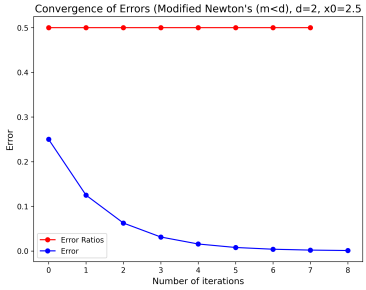
Figure 4: Iteration Results (Modified Newton's (m¿d), $d = 5$, $x_0 = 4.0$)

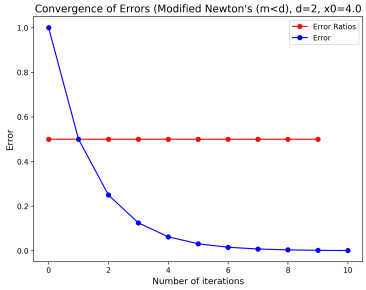
Modified Newton's method ($m > d$) converges when $d > 1$ (we note divergence in the plot where $d = 1$). As $d > 1$ increases, the number of iterations until convergence decreases (per initial input value x_0). Newton's method takes more iterations to converge when x_0 is farther from the actual root, but the overall number of iterations is similar for $x_0 = 2.5$ and $x_0 = 4.0$. We notice that the error plots again show linear convergence of the ratios for the error terms.

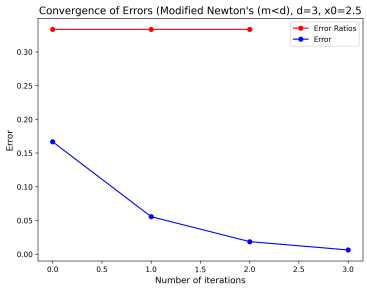
4. **Modified Newton's Method** ($m < d$): Test Newton's method when $m < d$, where d is the multiplicity tested. We set $m = d - 1$ and again test initial values $x_0 = \{2.5, 4.0\}$.

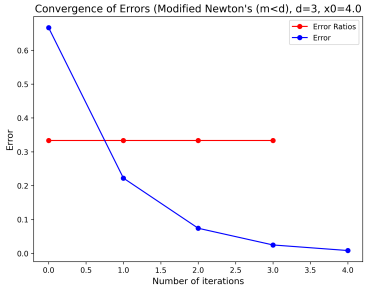


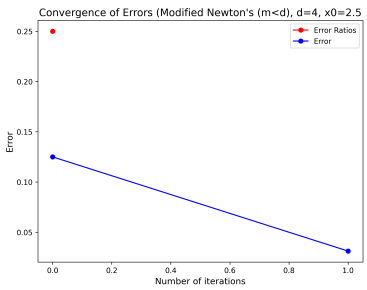


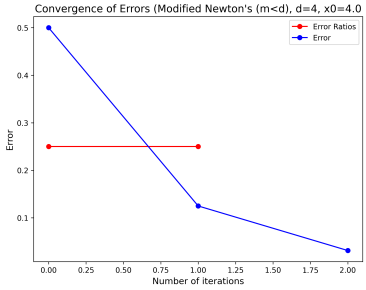












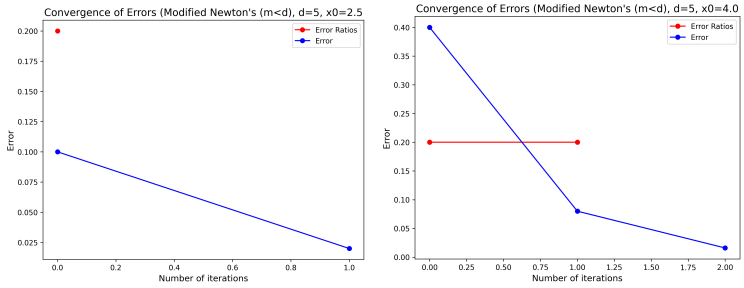


table-Modified Newton's (m<d)-2-4-1.csv

```
Iteration,x_k,f(x_k)
1,3.0,1.0
2,2.5,0.25
3,2.25,0.0625
4,2.125,0.01562
5,2.0625,0.00391
6,2.03125,0.00098
7,2.01562,0.00024
8,2.00781,6e-05
9,2.00391,2e-05
10,2.00195,0.0
11,2.00098,0.0
```

Figure 5: Iteration Results (Modified Newton's (m¡d), $d = 2$, $x_0 = 4.0$)

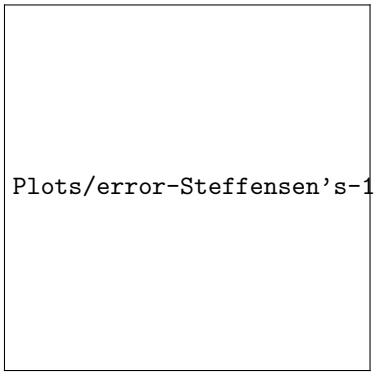
table-Modified Newton's (m<d)-5-4-1.csv

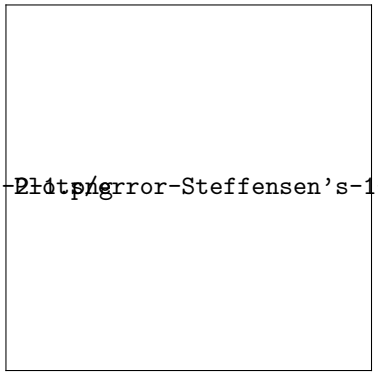
```
Iteration,x_k,f(x_k)
1,2.4,0.01024
2,2.08,0.0
3,2.016,0.0
```

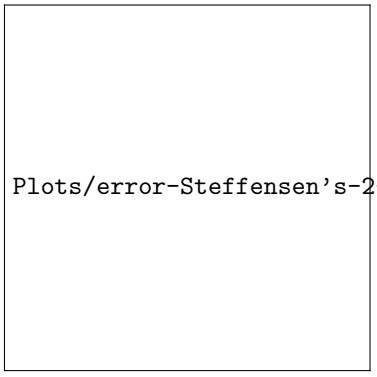
Figure 6: Iteration Results (Modified Newton's (m¡d), $d = 5$, $x_0 = 4.0$)

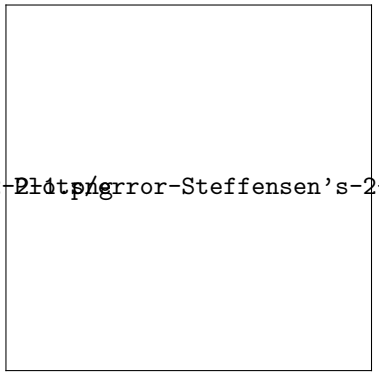
Modified Newton's method ($m < d$) converges when $d > 1$ (we note divergence in the plot where $d = 1$). As $d > 1$ increases, the number of iterations until convergence decreases (per initial input value x_0). Newton's method takes more iterations to converge when x_0 is farther from the actual root, but the overall number of iterations is similar for $x_0 = 2.5$ and $x_0 = 4.0$. We notice that the error plots again show linear convergence of the ratios for the error terms. We also point out that these plots look nearly identical to the plots when $m > d$ for Modified Newton's Method.

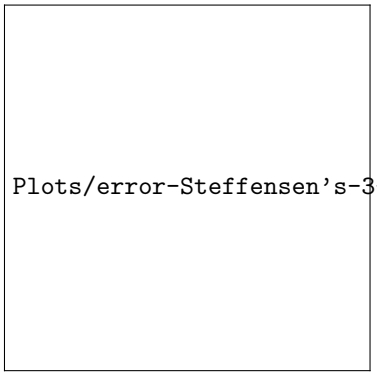
5. **Steffensen's Method**: Test Steffensen's method for initial values $x_0 = \{2.5, 4.0\}$.

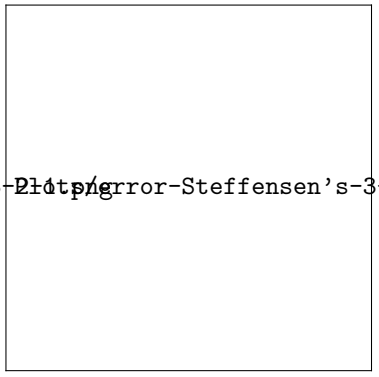


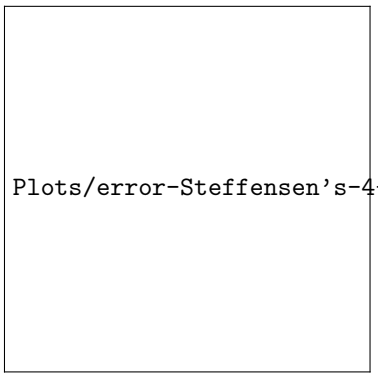


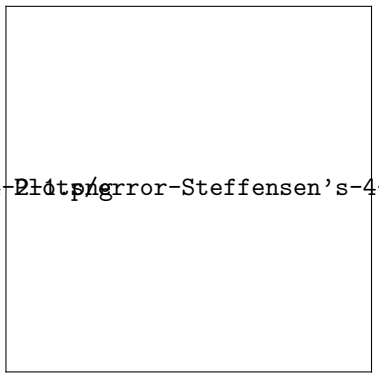


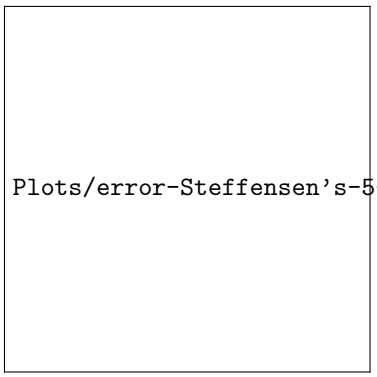







Plots/error-Steffensen's-4-4-1.png


Plots/error-Steffensen's-4-4-1.png


Plots/error-Steffensen's-5-4-1.png

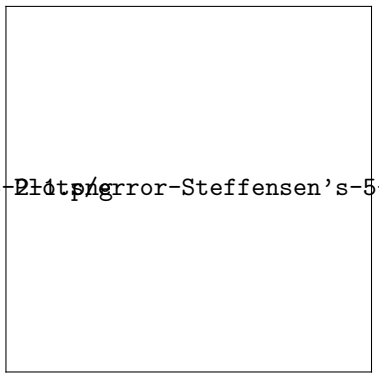

Plots/error-Steffensen's-5-4-1.png

table-Steffenson's-2-2-1.csv

```
Iteration,x_k,f(x_k)
1,2.3,0.09
2,2.16957,0.02875
3,2.09141,0.00836
4,2.0477,0.00228
5,2.02441,0.0006
6,2.01235,0.00015
7,2.00621,4e-05
```

Figure 7: Iteration Results (Steffensen's, $d = 2$, $x_0 = 2.5$)

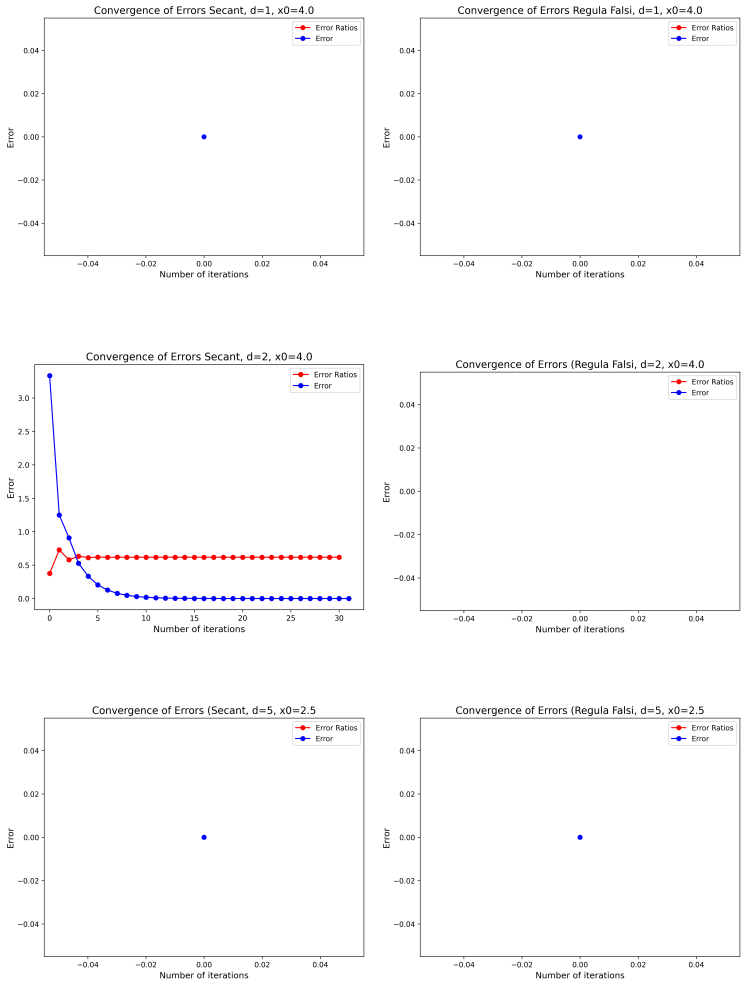
table-Steffenson's-2-4-1.csv

```
Iteration,x_k,f(x_k)
1,3.5,2.25
2,3.07143,1.14796
3,2.72259,0.52214
4,2.45719,0.20902
5,2.27112,0.07351
6,2.15175,0.02303
7,2.08122,0.0066
8,2.0422,0.00178
9,2.02153,0.00046
10,2.01088,0.00012
11,2.00547,3e-05
```

Figure 8: Iteration Results (Steffensen's, $d = 2$, $x_0 = 4.0$)

Steffensen's method converges for all multiplicities d. We note one-step convegrence when $d = 1$. As d increases, the number of iterations until convergence increases (per initial input value x_0). When the initial guess is close to the root, as is $x_0 = 2.5$, we see the ratios of the error terms converge nearly linearly and in fewer iterations per multiplicity. When $x_0 = 4.0$, we notice that the number of iterations increase nearly exponentially as d increases, but we still continue to observe nearly linear convergence of the error ratios.

6. **Regula Falsi and Secant Method**: Test the Regula Falsi and Secant methods for initial values on the intervals $[x_0, x_1] = [-2.5, 2.5], [-3.0, 4.0]$. We compare the plots side by side.



For the Regula Falsi and Secant methods, we observe one-step convergence when $d = 1$ and $d = 5$. This result may be attributed to the odd powers of the function. We note an interesting result when $d = 2$, however. For initial value $x_0 = 4.0$, Secant converges in 32 iterations, whereas Regula Falsi does not run at all. This is due to the nature of the Regula Falsi algorithm. The method guarantees convergence only when $f(x)$ has opposite signs at the ends of the interval. Since the function $f(x) = (x-2)^2$ is always non-negative, $f(x_0)$ and $f(x_1)$ will always have the same sign, regardless of initial values of x_0 or x_1.

4.2 Three Distinct Roots

We next observe the problem

$$f(x) = x^3 - \rho^2 x, \tag{9}$$

where we again set $\rho = 2$. We note that this problem has three distinct roots:

$$x^* = 0, 2, -2$$

and a derivative of the form

$$f'(x) = 3x^2 - \rho^2 = 3(x - \alpha)(x + \alpha), \quad \alpha = \frac{2}{\sqrt{3}}. \tag{10}$$

We examine the behavior of Newton's method on intervals of the form

$$\begin{aligned} x > \rho \quad &\longrightarrow \quad x > 2, \\ \alpha < x < \rho \quad &\longrightarrow \quad \frac{2}{\sqrt{3}} < x < 2, \\ -\xi < x < \xi \quad &\longrightarrow \quad -\frac{2}{\sqrt{5}} < x < \frac{2}{\sqrt{5}} \end{aligned}$$

where $\xi = \frac{2}{\sqrt{5}}$ is the "cycling point" for Newton's method for this particular $f(x)$ and ρ. For our experiments, we will thus examine the behavior of Newton's method for initial values in the set $x_0 = \{3.0, \frac{\rho+\alpha}{2}, \xi - \epsilon\}$, where $\epsilon = 10^{-6}$. We will compare these results to results from the other methods. We demonstrate the following results:

1. **Standard Newton's Method**:

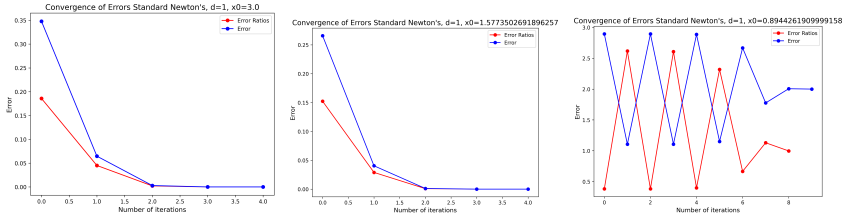


table-Standard Newton's-1-3-2.csv

	Iteration	x_k	f(x_k)
1	1	2.34783	3.55059
2	2	2.06461	0.54223
3	3	2.00291	0.02335
4	4	2.00001	5e-05
5	5	2.0	0.0

Figure 9: Iteration Results (Standard Newton's, $x_0 = 3$)

table-Standard Newton's-1-1-2.csv

	Iteration	x_k	f(x_k)
1	1	2.26581	2.56921
2	2	2.04048	0.33371
3	3	2.00117	0.0094
4	4	2.0	1e-05
5	5	2.0	0.0

Figure 10: Iteration Results (Standard Newton's, $x_0 = \frac{\rho+\alpha}{2}$)

Our results show convergence for Standard Newton's method for all three initial values x_0 tested. When $x_0 > \rho$ and $\alpha < x_0 < \rho$, we examine the convergence to the root $x^* = 2$. In the third plot, however, we examine that the errors terms and error ratio terms demonstrate cyclic behavior when we take $x_0 \approx \xi$. Newton's method exhibits sensitivity near the cycling point ξ. Since convergence is not guaranteed at this point, the method results in an oscillation of the error terms.

2. **Steffensen's Method**: We again test initial values $x_0 = \{3.0, \frac{\rho+\alpha}{2}, \xi+\epsilon\}$.

Plots/error-Steffensen's-1-3-2.png	Plots/error-Steffensen's-1-1-2.png	Plots/error-Steffensen's-1-0-2.png

table-Steffenson's-1-1-2.csv

```
Iteration,x_k,f(x_k)
1,0.45958,-1.74124
2,-0.17704,0.70261
3,0.00857,-0.03427
4,-0.0,0.0
5,0.0,-0.0
```

Figure 11: Iteration Results (Steffensen's, $x_0 = \frac{\rho+\alpha}{2}$)

Our results show Steffensen's method converges for all three initial values x_0 tested. When $x_0 > \rho$, we examine superlinear (quadratic) convergence of the error ratio terms, and we see the method requires more iterations until convergence than when $\alpha < x_0 < \rho$ or $-\xi < x_0 < \xi$. The table shows that, for initial guess $x_0 = \frac{\rho+\alpha}{2}$, Steffensen's method converges to the root $x_0 = 0$ (whereas Newton's method converges to the root $x_0 = 2$ for the same initial x_0).

3. **Regula Falsi Method**: For testing of the Regula Falsi method, we observe initial values on the intervals $[x_0, x_1] = [\frac{\rho+\alpha}{2}, 3.0], [\alpha, 4.0]$.

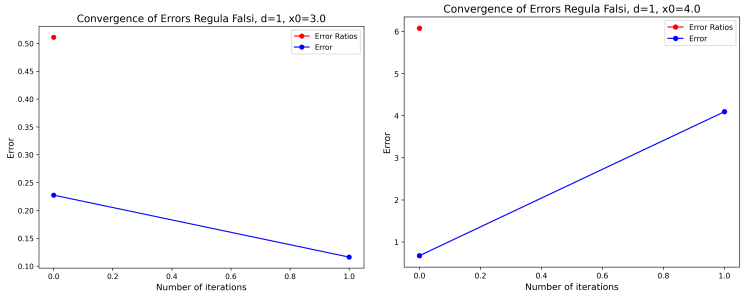


table-Regula Falsi-1-3-2.csv

1	Iteration,x_k,f(x_k)
2	1,1.77251,-1.52117
3	2,2.11622,1.01242

Figure 12: Iteration Results (Regula Falsi, $[x_0, x_1] = [\frac{\rho+\alpha}{2}, 3.0]$)

table-Regula Falsi-1-4-2.csv

1	Iteration,x_k,f(x_k)
2	1,1.32622,-2.97224
3	2,6.09254,201.77926

Figure 13: Iteration Results (Regula Falsi, $[x_0, x_1] = [\alpha, 4.0]$)

For the Regula Falsi method, we notice that the first interval tested, $[x_0, x_1] = [\frac{\rho+\alpha}{2}, 3.0]$, converges to the root $x^* = 2$. We notice divergence in the second interval tested, $[x_0, x_1] = [\alpha, 4.0]$. Although the algorithm stops after two iterations, we notice that $f(x_k)$ does not approach 0 and the x_k term does not converge to a root of $f(x)$.

4. **Secant Method**: As with Regula Falsi, we observe the perfomance of the Secant method on initial intervals $[x_0, x_1] = [\frac{\rho+\alpha}{2}, 3.0], [\alpha, 4.0]$.

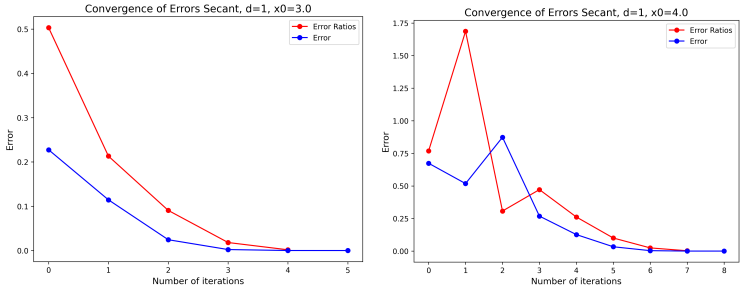
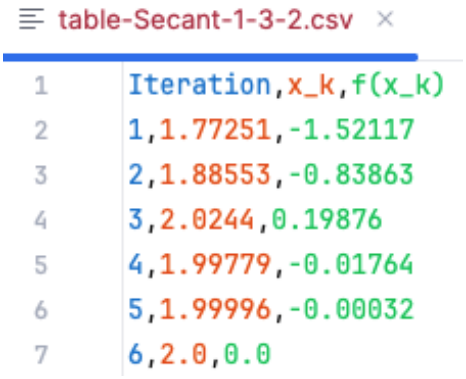


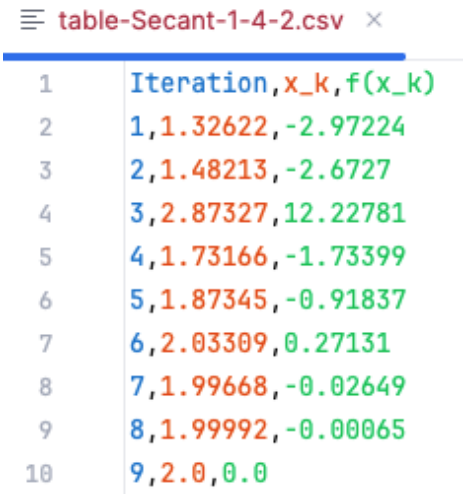
table-Secant-1-3-2.csv



```
Iteration,x_k,f(x_k)
1,1.77251,-1.52117
2,1.88553,-0.83863
3,2.0244,0.19876
4,1.99779,-0.01764
5,1.99996,-0.00032
6,2.0,0.0
```

Figure 14: Iteration Results (Secant, $[x_0, x_1] = [\frac{\rho+\alpha}{2}, 3.0]$)

table-Secant-1-4-2.csv



```
Iteration,x_k,f(x_k)
1,1.32622,-2.97224
2,1.48213,-2.6727
3,2.87327,12.22781
4,1.73166,-1.73399
5,1.87345,-0.91837
6,2.03309,0.27131
7,1.99668,-0.02649
8,1.99992,-0.00065
9,2.0,0.0
```

Figure 15: Iteration Results (Secant, $[x_0, x_1] = [\alpha, 4.0]$)

We observe the Secant method converges to the root $x^* = 2$ for both intervals tested.

4.3 Scaling

Recall that Newton's method for finding a root of a function $f(x)$ is given by the iteration:

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

If we consider a scaled function $\tilde{f}(x) = \sigma \cdot f(x)$, where $\sigma \neq 0$, then the derivative is

$$\tilde{f}'(x) = \sigma \cdot f'(x).$$

Applying Newton's method to $\tilde{f}(x)$ yields

$$x_{n+1} = x_n - \frac{g(x_n)}{g'(x_n)} = x_n - \frac{\sigma f(x_n)}{\sigma f'(x_n)} = x_n - \frac{f(x_n)}{f'(x_n)},$$

which is identical to the original iteration for $f(x)$. Theoretically, our results should remain unchanged. We examine the function in the previous section, where $f(x) = x^3 - \rho^2 x$ with $\rho = 2$. We test values $\sigma = [2, 5, 10]$ on the initial guess $x_0 = 3$. Our results follow.

1. **Standard Newton's Method:**

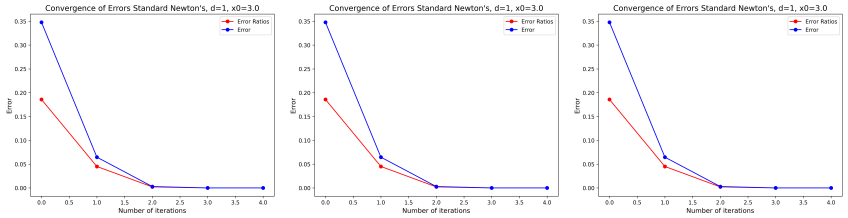


Figure 16: $\sigma = 2, 5, 10$

Our results align perfectly with our theoretical hypotheses. The plots remain identical for each value of σ. Therefore we observe no change in the number of iterations until convergence or changes in the error terms when scaling $f(x)$. We observe the same result for all other methods tested.

4.4 Root Coalescing

For the final series of tests, we investigate behavior of the Newton's and Secant methods on the function

$$f(x) = x(x - \rho_1)(x - \rho_2) \tag{11}$$

for different values of ρ_1, ρ_2. We will use the initial guess $x_0 = 2.0$ for the following tests of Newton's Method. For the Secant method, we will use the initial interval $[x_0, x_1] = [0.5, 2.0]$ for testing. Consider the following results:

1. **Standard Newton's Method (Varying ρ_1):** We want to observe the behavior of Newton's as $\rho_1 \to 0$. We set $\rho_2 = 1.5$ and examine the values $\rho_1 = [1.0, 0.1, 0.01, 0.001]$.

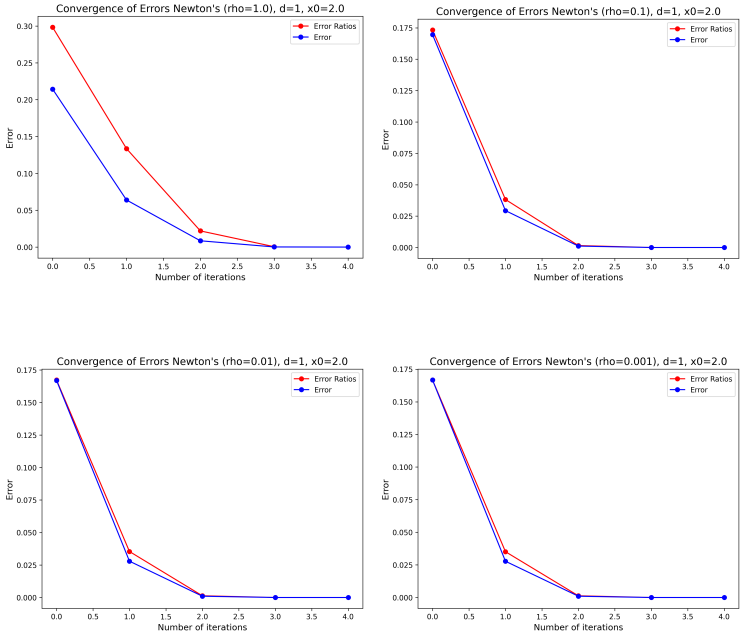


table-Newton's (rho=1.0)-1-2-3.csv

	Iteration	x_k	f(x_k)
1	1	1.71429	0.26239
2	2	1.56391	0.05636
3	3	1.50854	0.00655
4	4	1.50019	0.00014
5	5	1.5	0.0

Figure 17: Iteration Results (Standard Newton's, $\rho_1 = 1.0$)

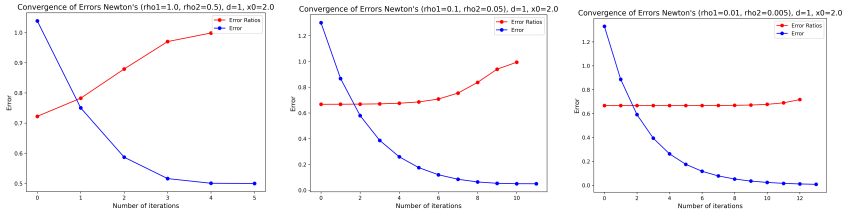
table-Newton's (rho=0.001)-1-2-3.csv

	Iteration	x_k	f(x_k)
1	1	1.66669	0.46278
2	2	1.52779	0.06483
3	3	1.50098	0.0022
4	4	1.5	0.0
5	5	1.5	0.0

Figure 18: Iteration Results (Standard Newton's, $\rho_1 = 0.001$)

When we vary ρ_1, we examine sublinear convergence of the error terms and the ratios of the error terms for Standard Newton's method. We also observe convergence in few iterations (≈ 5) and we see the errors and error ratios remaining close in value per iteration, particularly when we choose $\rho_1 \leq .1$. As $\rho_1 \to 0$, the function $f(x)$ almost develops a "double root" at $x = 0$. As a result, we see the terms converge to ρ_2, i.e. we observe $x_k \to \rho_2 = 1.5$.

2. **Standard Newton's Method (Varying ρ_1 and ρ_2):** We want to observe the behavior of Newton's as $\rho_1 \to 0$ and $\rho_2 \to 0$. We examine the values $(\rho_1, \rho_2) = (1.0, .5), (.1, .05), (.01, .005)$.



When we vary ρ_1 and ρ_2, we again see the error terms follow a similar pattern of convergence for Standard Newton's Method. However, we observe the error ratios converge more linearly as ρ_1 and ρ_2 approach 0. As $\rho_1 \to 0$ and $\rho_2 \to 0$, the function $f(x)$ becomes $f(x) \approx x^3$, where $x^* = 0$ is the

only root of the equation. We therefore observe much slower convergence for these parameters, compared to when we only vary ρ_1.

3. **Secant Method:** We perform the previous tests (varying only ρ_1 and varying ρ_1 and ρ_2) using the Secant method. All parameters (roots) tested resulted in a standard Python output error:

```
Traceback (most recent call last):
  File "/Users/jennypetrova/Desktop/PyCharm/FCM/project4/main.py", line 314, in <module>
    root_secant, iter_secant, results_secant, errs_secant = secant_method(f, x0, x1, rho2)
  File "/Users/jennypetrova/Desktop/PyCharm/FCM/project4/main.py", line 46, in secant_method
    q = ( f(x1) - f(x0) ) / (x1 - x0)
ZeroDivisionError: float division by zero
```

We see $x_1 - x_0 \to 0$ in the iteration steps for this function $f(x)$ when the Secant method is applied.

5 Conclusions

We observed the performance of (Standard) Newton's method on three different function $f(x)$, testing the behavior of this algorithm for several multiplicities and varied root values. We compared the performance of Newton's method to the Modified Newton's, Steffensen's, Secant, and Regula Falsi root-finding methods. We conclude that Newton's method demonstrates high-performance, generally requiring few iterations until convergence. We also generally observed linear convergence of the error ratios for the error terms, although theoretical analysis suggests quadratic convergence is to be observed. We may extend these tests to examine the performance of Newton's method when our initial guesses x_0 are much farther from the true root since, in practice, we will not know the value of the root prior to using the method. We may also test the behavior of Newton's method on more complicated functions, to better examine the degree of precision and performance of this algorithm.

References

[1] Alfio Quarteroni and Fausto Saleri, "Numerical Methods for Scientists and Engineers," *Springer*, 2000. https://link.springer.com/book/10.1007/978-3-662-04166-0, [Online; accessed 9-December-2024].

[2] Dr. Kyle A. Gallivan, "Set 13: Nonlinear Equations: Part 3", Class Lecture, MAD5403: Foundations of Computational Mathematics I, Florida State University, December 2024.

[3] Dr. Kyle A. Gallivan, "Set 14: Nonlinear Equations: Part 4", Class Lecture, MAD5403: Foundations of Computational Mathematics I, Florida State University, December 2024.

[4] Wikipedia contributors, "Rate of convergence," *Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/wiki/Rate_of_convergence, [Online; accessed 9-December-2024].

[5] Wikipedia contributors, "Root-finding algorithm," *Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/wiki/Root-finding_algorithm, [Online; accessed 9-December-2024].

[6] Wikipedia contributors, "Newton's Method," *Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/wiki/Newton's_method, [Online; accessed 9-December-2024].

[7] Wikipedia contributors, "Steffensen's Method," *Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/wiki/Steffensen's_method, [Online; accessed 9-December-2024].

[8] Wikipedia contributors, "Regula falsi," *Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/wiki/Regula_falsi, [Online; accessed 9-December-2024].

[9] Wikipedia contributors, "Secant method," *Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/wiki/Secant_method, [Online; accessed 9-December-2024].