ITERATIVE ROOT-FINDING METHODS FOR POLYNOMIALS

DARREN YAO, KARTIKESH MISHRA, BRYCE HANCOCK

Abstract. In this paper, we discuss Newton's method, the bisection method, and the secant method for estimating roots of polynomials by comparing convergence rates and reliability through computational and theoretical analysis. Newton's method exhibits quadratic order of convergence, which is the fastest of the three methods, but lacks robustness compared to the reliable bisection method which exhibits linear convergence. The secant method depends on function conditions for golden ratio convergence, striking a balance between speed and reliability. This examination provides insight into selecting the optimal numerical root-finding approach given speed, accuracy, and convergence criteria. Finally, we present the Polynomial Factorization Procedure, an algorithm that combines Newton's method and the bisection method to find all roots for any general polynomial.

1. Introduction

A recurring problem in mathematics is to find the roots of polynomials. Computer software systems usually look for roots numerically rather than symbolically, through iterative estimation methods that converge closer and closer to the actual root with each step. We can compare root-finding methods by how fast they converge and by their failure cases; often, choosing the appropriate method involves a trade-off between fast convergence and robustness (having minimal failure cases).

In this paper, we investigate several common iterative root-finding methods for polynomials in one variable: Newton's method, the secant method, and the bisection method. For each of these methods, we analyze the parameters for initial guesses that may cause them to fail to converge, the relation between input parameters and roots they converge to, and when they converge, their order of convergence.

First, we will begin with a simple example to demonstrate the three methods and intuitively describe what the three methods seek to accomplish. For consistency, we will be using the same function for all three initial examples: $f(x) = \frac{1}{8}x^2 - \frac{1}{2}$. The true root we are trying to find through these algorithms, which we will call α, equals 2.

1.1. Newton's Method.

Newton's method accepts an initial guess, x_0. Then, it evaluates the function, $f(x_0)$, and its derivative, $f'(x_0)$, and uses these values to draw a tangent line to the curve at the point $(x_0, f(x_0))$. The algorithm's following guess, which we will call x_n (for integers $n > 0$), is the x coordinate where this tangent line crosses the x-axis. We repeat the process until $f(x_n)$ is sufficiently close to 0. Figure 1 and Table 1 show the iterations of Newton's method on our example polynomial, with the process terminating when the error term, defined to be $e_n = x_n - \alpha$, is less than 10^{-5} in absolute value.

Table 1. Newton's Method Iterations for $f(x) = \frac{1}{8}x^2 - \frac{1}{2}$

Step	x_n	α	e_n	$f(x_n)$
0	4	2	2	1.5
1	2.5	2	0.5	0.281250
2	2.05	2	0.05	0.025312
3	2.000610	2	0.000610	0.000305
4	2.000000	2	0.000000	0.000000

Date: November 2023.

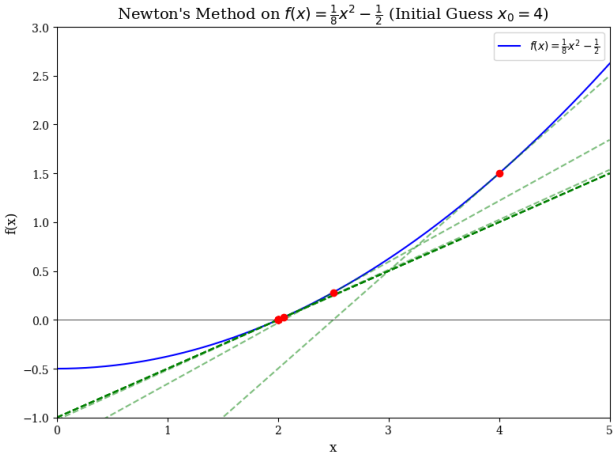


Figure 1. Example of Successful Convergence of Newton's Method

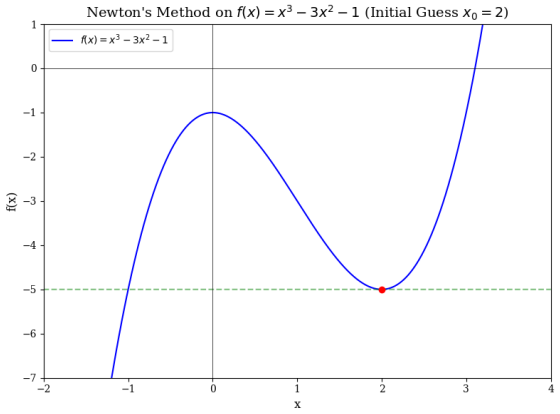


Figure 2. Example of Newton Method Failure: $f'(x_0) = 0$

We immediately notice that, since Newton's method requires evaluating $f'(x_n)$ at every step in order to draw the tangent line, the method will fail if, at any point, $f'(x_n)$ equals zero. Visually, $f'(x_n)$ equalling zero would cause the tangent line to run parallel to the x-axis, as seen in the example in Figure 2. This is an example of a poor initial guess that causes a failure of the method.

1.2. **Secant Method.** The Secant method is another root-finding algorithm that uses a sequence of values to approximate a root of a function. Unlike Newton's method, which requires the evaluation of the derivative of the function, the Secant method uses two initial guesses, x_0 and x_1, to approximate the derivative by a secant line. At each step n, the function is evaluated at two points, $x_n - 1$ and x_n, and the secant line through these points is computed. The x-intercept of this line is then used as the next approximation to the root. This process is repeated until the change between consecutive approximations is sufficiently small. Figure 3 and Table 2 display the iterations of the Secant method applied to our example polynomial. The iterations terminate when e_n is less than 10^{-5} in absolute value.

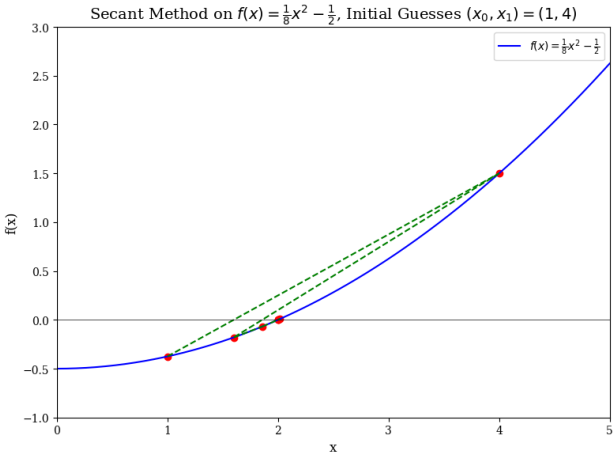


Figure 3. Example of Successful Convergence of Secant Method

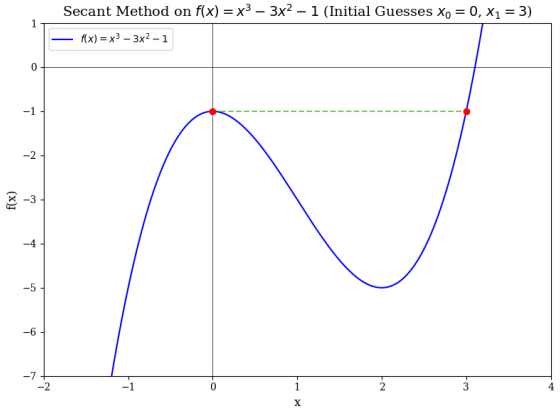


Figure 4. Example of Secant Method Failure: $f(x_0) = f(x_1)$

Similarly to Newton's method, the secant method will fail when the secant line connecting $x_n - 1$ and x_n fails to cross the x-axis. This will occur when $f(x_n - 1)$ equals $f(x_n)$. An example of this failure is illustrated in Figure 4, which is again caused by the initial guess.

Table 2. Secant Method Iterations for $f(x) = \frac{1}{8}x^2 - \frac{1}{2}$

Step	x_n	x_{n+1}	α	e_n	$f(x_n)$
0	1	4	2	-1	-0.375
1	4	1.6	2	2	1.5
2	1.6	1.857143	2	-0.4	-0.18
3	1.857143	2.016529	2	-0.142857	-0.068878
4	2.016529	1.999390	2	0.016529	0.008299
5	1.999390	1.999997	2	-0.000610	-0.000305
6	1.999997	2.000000	2	-0.000003	-0.000001

1.3. **Bisection Method.** The Bisection method is the third root-finding algorithm we choose to analyze. The Bisection method is another root-finding technique that iteratively narrows down the interval where a root of the function must lie. It starts with an initial interval $[l_0, r_0]$ that brackets the root, requiring that the function values at these points have opposite signs. At each step, the method calculates the midpoint x_n of the interval $[l_n, r_n]$ and evaluates the function at this point. Comparing the sign of the midpoint to the signs of the endpoints determines which half of the interval contains the root. The process is repeated with the new interval until the midpoint is sufficiently close to the actual root, and in this example, that is when e_n is less than 10^{-5} in absolute value. Figure 5 and Table 3 illustrate the Bisection method applied to the function $f(x) = \frac{1}{8}x^2 - \frac{1}{2}$.

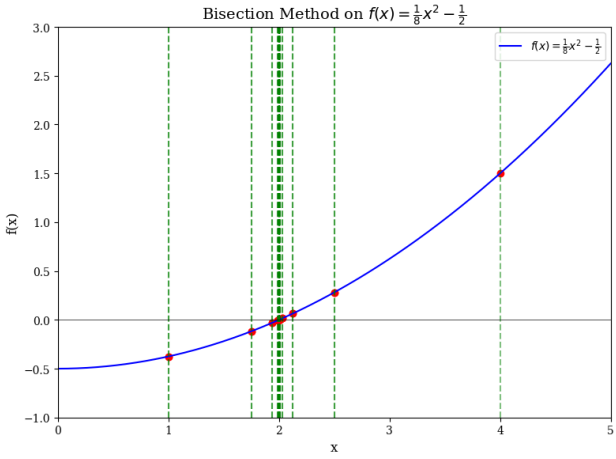


Figure 5. Bisection Method Iterations with Initial Interval $[1, 4]$

Table 3. Bisection Method Iterations for $f(x) = \frac{1}{8}x^2 - \frac{1}{2}$

Step	l_n	r_n	α	e_n	$f(x_n)$
0	1	4	2	0.5	0.281250
1	1	2.5	2	-0.25	-0.117188
2	1.75	2.5	2	0.125	0.064453
3	1.75	2.125	2	-0.0625	-0.030762
4	1.9375	2.125	2	0.031250	0.015747
5	1.9375	2.031250	2	-0.015625	-0.007782
6	1.984375	2.031250	2	0.007812	0.003914
7	1.984375	2.007812	2	-0.003906	-0.001951
8	1.996094	2.007812	2	0.001953	0.000977
9	1.996094	2.001953	2	-0.000977	-0.000488
10	1.999023	2.001953	2	0.000488	0.000244
11	1.999023	2.000488	2	-0.000244	-0.000122
12	1.999756	2.000488	2	0.000122	0.000061
13	1.999756	2.000122	2	-0.000061	-0.000031
14	1.999939	2.000122	2	0.000031	0.000015
15	1.999939	2.000031	2	-0.000015	-0.000008
16	1.999985	2.000031	2	0.000008	0.000004

We find that Newton's method generally converges fastest, although in some cases it can converge slowly, oscillate, or diverge entirely. One must ensure that precise conditions for the initial guess are met in order to guarantee convergence to α using Newton's method. The secant method converges slower than Newton's method, but has error cases that are easily described and avoided, and it is more computationally efficient than Newton's method. The bisection method exhibits the slowest convergence, but only fails in the specific case where all the roots have even multiplicity, and is therefore the most robust method. Our results can be described through a comparison of properties of the three methods:

Method	Order of convergence	Robustness	Computational Difficulty
Newton	2 (fastest, with exceptions)	Many failure cases	Requires computing derivatives
Secant	$\frac{1+\sqrt{5}}{2} \approx 1.6$	Less failure cases	No derivatives required
Bisection	1 (slowest)	Very few failure cases	No derivatives required

We formally define *order of convergence*, a metric that describes how fast an estimate converges to a true value (in this case, the root).

Definition 1. *A sequence x_n that converges to a value α has* ***order of convergence*** *q if*

$$\lim_{n\to\infty} \frac{|x_{n+1} - \alpha|}{|x_n - \alpha|^q} = \mu$$

for some constant μ, called the asymptotic error constant.

For example, an estimate that converges with order 1 should gain a constant number of accurate digits per iteration. An estimate that converges with order 2 should increase the number of accurate digits by a factor of μ per iteration.

One weakness of all three methods is that they do not easily identify polynomials that have no roots. In addition, because the purpose of these algorithms is to find one root, they give us no global information on how many roots the polynomial has. If one requires approximations to all roots of a function, none of these algorithms alone would be effective in their current state, since they are all methods to solve for one local root of the function. Furthermore, there are particular conditions for the initial guesses for Newton's method and the secant method that must be met to ensure convergence. To address these issues, we combine Newton's Method and the bisection Method to come up with an algorithm to factorize polynomials. That is, a procedure that estimates all roots of a polynomial and works for all polynomials.

The rest of the paper is structured as follows: we analyze Newton's Method in section (3), the Secant method in section (4) and the Bisection method in section (5) and finally we present the Polynomial Factorization Procedure in section (6).

2. Polynomials and Roots

In this section, we're laying the groundwork to analyze various root-finding methods by uncovering the essential properties of polynomials and their roots. To start, we'll establish a fundamental principle: roots remain unchanged when we scale a polynomial and simultaneously translate the independent variable. This property allows us to determine the magnitude boundaries of the roots. Understanding these bounds aids in narrowing down our search area while employing root-finding algorithms. Following this, we'll delve into exploring the relationship between critical points and roots. The rationale behind this exploration is simple: critical points, being roots of the polynomial's first derivative, hold lower complexity compared to the original polynomial, making them efficient tools for locating roots.

To be precise, imagine a polynomial, p, having roots $\alpha_1, \ldots, \alpha_k$. Now, if we transform this polynomial, let's call it q, by scaling it with constants a, b, and c where $a, c \neq 0$, in the form, $q(x) = c * p(ax + b)$. A quick observation is that for any root α of p, $q(\frac{\alpha-b}{a}) = cp(\alpha) = 0 \implies \frac{\alpha-b}{a}$ is a root of q. Thus, $\frac{\alpha_1-b}{a}, \ldots, \frac{\alpha_k-b}{a}$ are roots of q. It's important to note that the roots remain unaffected by c, suggesting our analysis should focus on polynomials with a leading coefficient of 1. Leveraging this invariance, we can calculate the boundaries for root magnitudes, illustrated in the subsequent proposition.

Proposition 1 (Bounds on Root Magnitudes)**.** *Given a polynomial $p(x) = \sum_{r=0}^{n} a_r x^r$, all of its roots have magnitude within bound $(-M_p, M_p)$ where $M_p = \max\{1, \sum_{r=0}^{n-1} |\frac{a_r}{a_n}|\}$*

Proof. Consider the polynomial, $q(x) = x^n + \sum_{r=0}^{n-1} b_r x^r$ where $b_r = \frac{a_r}{a_n}$. Then q has exactly the same root as p, so it will be enough to prove the bounds on the polynomial q. For that, take any root α of q,

$$q(\alpha) = 0 \implies |\alpha^n| = \left| \sum_{r=0}^{n-1} b_r \alpha^r \right| \leq \sum_{r=0}^{n-1} |b_r \alpha^r|$$

using triangle inequality. Now, assume $|\alpha| \geq 1$ then $|\alpha^u| \leq |\alpha^v|$ whenever $v \geq u \geq 0$. This leads to

$$|\alpha^n| \leq |\alpha^{n-1}| \sum_{r=0}^{n-1} |b_r| \implies |\alpha| \leq \sum_{r=0}^{n-1} |b_r| = \sum_{r=0}^{n-1} |\frac{a_r}{a_n}|.$$

Thus, either $|\alpha| < 1$ or $|\alpha| \leq \sum_{r=0}^{n-1} |\frac{a_r}{a_n}|$ which yields $|\alpha| \leq \max\{1, \sum_{r=0}^{n-1} |\frac{a_r}{a_n}|\} = M_p$ as desired. □

This proposition gives us a limit on where all the roots can be located. So instead of having to search between negative and positive infinity, we now have a much smaller area to look in. Next, we will focus on finding a closer link between where the critical points are and where the roots are. We will start by showing that the limit we found before is valid for all the critical points too.

Corollary 1. *All critical points are also bounded within* $[-M_p, M_p]$.

Proof. Given a polynomial p, $p(x) = \sum_{r=0}^{n} a_r x^r \implies p'(x) = \sum_{r=1}^{n} r a_r x^{r-1}$ which is also a polynomial. Thus by proposition (1), all critical points are bounded within $[-M_{p'}, M_{p'}]$ where

$$M_{p'} = \max\{1, \sum_{r=1}^{n-1} |\frac{r a_r}{n a_n}|\} \leq \max\{1, \sum_{r=1}^{n-1} |\frac{a_r}{a_n}|\} \leq \max\{1, \sum_{r=0}^{n-1} |\frac{a_r}{a_n}|\} = M_p$$

as desired. □

We have shown that critical points have the same limit as roots. But more interestingly, we can show that roots and critical points actually mark the limits for each other. This will help us narrow down our search area even further when we're looking for a particular root.
For now, remember that a polynomial of degree n has at most n real roots and $n-1$ critical points. Building on that fact, the next statement shows a similar link between the minimum numbers of roots and critical points. By connecting these ideas, we are developing a clearer picture of how roots and critical points relate to and constrain each other.

Proposition 2. *There is always a critical point between any two roots of the polynomial p(x). Moreover, if p(x) has k roots then it must have at least* $k-1$ *critical points.*

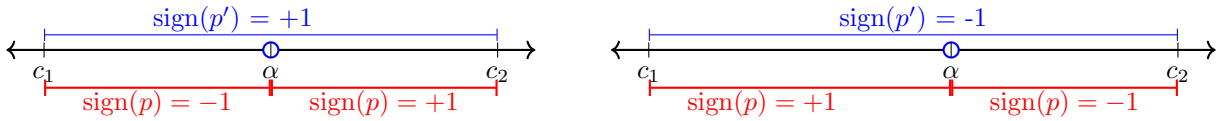
Proof. This can be seen from the Mean value theorem that there exists a $c \in [a, b] : f'(c) = \frac{f(b)-f(a)}{b-a}$ if f is continuous on $[a, b]$ and differentiable on (a, b). Since all polynomials are continuously differentiable, suppose a and b are two roots with $b \neq a$ then $\exists c \in [a, b] : p'(c) = \frac{p(b)-p(a)}{b-a} = 0$. Hence, there is indeed a critical point between any two roots. Now, if there are k roots then between each of two consecutive roots, there is a critical point, thus proving the desired result of having at least k-1 critical points. □

This proposition helps us in estimating the number of critical points based on the number of roots but the reverse may not be true as there can be no roots and still some critical points. Nonetheless, an estimate of the number of roots based on critical points can still be made as in the following corollary.

Corollary 2. *There can be at most one root between two consecutive critical points (CCPs).*

Proof. Suppose there is more than one root between two consecutive critical points c_1 and c_2. But then by proposition (2), there is at least one critical point between c_1 and c_2 which contradicts c_1 and c_2 being consecutive critical points. □

With this corollary and the proposition (2) presented above, roots and critical points indeed limit each other, while the former always holds, the latter may not as some CCPs may not contain a root. To determine if two CCPs contain a root or not, we can use the Intermediate Value Theorem which suggests the following fact that a continuous function changes sign only at the root. Using this fact, we can establish the relation between the signs of a polynomial and its derivative within two CCPs. This is crucial in deducing the monotonicity of polynomials and the functions related to it.



Proposition 3. *Given an interval of two CCPs (c_1, c_2) containing a root α, $p(x)$ and $p'(x)$ have different signs within $[c_1, \alpha)$ and the same sign within $(\alpha, c_2]$.*

Proof. By the intermediate value theorem, as we can see from the figure above p changes sign at α and p' has a constant sign in the interval $[c_1, c_2]$. Thus, it suffices to show that p and p' have the same sign for any x in the interval $(\alpha, c_2]$. We can see this by the definition of p', for some $x = \alpha + \epsilon$ where $\epsilon > 0$, $p'(x) = \lim_{\epsilon \to 0} \frac{p(x)-p(\alpha)}{\epsilon}$. Since $p(\alpha) = 0$, it can be deduced that $p(x)$ and $p'(x)$ have the same sign. □

We established an important link between the signs of a polynomial and its derivative within an interval bounded by consecutive critical points. But what if a critical point is a root itself, note that the proposition (3) does not cover this case. In fact, the sign of the polynomial does not change at such root as if there were two roots at the same point, to maintain the consistency with the intermediate value theorem. As a final lemma of this section, we will prove indeed this is the case and in fact, it can be more generalized as seen in the following definition and lemma that follows.

Definition 2. *We say α is a root with multiplicity k if $p^{'l}(\alpha) = 0$ for all $l < k$ and $p^{'k}(\alpha) \neq 0$*

Lemma 1. *A root is with multiplicity k iff it is a k-time repeated root.*

Proof. For a root α is $k-$repeated then, $p(x) = (x-\alpha)^k q(x)$ where $q(\alpha) \neq 0$ then it can be quickly checked that $p^{'l}(x) = (x-\alpha)^{k-l} q_l(x)$ where $q_l(\alpha) \neq 0$ thus, α is a root with multiplicity k.

Now, For $k = 1$, suppose α is a root with multiplicity 1, then $p'(\alpha) \neq 0$. Since α is a root of $p(x)$, $p(x) = (x-\alpha)q(x)$ for some $q(x)$. Now, $p'(x) = (x-\alpha)q'(x) + q(x)$ then $q(\alpha) = p'(\alpha) \neq 0$ and hence, α is a single root of $p(x)$.
Now, assume instead the statement holds for all roots of any polynomial with multiplicity $m < n$. Take a root β of multiplicity n of polynomial p then, by definition, β is a root of multiplicity $n-1$ of polynomial p' thus,

$$p'(x) = (x-\beta)^{n-1} q(x)$$

for some polynomial q where $q(\beta) \neq 0$. Suppose β is an $r-$repeated root then,

$$p(x) = (x-\beta)^r s(x) \implies p'(x) = r(x-\beta)^{r-1} s(x) + (x-\beta)^r s'(x)$$

where $s(\beta) \neq 0$ thus, β is root of p' with multiplicity $r-1$ concluding $r = n$ and $q(x) = rs(x) + (x-\beta)s'(x)$. Thus, by mathematical induction, a root with multiplicity k is a $k-$repeated root.

□

3. Newton's method

Newton's method is a general algorithm for numerically approximating roots for any differentiable function. The approach involves initiating a presumed root and subsequently updating it iteratively by intersecting the tangent line at that point with the x-axis until a close approximation to the root is attained. More precisely, consider a function, f(x), and suppose x_k denotes the value after the kth iteration of Newton's method with an initial guess x_0. The sequence's successive values evolve by linking points $(x_k, f(x_k))$ and $(x_{k+1}, 0)$ with the tangent line derived from differentiation $f'(x)$.

$$f'(x_k) = \frac{f(x_k) - 0}{x_k - x_{k+1}} \implies x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

This recursive formula forms the core update step of Newton's method as can be seen in the following algorithm widely used to approximate roots with high precision:

Algorithm 1: Newton Method with Error Tolerance ϵ

Data: a function f(x), an initial guess x_0, $\epsilon \geq 0$
$x = x_0$
while $|f(x)| > \epsilon$ **do**
  $x \leftarrow x - \frac{f(x)}{f'(x)}$
Result: x

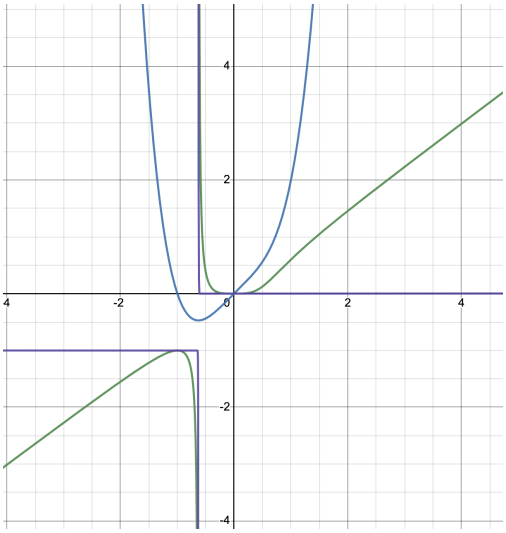


Figure 6. A polynomial $p(x) = x^4 + x$ in blue, $\Psi_p(x)$ is green and $\Psi_p^{10}(x)$ in purple showing convergence of Newton's sequence.

Building upon the groundwork laid in the previous section on polynomials and roots, particularly their interrelation with critical points, this section comprehensively analyzes Newton's method. Initially, we establish this method with an equivalent rational function (say Ψ) in terms of analysis. And then, we analyze its compositional sequence i.e. Ψ^n, their limitations in being defined over $\mathbb{R}$, and their remarkable strength in terms of rapid convergence.

Revisiting the recursive formula, let's define a rational function as follows:

Definition 3. *Given a function, f, we define Newton's function as* $\Psi_f : \mathbb{R} \to \mathbb{R}$ *s.t.* $\Psi_f(x) = x - \frac{f(x)}{f'(x)}$ *and the corresponding Newton's sequence as* $\Psi_f^n(x) = \Psi_f(\Psi_f^{n-1}(x))$ *for some x.*

Here, we note the trivial connection that Newton's Method is equivalent to finding convergence of this sequence i.e. it suggests the sequence $\Psi_f^n(x)$ converges to a root of the function f for some x. It can be visualized in the figure (6). We will now establish this property as a proposition for polynomials.

Proposition 4. *For any polynomial p, given some* x_0*, if* $x_n = \Psi_p^n(x_0) \to \alpha$ *then* α *is a root of the polynomial p.*

Proof. We know, since all polynomials are continuously differentiable, $x_n \to \alpha$ implies $p(x_n) \to p(\alpha)$ and $p'(x_n) \to p'(\alpha)$. These two sequences $p(x_n)$ and $p'(x_n)$ is related by the above Newton's function as follows,

$$p(x_n) = p'(x_n)y_n$$

where $y_n = x_{n+1} - x_n \to 0$ by the definition of convergence. This suggests,

$$p(x_n) = p'(x_n)(x_{n+1} - x_n) \to p'(\alpha) * 0 = 0$$

. Since this sequence is defined on real numbers, and a sequence in real numbers if converges, converges uniquely. But we have $p(x_n) \to p(\alpha)$ and $p(x_n) \to 0$. Thus, $p(\alpha) = 0$ as desired, and α is the root of the polynomial. □

This proposition shows that whenever the compositional sequence of Newton's function converges, it always converges to a root. However, it opens up questions regarding whether it converges for every point on a real line. The answer is no as can be seen as an asymptote in figure (6). Note that Newton's functions are not defined for critical points of the polynomial, let alone form a sequence to converge. Though a great setback with this method, we will show that this method is still very useful as it can reliably help find all roots, despite limitations on convergence for some inputs.

To prove the reliable convergence of Newton's method to any roots from properly defined starting points, we will first find specific conditions an interval around a root should satisfy to guarantee sure convergence of Newton's sequence. We will then show that around every root, such an interval exists.

Lemma 2. *Given a polynomial, p is monotonic and the newton's function Ψ_p is monotonically increasing in the interval $[a, b]$ which contains a root α then $\forall x \in (a, b), \Psi_p^n(x) \to \alpha$*

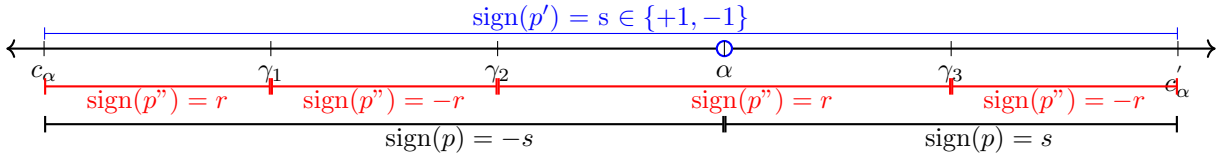
Proof. Here, since p is monotonic over the interval $[a, b]$, there cannot be any critical point in (a, b) as by definition, critical points break monotonicity. This implies the root α is unique and with multiplicity 1. Now, for any $x \in (a, \alpha)$, by proposition (3), we have $\frac{p(x)}{p'(x)} < 0 \implies \Psi_p(x) = x - \frac{p(x)}{p'(x)} > x$ and $\Psi_p(x) \leq \Psi_p(\alpha) = \alpha$ by monotonicity of Ψ_p. This implies $|\Psi_p(x) - \alpha| < |x - \alpha|$ and hence the sequence $\Psi_p^n(x)$ is monotonically increasing and bounded hence by monotone convergence theorem, it converges and by proposition (4), it converges to α.
Similarly for any $x \in (\alpha, b)$, by proposition (3), we have $\frac{p(x)}{p'(x)} > 0 \implies \Psi_p(x) = x - \frac{p(x)}{p'(x)} < x$ and $\Psi_p(x) \geq \Psi_p(\alpha) = \alpha$. Similar to above it can be deduced that the sequence $\Psi_p^n(x)$ is monotonically decreasing and bounded hence again by monotone convergence theorem, it converges and by proposition (4), it converges to α. □

Now we will prove one of the main usefulness of Newton's method which is for all roots, there is an interval that leads Newton's sequence towards it. The proof uses sign arguments from the proposition (3) to conclude the monotonicity necessary in the lemma (2).

Proposition 5. *Given a polynomial p, for any root α, there exists an open interval (a, b) with $a \neq b$ s.t. $\forall x \in (a, b)$, $\Psi_p^n(x) \to \alpha$*

Proof. Suppose the polynomial p has k distinct real roots, namely $\alpha_1, \alpha_2, ..., \alpha_k$ in increasing order. From proposition (2), we have at least one critical point between each root. This implies for any root α from $\alpha_2, ..., \alpha_{k-1}$, it is bounded by two consecutive critical points, i.e. $\alpha \in [c_\alpha, c'_\alpha] = I_\alpha$ (say) where c_α, c'_α are some consecutive critical points pair. Note that $p(x)$ is monotonic in I_α by construction. Using proposition (2) again, it can be deduced that there is at least one point, say $\gamma \in I_\alpha$ such that $p"(\gamma) = 0$.

Suppose there are r such points $\gamma_1, \ldots, \gamma_r$, partitioning $I_\alpha = [c_\alpha, \gamma_1] \cup \ldots \cup [\gamma_r, c'_\alpha]$. Using the intermediate value argument presented in proposition (3), there is a unique interval with a sign change for p i.e. which contains the root. Say it is $[\gamma_t, \gamma_{t+1}]$ where $t \in \{0, 1, \ldots, r\}$. For simplicity, let $\gamma_0 = c_\alpha$ and $\gamma_{r+1} = c'_\alpha$. Note, again by the same argument, $p"$ alternates sign as it moves from one interval to another. The sign change can be visualized as follows.



As in the figure above, recall p' has a constant sign throughout I_α WLOG it is $s \in \{+1, -1\}$. Then by proposition (3), p has sign $-s$ in the interval $[\gamma_t, \alpha)$ and sign s in the interval $(\alpha, \gamma_{t+1}]$.
Now, suppose $p"$ has sign s in the interval $[\gamma_t, \gamma_{t+1}]$ i.e. $s = r$ in the figure above then, p and $p"$ has the same sign in the interval $(\alpha, \gamma_{t+1}]$. Similarly, if $p"$ has sign $-s$ in the interval $[\gamma_t, \gamma_{t+1}]$ i.e. $r = -s$ in the figure above then, p and $p"$ has the same sign in the interval $[\gamma_t, \alpha)$. Thus, there exists a close interval within $[c_\alpha, c'_\alpha]$ for which p and $p"$ have the same sign. In such interval, p is monotonic, and, $p(x)$ and $p"(x)$ have the same sign implies $\Psi'_p(x) = \frac{p(x)p"(x)}{p'(x)^2} > 0$ and thus, Ψ_p is monotonically increasing. Hence, by lemma (2), there exists an open interval for which Newton's sequence converges to the root unless γ in the interval is the root but then for all (γ_t, γ_{t+1}), p and $p"$ have the same sign and thus Ψ_p is monotonic hence the required interval exist anyway.

For the roots α_1 and α_k, considering they are not bounded by CCPs, note that $\alpha_1 \in (-\infty, c_1)$ and $\alpha_k \in (c_2, \infty)$ where c_1 is the first and c_2 is the last critical point. If there is a point of inflection i.e. roots of $p"$ in $(-\infty, c_1)$ or (c_2, ∞) then the above argument can be applied otherwise note that p and $p"$ has same sign in

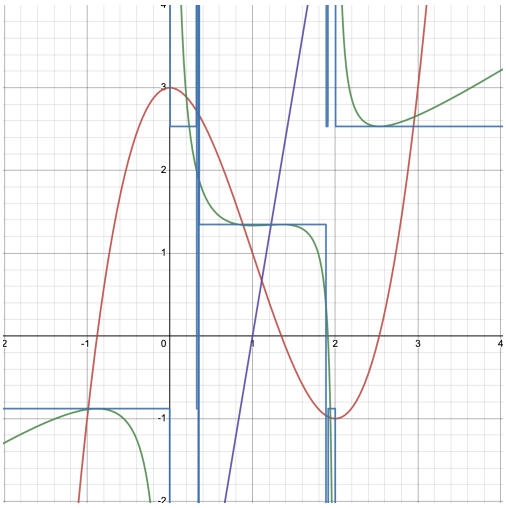


Figure 7. Polynomial $p(x) = (x-2)^2(x+1) - 1$ in red, p" in purple showing point of inflection at $x = 1$, Newton's function Ψ_p in green and Ψ_p^{10} in blue.

the intervals $(-\infty, \alpha_1)$ and (α_k, ∞) by applying proposition (3) for the root of polynomial and for a critical point as the root of the first derivative of the polynomial. □

Even though we have shown that for every root, there exists some initial guess guaranteeing convergence, as can be seen in figure, (7), there can be some complicated points other than critical points where Newton's sequence does not converge. One such class of points is cyclic points as can be seen in this example, take a polynomial, $p(x) = x^3 - 2x + 2$ then $\Psi_p(x) = x - \frac{x^3-2x+2}{3x^2-2}$ and $\Psi_p(0) = 1, \Psi_p(1) = 0$. Also, in the figure (7), the bound for points converging to the root at $x = 1.347$ is an orbit of cycle 2. Based on our observation and analysis, we propose some conjectures associated with the orbits for future continuation of the works in this paper.

Definition 4. *Precisely, A set of points $\{x_1, x_2, ..., x_k\}$ is called orbit of cycle k for polynomial p iff $\Psi_p(x_k) = x_1$ and for all $i \in [k-1], \Psi_p(x_i) = x_{i+1}$*

Conjecture 1. *Any maxima below the x-axis and any minima above the x-axis have a unique 2-cycle orbit.*

Conjecture 2. *There exists a 2-cycle orbit between two CCPs containing a root and they form a bound for all points that converge to the root.*

Despite all these setbacks, there is an even stronger reason than Proposition (1), to use this method. Newton's method converges incredibly fast, with an order of 2 for all roots with multiplicity 1 which is usually the case.

Theorem 1. *For any polynomial, when $x_n = \Psi_p^n(x_0) \to \alpha$ for some alpha then it converges with order 2 if α is 1st simple root, else with an order of 1.*

Proof. Let $e_n = x_n - \alpha$ then, $e_{n+1} = x_{n+1} - \alpha = x_n - \alpha - \frac{p(x_n)}{p'(x_n)} = e_n - \frac{p(x_n)}{p'(x_n)}$. Using Taylor expansion of p(x) near $x = \alpha$ gives us $p(x) = \sum_{r=1}^{d} \frac{p^{'r}(\alpha)}{r!}(x-\alpha)^r$ where d is the degree of the polynomial p(x) and starting from r=1 as $p^{'0}(\alpha) = p(\alpha) = 0$. This also implies $p'(x) = \sum_{r=1}^{d} \frac{p^{'r}(\alpha)}{(r-1)!}(x-\alpha)^{r-1}$ and thus,

$$e_{n+1} = e_n - \frac{\sum_{r=1}^{d} \frac{p^{'r}(\alpha)}{r!}(e_n)^r}{\sum_{r=1}^{d} \frac{p^{'r}(\alpha)}{(r-1)!}(e_n)^{r-1}}$$

Suppose now, α is kth simple root for some k then,

$$e_{n+1} = e_n - \frac{\sum_{r=k}^{d} \frac{p^{'r}(\alpha)}{r!}(e_n)^r}{\sum_{r=k}^{d} \frac{p^{'r}(\alpha)}{(r-1)!}(e_n)^{r-1}} \approx e_n - \frac{\frac{p^{'k}(\alpha)e_n^k}{k!} + \frac{p^{'k+1}(\alpha)e_n^{k+1}}{(k+1)!} + \mathcal{O}(e_n^{k+2})}{\frac{p^{'k}(\alpha)e_n^{k-1}}{(k-1)!} + \mathcal{O}(e_n^k)} \to e_n(1 - \frac{1}{k}) + e_n^2 \frac{p^{'k+1}(\alpha)}{k(k+1)p^{'k}(\alpha)}$$

as $n \to \infty$. The approximation is justified by, when n is large, $x_n \approx \alpha \implies \mathcal{O}(e_n^k) \approx 0$, i.e. higher power of e_n can be approximated by 0. This implies that, if $k = 1$ i.e. $p'(\alpha) \neq 0$ then, $\lim_{n\to\infty} \frac{|e_{n+1}|}{|e_n|^2} = \frac{p''(\alpha)}{2p'(\alpha)}$ otherwise, $\lim_{n\to\infty} \frac{|e_{n+1}|}{|e_n|} = 1 - \frac{1}{k}$, as desired. □

4. Secant Method

The Secant method offers a compromise between speed and simplicity by using a linear approximation to the derivative. Unlike Newton's method, the Secant method does not require the computation of derivatives, making it particularly useful for functions where the derivative is difficult to calculate.

4.1. Secant Method Formulation and Algorithm.

The Secant method is based on the formula:

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

This formula iteratively refines the approximation of the root by using the most recent two estimates. Unlike the Bisection method, it is not a requirement that $f(x_n) \cdot f(x_{n-1}) < 0$ for the method to converge, but it is required that $f(x_n) \neq f(x_{n-1})$ for all n. If this condition is not met, the secant line will not cross the x-axis, and the algorithm will fail due to division by zero; computer implementations will return an error. Below is the algorithm for the secant method:

Algorithm 2: Secant Method

Input: A continuous function $f(x)$, two initial guesses x_0 and x_1, and an error tolerance $\varepsilon \geq 0$
Output: An estimate of a root α such that $|f(\alpha)| < \varepsilon$
while $|f(x_1)| > \varepsilon$ **do**
  $x_{new} \leftarrow x_1 - f(x_1) \cdot \frac{x_1 - x_0}{f(x_1) - f(x_0)}$
  if $|f(x_{new})| < \varepsilon$ **then**
    return x_{new}
  `// Update the guesses for the next iteration`
  $x_0 \leftarrow x_1$
  $x_1 \leftarrow x_{new}$
return x_1

4.2. Order of Convergence of the Secant Method.

Theorem 2. *If the secant method converges for some function $f(x)$ which has nonzero, real first and second derivatives at its roots, then the order of convergence of the procedure is ϕ, the golden ratio.*

Proof. Consider x_n representing the sequence formed by the iterative secant method, which converges to a fixed root α. Taking $e_n = x_n - \alpha$, i.e. error in root approximation at the nth step, and using equation (1) we obtain,

$$e_{n+1} + \alpha = e_n + \alpha - f(x_n)\left(\frac{e_n + \alpha - (e_{n-1} + \alpha)}{f(x_n) - f(x_{n-1})}\right)$$

Simplifying and rearranging gives us:

$$e_{n+1} = \left(\frac{e_n - e_{n-1}}{f(x_n) - f(x_{n-1})}\right)\left[\left(\frac{f(x_n)}{e_n} - \frac{f(x_{n-1})}{e_{n-1}}\right)\frac{e_n e_{n-1}}{e_n - e_{n-1}}\right].$$

Let $U_n = \frac{f(x_n) - f(x_{n-1})}{e_n - e_{n-1}}$ and $V_n = \left(\frac{f(x_n)}{e_n} - \frac{f(x_{n-1})}{e_{n-1}}\right)\frac{1}{e_n - e_{n-1}}$ then,

$$e_{n+1} = \frac{V_n}{U_n} e_n e_{n-1} \implies \log(e_{n+1}) = \log(\frac{V_n}{U_n}) + \log(e_n) + \log(e_{n-1}).$$

Now, using Taylor expansion $f(x_n)$ around α:

$$f(x_n) = f(\alpha) + f'(\alpha)e_n + \frac{1}{2}f''(\alpha)e_n^2 + \mathcal{O}(e_n^3)$$

then

$$\begin{aligned} U_n &= f'(\alpha) + \frac{f"(\alpha)}{2}\frac{e_n^2 - e_{n-1}^2}{e_n - e_{n-1}} + \frac{\mathcal{O}(e_n^3 - e_{n-1}^3)}{e_n - e_{n-1}} \\ &= f'(\alpha) + \frac{f"(\alpha)}{2}(e_n + e_{n-1}) + \mathcal{O}(e_n^2 + e_n e_{n-1} + e_{n-1}^2) \to f'(\alpha) \end{aligned}$$

as $n \to \infty$. Since $f(\alpha) = 0$, we obtain,

$$\frac{f(x_n)}{e_n} = f'(\alpha) + \frac{1}{2}f''(\alpha)e_n + \mathcal{O}(e_n^2) \implies V_n = \left(\frac{1}{2}f''(\alpha)(e_n - e_{n-1}) + \mathcal{O}(e_n^2) - \mathcal{O}(e_{n-1}^2)\right)\frac{1}{e_n - e_{n-1}}$$

This gives us,

$$V_n = \frac{1}{2}f''(\alpha) + (\mathcal{O}(e_n + e_{n-1}) \to \frac{1}{2}f''(\alpha) \text{ as } n \to \infty$$

because $e_n \to 0$. Thus, as $n \to \infty$, $\log(\frac{V_n}{U_n}) \to \log\left(\frac{f"(\alpha)}{2f'(\alpha)}\right)$. For ease of notation, let the nonzero constant C be defined as

$$C = \frac{f''(\alpha)}{2f'(\alpha)}.$$

We can now solve for the order of convergence for the procedure. Take $g_n = \log(C) + \log(e_n) = \log(Ce_n)$ then from above, $g_{n+1} = g_n + g_{n-1}$ as $n \to \infty$. Thus, the sequence g_n is Fibonacci, and

$$\lim_{n\to\infty}\frac{g_{n+1}}{g_n} = \phi = \frac{1+\sqrt{5}}{2}.$$

So by transforming g_n back in e_n and dividing by ϕ, we obtain,

$$\lim_{n\to\infty}\frac{\log(Ce_{n+1})}{\log(Ce_n)} = \phi \implies \lim_{n\to\infty}\frac{\log(Ce_{n+1})}{\log(C^\phi e_n^\phi)} = 1.$$

This implies

$$\lim_{n\to\infty}\frac{Ce_{n+1}}{C^\phi e_n^\phi} = 1 \implies \lim_{n\to\infty}\frac{e_{n+1}}{e_n^\phi} = C^{\phi-1}.$$

Thus we conclude that the secant method converges with an order ϕ. □

4.3. **Advantages and Disadvantages.** The secant method is computationally more efficient than Newton's method since it only requires function evaluations. It is faster in convergence than the Bisection method, but slower than Newton's method. However, the secant method can fail if the two initial guesses are not close to the actual root or if $f(x_n)$ equals $f(x_{n-1})$ at any step. Compared to Newton's method, this failure condition is relatively easy to deal with in polynomials. The secant method does not encounter the problem of cycling or divergence as Newton's method does, and if we run the algorithm and experience the divide by zero error when $f(x_n)$ equals $f(x_{n-1})$, we can simply try again by arbitrarily shifting $f(x_n)$ by some small value. Thus, this method is best reserved for functions that are computationally expensive to differentiate, or for functions where Newton's method may be more prone to experiencing its unique problems of cycling and divergence.

5. Bisection method

The bisection method is the slowest method of the three, but converges in cases where the other methods fail to do so. Similarly to the secant method, the bisection method does not require calculation of derivatives. However, the bisection method differs from the other two methods in that rather than iteratively updating an estimate to the root, it instead maintains bounds within which the root must lie, and then narrows those bounds at each step. In a similar manner to binary search on a sorted array, in each iteration, we can determine which half of the range the root is in, and cut the range in half.

In introductory calculus, a fundamental result is the intermediate value theorem, which states that for a continuous function $f(x)$ on a range $[l, r]$, then $f(x)$ takes on all values between $f(l)$ and $f(r)$ in this interval. As a corollary of this theorem, if $f(l)$ and $f(r)$ have opposite signs, then at some point in the range $[l, r]$, we must have $f(x) = 0$, meaning there must be a root in the range $[l, r]$. This motivates the bisection root-finding method, which works as follows: If $f(l_n)$ and $f(r_n)$ have opposite signs, then for the midpoint of the range x_n, then exactly one of the following statements must be true:

- $f(l_n)$ and $f(x_n)$ have opposite signs, in which case the root lies within the range $[l_n, x_n]$. Then, for the next iteration we can use bounds $[l_{n+1}, r_{n+1}] = [l_n, x_n]$
- $f(x_n)$ and $f(r_n)$ have opposite signs, in which case the root lies within the range $[x_n, r_n]$. Then, for the next iteration we can use bounds $[l_{n+1}, r_{n+1}] = [x_n, r_n]$

By repeatedly applying this procedure, we reduce the length of the range where we are searching for the root, and therefore eventually converge closer and closer to the root. To know when to terminate, we define an accuracy parameter ε; if $|f(c)| < \varepsilon$, we are sufficiently close to the root and can return c as the estimate of the root.
Here is the formal definition of the (naive) bisection method:

Algorithm 3: bisection

Input: A continuous function f, two bounds l_0 and r_0 such that $l_0 < r_0$ and $f(l_0)$ and $f(r_0)$ have opposite signs, and an accuracy parameter ε
Output: An estimate of a root α such that $|f(\alpha)| < \varepsilon$
Let $x_n = \frac{l_n + r_n}{2}$ **if** $|f(x_n)| < \varepsilon$ **then**
  return x_n
else
  if $f(l_n)$ *and* $f(x_n)$ *have opposite signs* **then**
    $[l_{n+1}, r_{n+1}] = [l_n, x_n]$
  else
    $[l_{n+1}, r_{n+1}] = [x_n, r_n]$
  return *bisection*$(f, l_{n+1}, r_{n+1}, \varepsilon)$

Every iteration, our search space is cut in half, so the number of iterations required to reach the desired accuracy is $O(-\log \varepsilon)$.

5.1. **Convergence.**

Theorem 3. *If the bisection method converges for a function $f(x)$, it does so with order 1.*

Proof. Starting from initial bounds l_0 and r_0, let l_n and r_n be the bounds of the bisection method after n iterations, and let $x_n = \frac{l_n + r_n}{2}$ be the estimate after n iterations. The bisection method cuts the interval in half each time, so after n iterations, we have $r_n - l_n = \frac{r_0 - l_0}{2^n}$.
We have

$$\lim_{n \to \infty} \frac{|x_{n+1} - \alpha|}{|x_n - \alpha|} = \frac{r_{n+1} - l_{n+1}}{r_n - l_n} = \frac{\frac{r_0 - l_0}{2^{n+1}}}{\frac{r_0 - l_0}{2^n}} = \frac{1}{2}.$$

Therefore, the bisection method has order of convergence 1.

□

5.2. **Starting Bounds.** Now that we've defined the bisection method, the next logical question to ask is what initial bounds $[l_0, r_0]$ we should use. For odd-degree polynomials, it suffices to take $[-M, M]$ as given in lemma 1. For even-degree polynomials, we can find the global extrema and use that as a bound. This is because if $f(x)$ has even degree and a global extrema at x_e, then either $f(x_e)$ and $\lim_{x \to \infty} f(x)$ have opposite signs, or $f(x)$ must have no real roots. To find this global extrema, we find all roots of $f'(x)$ (which are local extrema of $f(x)$) with further iterations of the bisection method. Then, we compare whether $f(e)$ and $f(M)$ have different signs.

Algorithm 4: Finding starting bounds for bisection method

Input: A polynomial function $f(x)$ with degree d
Output: Suitable initial bounds $[l_0, r_0]$ to run bisection method on $f(x)$
if $f(x$ *has odd degree* **then**
  return $[-M, M]$
if $f(x)$ *has even degree* **then**
  $k = 0$
  while *the bisection subprocedure continues to find new roots* **do**
    Let $x_1, \ldots, x_k$ be the roots of $f'(x)$ found so far.
    Run bisection method on $[-M, x_1 - \varepsilon]$, $[x_1 + \varepsilon, x_2 - \varepsilon], \cdots, [x_k + \varepsilon, M]$
    $k \leftarrow k + 1$
  Let $x_e = \begin{cases} \arg\max_{1 \le i \le d-1} f(x_i) & f(x) \text{ has negative leading coefficient} \\ \arg\min_{1 \le i \le d-1} f(x_i) & f(x) \text{ has positive leading coefficient} \end{cases}$
  if x_e *and* M *have the same sign* **then**
    return No roots found
  return $[x_e, \infty]$

5.3. **Advantages and Disadvantages.** The bisection method as given here is the most robust of the three methods. In nearly all cases, it will find a real root if one exists; it only fails if all roots are of even multiplicity. For instance, $f(x) = x^4$ has $f(-1) = 1$ and $f(1) = 1$ but the root $x = 0$ with multiplicity 4 is in the range $[-1, 1]$. The bisection method is also computationally simple, requiring only evaluation of the function, and does not require taking derivatives. However, it exhibits slower convergence than Newton's and secant methods, generally requiring more steps to achieve the same level of precision.

6. Polynomial Factorization Procedure

An ideal polynomial root finding method converges quickly, works for all polynomials and starting guesses/bounds, and is computationally inexpensive. Newton's method, the secant method, and the bisection method only find one root which depends on various properties of function convergence; it is difficult to know a priori which root each of these methods will return.
Therefore, we propose a new algorithm that finds all the roots of a polynomial. This algorithm, which we call the Polynomial Factorization Procedure, is completely robust, finding all roots with multiplicity for any polynomial. The algorithm combines parts of the bisection method with parts of Newton's method, primarily utilizing the fact that all multiple roots of a polynomial lie at a critical point $f'(x) = 0$, and all single roots lie either between two critical points or between a critical point and infinity.

Algorithm 5: Polynomial Factorization Procedure

Input: Polynomial $f(x)$
Output: Roots of $f(x)$
if $f(x)$ *has degree 1 or 2* **then**
  return the solution(s)
else
  Recursively call PFP on $f'(x)$ to get the roots of $f'(x)$ as $c_1, \ldots, c_k$ with multiplicity m_i for c_i.
  Recursively call PFP on $f"(x)$ to get the roots of $f"(x)$ as $d_1, \ldots, d_k$.
  for *each* c_i, $1 \le i \le k$ **do**
    if $f(c_i) = 0$ **then**
      Add c_i to the list of roots with multiplicity $m_i + 1$.
  Now, we can partition the search space into intervals $(-M, c_1), (c_1, c_2), \ldots, (c_k, M)$
  for *each of these intervals* **do**
    if f *has opposite signs at the endpoints of the interval* **then**
      Run the Newton Method from Algorithm (1) with the initial guess d_u such that it is between this interval. Add the result to the list of the roots of multiplicity 1.
  return *roots*

Example. We demonstrate the Polynomial Factorization Procedure on the function $f(x) = x^3 + x^2$.

Figure 8. Segmenting regions by critical points

First, we compute the derivative $f'(x) = 3x^2 + 2x$. Since this function is of degree 2, we can trivially find the roots, which are $-\frac{2}{3}$ and 0. These roots are the critical points of $f(x)$. We now check if they are also roots of $f(x)$; we find that $-\frac{2}{3}$ is not a root of $f(x)$, but 0 is. Taking derivatives, we have $f'(0) = 0$, but $f''(0) \neq 0$, so the root 0 of $f(x)$ has multiplicity 2. Now we examine the regions segmented by the critical points: $[-\infty, -\frac{2}{3} - \varepsilon], [-\frac{2}{3} + \varepsilon, 0 - \varepsilon], [0 + \varepsilon, \infty]$. The first region is the only one where the function has opposite signs at the endpoints, so we run Newton's method within this interval and find a single root at -1. Thus, the Polynomial Factorization Procedure successfully finds the roots of $-\frac{2}{3}$ with multiplicity 1 and 0 with multiplicity 2.

Theorem 4. *The polynomial factorization procedure approximates all roots with high precision.*

Proof. First note that by proposition (1), there exists a bound for all roots that can be computed. We know that every multiple root of $f(x)$ lies at a critical point by lemma (1). Therefore, by checking all critical points, we are only left with roots with multiplicity 1. Now, we are dividing the whole x-axis region by the critical points. From corollary (2), we note that there can be at most one root between these critical points if they have different signs of p. And then by lemma (2), Newton's method converges in this interval for a point of inflection as initial guess. □

7. Acknowledgements

This paper is the collaborative result of the efforts of Kartikesh Mishra, Bryce Hancock, and Darren Yao.

Kartikesh worked on the abstract and the analysis of the Polynomials and their roots in section (2) and Newton Method presented in section (3). Bryce worked on the analysis of the Secant method in the section (4) where he collaborated with Kartikesh on the proof of convergence of the Secant method. Bryce wrote the parts of the introduction that included tables, charts, and examples for the three root-finding methods, as well as some of the motivation for our new method. Darren wrote parts of the introduction including the motivation, table of results, and definitions. Darren also analyzed the Bisection Method in section (5). Darren and Kartikesh devised and wrote up the algorithm and example for the Polynomial Factorization Procedure in (6), and Kartikesh proved its correctness.

We also want to thank our mentors, Dominique Maldague and Thomas Pickering, for their incredible support and guidance. They've helped us immensely with both the technical and non-technical parts of our research.