Study of a Generalized Newton Method for Solution of Nonlinear Equations

Alycia Winter

Supervised by Yalcin Kaya

University of South Australia

Vacation Research Scholarships are funded jointly by the Department of Education and Training and the Australian Mathematical Sciences Institute.
Contents

1 Introduction 3

2 The Classical Newton Method 3

3 A Generalised Newton Method 4

4 Example Equations 8
   4.1 Polynomial equations . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8
      4.1.1 A cubic equation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8
      4.1.2 A quartic equation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
      4.1.3 A quintic equation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15
   4.2 An equation involving sine . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17
   4.3 An equation involving natural logarithm . . . . . . . . . . . . . . . . . . . . . 19
   4.4 An equation involving $x \ln(x)$ . . . . . . . . . . . . . . . . . . . . . . . . 21
   4.5 An equation involving exponential . . . . . . . . . . . . . . . . . . . . . . . . 22

5 The Lambert W Function 24

6 Further Examples With Higher Precision 31

7 Multivariate Generalised Newton Method 32

8 Discussion and Conclusion 37
1 Introduction

The Newton method is an extensively studied method for solving nonlinear equations. This is due to the fact that nonlinear equations appear in numerous disciplines and this technique has clear advantages over other methods. One such advantage is that, under mild conditions, the sequence generated by the Newton iteration is well defined and it converges to a solution quadratically, provided the initial approximation is chosen close enough to the solution. Many variations of the Newton method have been developed in the literature to improve the convergence properties further. One such variation is a Generalised Newton method due to Burachik, Kaya and Sabach [1] which is also shown to be quadratically convergent.

The Generalised Newton method was earlier shown in many cases to have a smaller asymptotic error constant than the Newton method and to converge to the solution in fewer iterations in a wider domain. These results depend on an auxiliary function $s(x)$ whose inverse and first derivative exist and derivative is bounded away from zero. In other words, the Generalised Newton method works better thanks to a “clever” choice of $s(x)$.

The focus of this research is how to best choose $s(x)$ for a given nonlinear equation, such that the Generalised Newton method is more effective than the Newton method in finding a solution. We desire $s(x)$ to lead to both faster local and global convergence. Here faster local convergence is defined as taking fewer iterations (e.g. a smaller asymptotic error constant) and being able to perform a single iteration relatively cheaply. Fast global convergence is defined as having a large interval in which the method converges to the solution in addition to taking fewer iterations and being able to perform a single iteration relatively cheaply.

This report is organised as follows. In Sect. 2, the classical Newton method is described, and standard convergence results for fixed-point methods are recalled. In Sect. 3, the Generalised Newton method is formulated and its properties described. In Sect. 4, we look at functions involving polynomials, trigonometric terms, natural logarithms and exponentials, and derive conditions under which the Generalised Newton method is more effective than the Newton method. In Sect. 5, we look at the special case of the Lambert W function because of its importance in mathematical modeling and due to the fact we can write down “sharp” results about local and global convergence. In Sect. 6, we look at an assortment of functions which contain a mixture of polynomial, trigonometric and exponential expressions where we obtain solutions with higher accuracies and make comparisons. In Sect. 7, we extend the Generalised Newton method for solving multivariate system of nonlinear equations. Finally, in Sect. 8, we discuss possible further questions and extensions of this research.

2 The Classical Newton Method

In this paper we desire to find a solution of a nonlinear equation:

$$f(x) = 0,$$  \hspace{1cm} (1)

where $f : \mathbb{R} \to \mathbb{R}$ is at least once continuously differentiable. One possible method to find a solution to this equation is the Newton method, an iteration of which is given by

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

When given an initial iterate, or approximation of the solution $x_0$, the Newton method generates a sequence $\{x_n\}_{n \in \mathbb{N}}$. Suppose $f(x^*) = 0$. It is well-known that if $x_0$ is “close enough” to $x^*$ and
\( f'(x^*) \neq 0 \), then \( \{x_n\}_{n \in \mathbb{N}} \) converges to \( x^* \) quadratically. The Newton method is the most widely used method for solving nonlinear equations due to this convergence rate and relatively small number of function evaluations. This method was devised by Newton in the 17\(^{th}\) century [2].

The Newton method is a type of fixed point method:

\[
g(x) := x - \frac{f(x)}{f'(x)},
\]

such that if \( g(x^*) = x^* \), then \( x^* \) solves \( f(x) = 0 \).

Consider an iterative method, which generates the sequence \( \{x_n\}_{n \in \mathbb{N}} \) converging to \( x^* \) with \( x_n \neq x^* \) for all \( n \in \mathbb{N} \). If

\[
\lim_{n \to \infty} \frac{|x_{n+1} - x^*|}{|x_n - x^*|^\alpha} = \lambda,
\]

then \( \{x_n\}_{n \in \mathbb{N}} \) is said to converge to \( x^* \) of order \( \alpha \) with asymptotic error constant \( \lambda \).

Two special cases are of interest in this study:

(i) If \( \alpha = 1 \) and \( \lambda < 1 \) then \( \{x_n\}_{n \in \mathbb{N}} \) is said to be linearly convergent;

(ii) If \( \alpha = 2 \) then \( \{x_n\}_{n \in \mathbb{N}} \) is said to be quadratically convergent.

Theorems 1 and 2 below state the requirements under which fixed point methods converge with linear and quadratic rates, respectively.

**Theorem 1. [Linear Rate of Convergence [3]]** Let \( g \in C^1[a,b] \) be such that \( g(x) \in [a,b] \) for any \( x \in [a,b] \), and \( |g'(x)| < 1 \), \( \forall x \in (a,b) \). If \( g'(x^*) \neq 0 \) then for any \( x_0 \neq x^* \) in \( [a,b] \), \( \{x_n\}_{n \in \mathbb{N}} \) converges only linearly to the unique fixed point \( x^* \) in \( [a,b] \).

**Theorem 2. [Quadratic Rate of Convergence [3]]** Let \( g \in C^2[a,b] \) be such that \( g'(x^*) = 0 \) and \( |g''(x)| \leq M \) on an open interval \( I \subset [a,b] \) containing \( x^* \). Then there exists \( \delta > 0 \) such that, for \( x_0 \in [x^* - \delta, x^* + \delta] \), \( \{x_n\}_{n \in \mathbb{N}} \) converges at least quadratically to \( x^* \). Moreover, for sufficiently large values of \( n \),

\[
|x_{n+1} - x^*| < \frac{M}{2} |x_n - x^*|^2.
\]

### 3 A Generalised Newton Method

This variation of the Newton method was originally motivated through Proximal Regularization in a paper written by Burachik, Kaya and Sabach [1]. As this requires a relatively higher level of optimisation techniques we take a more practical approach to devising the same method. Consider the nonlinear equation in (1).

Let \( \alpha \) be a constant, and proceed with the following manipulations.

\[
\begin{align*}
\alpha f(x) &= 0 \\
s(x) - s(x) + \alpha f(x) &= 0 \\
s(x) &= s(x) - \alpha f(x) \\
x &= s^{-1}(s(x) - \alpha f(x)) =: \tilde{g}(x),
\end{align*}
\]

(2)

Next, apply the function \( s \) on both sides of (2) and differentiate to get:

\[
\begin{align*}
s(\tilde{g}(x)) &= s(x) - \alpha f(x) \\
s'(\tilde{g}(x))\tilde{g}'(x) &= s'(x) - \alpha f'(x)
\end{align*}
\]

(3)
Let $\tilde{g}(x^*) = x^*$ such that $\tilde{g}(x)$ is a fixed point method. A necessary condition for quadratic convergence of $x_{n+1} = \tilde{g}(x_n)$ is that $\tilde{g}'(x^*) = 0$. With $\tilde{g}'(x^*) = 0$ and finite $s'(x^*)$, (3) becomes

$$0 = s'(x^*) - \alpha f'(x^*).$$

Rearranging gives

$$\alpha = \frac{s'(x^*)}{f'(x^*)}.$$ 

Let $\alpha$ now be treated as function of $x$ such that,

$$\alpha(x) = \frac{s'(x)}{f'(x)}.$$ 

We take this function $\alpha(x)$ as a candidate, and substitute it back into (2):

$$g(x) = s^{-1}\left(s(x) - s'(x)\frac{f(x)}{f'(x)}\right).$$

To prove quadratic convergence of this method some requirements must be imposed as shown in Lemma 3. Firstly we look at the necessary condition of quadratic convergence.

**Lemma 3.** Suppose that $f, s \in C^2[a, b]$, $f(x^*) = 0$, $s'(x^*) \neq 0 \neq f'(x^*)$. Then $g'(x^*) = 0$.

**Proof.**

$$s(g(x)) = s(x) - s'(x)\frac{f(x)}{f'(x)}$$

$$s'(g(x))g'(x) = s'(x) - \left[s''(x)f(x) + s'(x)f'(x)\right]s'(x) - s'(x)f(x)f''(x)$$

$$s'(g(x^*))g'(x^*) = s'(x^*) - \left[s'(x^*)f'(x^*)\right]s'(x^*)$$

$$s'(g(x^*))g'(x^*) = s'(x^*) - s'(x^*)$$

As $s'(g(x^*)) = s'(x^*) \neq 0$, therefore $g'(x^*) = 0$. 

In the following theorem, we state the sufficient conditions under which the Generalised Newton iteration, $x_{n+1} = g(x_n)$, is quadratically convergent.

**Theorem 4.** Let $f, s \in C^3[a, b]$ and let $s^{-1}$ exist over $x \in [a, b]$. Suppose that $f(x^*) = 0$ for some $x^* \in (a, b)$. Also suppose that $|f'(x^*)| \geq \rho_f$ and $|s'(x^*)| \geq \rho_s$, and that $|f''(x^*)| \leq M_f$ and $|s''(x^*)| \leq M_s$, for positive constants $\rho_f, \rho_s, M_f$ and $M_s$, on an open interval $I$ containing $x^*$. Then there exists $\delta > 0$ such that

$$x_{n+1} = g(x_n)$$

generates a sequence $\{x_n\}_{n \in N}$ converging at least quadratically to $x^*$ for any initial approximation $x_0 \in (x^* - \delta, x^* + \delta)$. Moreover, there exists a positive constant $M$ such that $|g''(x)| < M$ on $I$ and, for sufficiently large values of $n$, the following inequality holds:

$$|x_{n+1} - x^*| < \frac{M}{2}|x_n - x^*|^2.$$
Proof. From Lemma 3, we have that \( g'(x^*) = 0 \). Now it suffices to prove that \(|g''(x)| < M\). It is assumed that \(|s''(x)| \leq M_s\) and \(|f''(x)| \leq M_f\) on the open interval \( I \). Further note that

\[
    s(g(x)) = s(x) - s'(x) \frac{f(x)}{f'(x)}
\]

\[
    s'(g(x))g'(x) = s'(x) - \frac{s''(x)f(x) + s'(x)f'(x) - s'(x)f(x)f''(x)}{f'(x)^2}
\]

\[
    s''(g(x))[g'(x)]^2 + s'(g(x))g''(x) = s''(x) - \frac{2f(x)[f''(x)]^2s'(x) + 2[f'(x)]^3s''(x)}{[f'(x)]^3} + \frac{f''(x)s'(x)(f''(x)s'(x) + 2f''(x)s''(x))}{[f'(x)]^3}
\]

Now evaluate the above equation at the solution \( x^* \), use \( f(x^*) = 0 \), and carry out manipulations to get

\[
    s'(x^*)g''(x^*) = -s''(x^*) + \frac{f''(x^*)s'(x^*)}{f'(x^*)}
\]

\[
    g''(x^*) = -\frac{s''(x^*)}{s'(x^*)} + \frac{f''(x^*)}{f'(x^*)},
\]

where we have also used the fact that \( s'(x^*) \neq 0 \). Taking the absolute value of each side,

\[
    |g''(x^*)| = \left| \frac{f''(x^*)}{f'(x^*)} - \frac{s''(x^*)}{s'(x^*)} \right|.
\]

Using the triangle inequality,

\[
    |g''(x^*)| \leq \left| \frac{f''(x^*)}{f'(x^*)} \right| + \left| \frac{s''(x^*)}{s'(x^*)} \right|
\]

\[
    |g''(x^*)| \leq \frac{M_f}{\rho_f} + \frac{M_s}{\rho_s}.
\]

So by the continuity of \( g''(x) \) in the neighbourhood \( I \) of \( x^* \), there exists

\[
    M \geq \frac{M_f}{\rho_f} + \frac{M_s}{\rho_s}
\]

such that \(|g''(x)| < M\) for all \( x \in I \).

In the original paper [1], the hypotheses used for the same result were weaker than those posed in Theorem 4. By posing stronger assumptions, e.g. thrice continuous differentiability, we have made a compromise in proving the quadratic convergence. However our proof, in return, uses techniques which are more accessible to general undergraduate mathematics students. Theorem 6 below is an original theorem given in [1]. It relies on Lemma 5 which was also posed in the original paper. We include both results (Lemma 5 and Theorem 6 below) from [1] here for the sake of completeness.
Lemma 5. Let \( f, s \in C^1[a,b] \). Assume that for some \( \theta > 0 \), we have \( 0 < \theta < |s'(x)| \) for every \( x \in (a,b) \). If \( f', s' \) are Lipschitz in \((a,b)\), then \( f \circ s^{-1} \)' is well defined and Lipschitz in \( I := s([a,b]) \). Moreover, if \( L_f \) and \( L_s \) are the Lipschitz constants of \( f' \) and \( s' \), respectively, then the Lipschitz constant for \( (f \circ s^{-1})' \) over \( I \) is
\[
M_{fs} := \frac{M_f L_s + M_s L_f}{\theta^3},
\]
where \( M_f \geq \max \{|f'(x)| : x \in [a,b]\} \) and \( M_s \geq \max \{|s'(x)| : x \in [a,b]\} \).

Theorem 6. Let \( f, s \in C^1[a,b] \) satisfy the assumptions of Lemma 5 and assume that there exists \( x^* \in (a,b) \) such that \( f(x^*) = 0 \). Assume that for some \( \rho > 0 \), \( |f'(x)| \geq \rho \) for every \( x \in (a,b) \). Then there exists \( \eta > 0 \) such that \( x_0 \in (x^* - \eta, x^* + \eta) \) then the sequence \( \{x_n\} \) defined recursively as
\[
x_{n+1} = g(x_n), \quad k = 0, 1, 2, \ldots,
\]
where \( g \) is as given in 3, is well defined. In this situation, the sequence \( \{x_n\} \) converges quadratically to \( x^* \); in particular,
\[
|x_{n+1} - x^*| < \frac{M_{fs}(M_s)^3}{2\rho \theta} |x_n - x^*|^2.
\]
where \( \theta, M_{fs} \) and \( M_s \) are as in Lemma 5.

The asymptotic error constant for the Newton formula is given by
\[
\lambda_N = \frac{1}{2} \left| \frac{f''(x^*)}{f'(x^*)} \right|.
\]

The asymptotic error constant expression for the Generalised Newton method is provided in Proposition 1, which was given in [1], but we provide a proof here because the proof can be written in a simple way.

Proposition 1. Let \( f, s \in C^3[a,b] \). Then
\[
\lambda_{gN} = \frac{1}{2} \left| \frac{f''(x^*)}{f'(x^*)} - \frac{s''(x^*)}{s'(x^*)} \right|.
\]

Proof. Choose \( k \) in \((0,1)\) and \( \delta > 0 \) such that on the interval \((x^* - \delta, x^* + \delta)\), we have \( |g'(x)| \leq k \) and \( g'' \) continuous. Since \( |g'(x)| \leq k < 1 \). The terms of the sequence \( \{x_n\}_{n=0}^\infty \) are contained in \((x^* - \delta, x^* + \delta)\). Expanding \( g(x) \) in a linear Taylor polynomial for \( x \in (x^* - \delta, x^* + \delta) \) gives
\[
g(x) = g(x^*) + g'(x^*)(x - x^*) + \frac{g''(\xi)}{2}(x - x^*)^2.
\]
where \( \xi \) lies between \( x \) and \( x^* \). The hypotheses \( g(x^*) = x^* \) and \( g'(x^*) = 0 \) imply that
\[
g(x) = x^* + \frac{g''(\xi)}{2}(x - x^*)^2.
\]
In particular, when \( x = x_n \),
\[
x_{n+1} = g(x_n) = x^* + \frac{g''(\xi)}{2}(x - x^*)^2,
\]
with $\xi_n$ between $x_n$ and $x^*$. Thus,

$$x_{n+1} - x^* = \frac{g''(\xi)}{2} (x - x^*)^2.$$  

Since $|g'(x)| \leq k < 1$ on $(x^* - \delta, x^* + \delta)$ and $g$ maps $[x^* - \delta, x^* + \delta]$ into itself, it follows from the Fixed-Point Theorem that $\{x_n\}_{n=0}^\infty$ converges to $x^*$. But $\xi_n$ is between $x^*$ and $x_n$ for each $n$, so $\{\xi_n\}_{n=0}^\infty$ also converges to $x^*$, and

$$\lim_{n \to \infty} \frac{|x_{n+1} - x^*|}{|x_n - x^*|^2} = \frac{|g''(x^*)|}{2}.$$  

Hence,

$$\lambda_{gN} = \frac{1}{2} |g''(x)|.$$  

From (4), this can be written as

$$\lambda_{gN} = \frac{1}{2} \left| \frac{f''(x^*)}{f'(x^*)} - \frac{s''(x^*)}{s'(x^*)} \right|.$$  

A result from the proof of Theorem 4 is that an upper bound of the asymptotic error constant of the Generalised Newton formula is given by

$$\lambda_{gN} \leq \frac{M_f}{\rho_f} + \frac{M_s}{\rho_s}.$$  

This upper bound will always be greater than the one imposed on the Newton method, which is given by

$$\lambda_N \leq \frac{M_f}{\rho_f}.$$  

Table 3, which is also drawn from [1], illustrates how wide variety of generalised Newton methods can be obtained by choosing $s(x)$ differently.

4 Example Equations

In this section we consider different types of nonlinear equations and investigate the types of $s(x)$ we can use in the generalised Newton method.

4.1 Polynomial equations

4.1.1 A cubic equation

Consider the real cubic polynomial:

$$x^3 + ax^2 + bx + c = 0$$  

where $a, b$ and $c$ are real constants. Any polynomial can be rewritten in a depressed form, and this would make analysis simpler. In this case, to formulate the depressed cubic where the $x^2$–term is suppressed, let $x$ be defined as

$$x := y - \frac{a}{3}.$$  

8
Substituting it back into the polynomial:

\[ y^3 + \left( b - \frac{a^2}{3} \right) y + \left( c + \frac{2a^3}{27} - \frac{ab}{3} \right) = 0 \]

Hence we can consider the depressed cubic and all general cubic polynomials will be covered:

\[ y^3 + Ay + B = 0, \]

where \( A \) and \( B \) are real constants. So we will consider the depressed form

\[ x^3 + Ax + B = 0 \quad (5) \]

to represent all cubic equations. Using Cardano’s formula the roots of this cubic can be given as:

\[
x = \sqrt[3]{-\frac{B}{2} + \sqrt{\frac{B^2}{4} + \frac{A^3}{27}}} - \sqrt[3]{-\frac{B}{2} + \sqrt{\frac{B^2}{4} + \frac{A^3}{27}}} \quad (6)
\]

Now, \( f'(x) = 3x^2 + A \) and \( f''(x) = 6x \). Take \( s(x) = x^3 \). The asymptotic error constants at \( x = x^* \) are:

\[
\lambda_N = \frac{1}{2} \left| \frac{6x}{3x^2 + A} \right|, \quad \lambda_{gN} = \frac{1}{2} \left| \frac{6x}{3x^2 + A} - \frac{2}{x} \right| = \left| \frac{A}{x(3x^2 + A)} \right|
\]

Next, compute the condition when \( \lambda_{gN} < \lambda_N \) holds:

\[
\lambda_{gN} < \lambda_N
\]

\[
\left| \frac{A}{x(3x^2 + A)} \right| < \left| \frac{3x}{3x^2 + A} \right|
\]

\[
\left| \frac{A}{x} \right| \frac{1}{3x^2 + A} < 3|x| \left| \frac{1}{3x^2 + A} \right|
\]

\[
|x|^2 > \frac{|A|}{3} \quad (7)
\]

<table>
<thead>
<tr>
<th>( s(x) )</th>
<th>( s^{-1}(x) )</th>
<th>( s'(x) )</th>
<th>( g(x) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x^{p-1} )</td>
<td>( x^{1/(p-1)} )</td>
<td>( (p-1)x^{p-2} )</td>
<td>( x^{(p-2)/(p-1)} \left(x - (p-1) \frac{f(x)}{f'(x)}\right)^{1/(p-1)} )</td>
</tr>
<tr>
<td>( e^x )</td>
<td>( \ln x )</td>
<td>( e^x )</td>
<td>( x + \ln \left( 1 - \frac{f(x)}{f'(x)} \right) )</td>
</tr>
<tr>
<td>( \ln x )</td>
<td>( e^x )</td>
<td>( 1/x )</td>
<td>( xe^{-f(x)/(xf'(x))} )</td>
</tr>
<tr>
<td>( 1/x )</td>
<td>( 1/x )</td>
<td>(-1/x^2 )</td>
<td>( x \left( \frac{1 + f(x)}{f'(x)} \right)^{-1} )</td>
</tr>
<tr>
<td>( \sinh x )</td>
<td>( \arcsinh x )</td>
<td>( \cosh x )</td>
<td>( \arcsinh \left( \sinh x - \cosh x \frac{f(x)}{f'(x)} \right) )</td>
</tr>
<tr>
<td>( \tan x )</td>
<td>( \arctan x )</td>
<td>( \sec^2 x )</td>
<td>( \arctan \left( \tan x - \sec^2 x \frac{f(x)}{f'(x)} \right) )</td>
</tr>
<tr>
<td>( \arctan x )</td>
<td>( \tan x )</td>
<td>( 1/(1 + x^2) )</td>
<td>( \tan \left( \arctan x - \frac{f(x)}{(1 + x^2)f'(x)} \right) )</td>
</tr>
</tbody>
</table>

Table 1: Generalised Newton function \( g(x) \) for several choices of function \( s(x) \) [1].
Here, \( x \) in the above inequality is any of the roots. Using (6), one gets
\[
\left( \sqrt{\frac{B}{2}} + \sqrt{\frac{B^2}{4} + \frac{A^3}{27}} + \sqrt{\frac{B}{2} - \sqrt{\frac{B^2}{4} + \frac{A^3}{27}}} \right)^2 < \frac{|A|}{3}.
\]

Unfortunately, this inequality does not allow for much simplification. So instead we will look at particular cases of Cardano’s formula:

1. \( \frac{B^2}{4} + \frac{A^3}{27} = 0 \) and \( A, B \neq 0 \).
2. \( B = 0 \) and \( A \neq 0 \).
3. \( A = 0 \) and \( B \neq 0 \).
4. \( \frac{B^2}{4} + \frac{A^3}{27} < 0 \) and \( A, B \neq 0 \).
5. \( \frac{B^2}{4} + \frac{A^3}{27} > 0 \) and \( A, B \neq 0 \).

**Case 1:** \( B^2/4 + A^3/27 = 0 \) and \( A, B \neq 0 \). It can be observed from this equation that \( A \) must always be negative. When this occurs the cubic can be simplified to
\[
(x - \frac{3B}{A}) (x + \frac{3B}{2A})^2 = 0
\]
which gives the roots \( x = 3B/A \) and the double root \( x = -3B/(2A) \). Rearranging the given condition to be in terms of \( B \) we get, \( B^2 = -4 \cdot A^3/27 \). First take the single root, and check (7).

\[
\left| \frac{3B}{A} \right| > \frac{|A|}{3}
\]
\[
27 \cdot \left| -4 \cdot \frac{A^3}{27} \right| > |A|^3
\]
\[
|4 \cdot A^3| > |A|^3
\]
\[
4 > 1,
\]
which is always true. Therefore the condition is met and so the Generalised Newton method is more efficient in finding the single root, \( x^* = 3B/A \).

As for the double root \( x = -3B/(2A) \), we note that \( f' \left( \frac{-3B}{2A} \right) = 0 \), so neither the Newton nor the Generalised Newton method is applicable.

Numerical examples with a precision of \( 10^{-10} \) for the single root \( x^* = \frac{3B}{A} \) can be seen in Table 2.

**Case 2:** \( B = 0 \) and \( A \neq 0 \). Therefore, \( x^3 + Ax = 0 \) and it can be simplified to \( x(x^2 + A) = 0 \). Therefore, (i) if \( A > 0 \), then there is the single root \( x = 0 \) and (ii) if \( A < 0 \), then \( x = 0, \sqrt{-A} \) or \( -\sqrt{-A} \).

When \( x = 0 \), \( s'(x) = 0 \), so the Generalised Newton method is not applicable.

With any of \( x = \sqrt{-A} \) and \( x = -\sqrt{-A} \) (7) becomes
\[
\left| \sqrt{-A} \right|^2 > \frac{|A|}{3}
\]
\[
1 > \frac{1}{3}.
\]

<table>
<thead>
<tr>
<th>Function</th>
<th>$x^*$</th>
<th>$\lambda_N$</th>
<th>$\lambda_{gN}$</th>
<th>$x_0$</th>
<th>N Iter.</th>
<th>GN Iter.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x^3 - 3x + 2$</td>
<td>$-2$</td>
<td>$2/3$</td>
<td>$1/6$</td>
<td>$-2.5$</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>$x^3 - 12x - 16$</td>
<td>$4$</td>
<td>$1/3$</td>
<td>$1/12$</td>
<td>$4.5$</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>$x^3 - x/8 - 1/(24\sqrt{6})$</td>
<td>$1/\sqrt{6}$</td>
<td>$3.26599$</td>
<td>$0.81650$</td>
<td>1</td>
<td>8</td>
<td>5</td>
</tr>
</tbody>
</table>

Table 3: Numerical examples comparing the Newton method and the Generalised Newton method with a precision of $10^{-10}$ for cubic polynomial (5) where $B = 0$.

<table>
<thead>
<tr>
<th>Function</th>
<th>$x^*$</th>
<th>$\lambda_N$</th>
<th>$\lambda_{gN}$</th>
<th>$x_0$</th>
<th>N Iter.</th>
<th>GN Iter.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x^3 - 9x$</td>
<td>$-3$</td>
<td>$1$</td>
<td>$1/6$</td>
<td>$-3.5$</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>$x^3 - 9x$</td>
<td>$3$</td>
<td>$1$</td>
<td>$1/6$</td>
<td>$2.5$</td>
<td>6</td>
<td>5</td>
</tr>
</tbody>
</table>

which always holds. This means the Generalised Newton method converges in fewer iterations in finding the roots $x = \sqrt{-A}$ and $x = -\sqrt{-A}$ than the Newton method. Numerical examples with a precision of $10^{-10}$ for these cases can be seen in Table 3.

**Case 3:** $A = 0$ and $B \neq 0$. Therefore, $x^3 + B = 0$. This gives the only real root $x = -\sqrt[3]{B}$, and so (7) becomes

$$| -\sqrt[3]{B}|^2 > 0$$

$$B^{2/3} > 0,$$

which always holds. Hence the Generalised Newton method converges in fewer iterations. Numerical examples with a precision of $10^{-10}$ for these cases can be seen in Table 4. For any cubic in this form, the Generalised Newton method will take at most 1 iteration to find the root because the method would stop with $f(x_1) = 0$, for any $x_0$. This is due to the iterative formula as seen below.

$$x_{n+1} = \left( \frac{x_n^3 - 3x_n^2x_{n+1} + B}{3x_n^2} \right)^{1/3}$$

$$= \left( x_n^3 - x_{n+1}^3 - B \right)^{1/3}$$

$$= (-B)^{1/3}$$

**Case 4:** $B^2/4 + A^3/27 < 0$ and $A, B \neq 0$. In this case there are three real roots. Since $B^2 > 0$, one must have $A < 0$. Not much simplification can be done in this case so performing numerical experiments is the only way to observe some behaviour as seen in Table 5.

<table>
<thead>
<tr>
<th>Function</th>
<th>$x^*$</th>
<th>$\lambda_N$</th>
<th>$\lambda_{gN}$</th>
<th>$x_0$</th>
<th>N Iter.</th>
<th>GN Iter.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x^3 + 27$</td>
<td>$-3$</td>
<td>$1/3$</td>
<td>$0$</td>
<td>$-2.5$</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>$x^3 - 64$</td>
<td>$4$</td>
<td>$1/4$</td>
<td>$0$</td>
<td>$3.5$</td>
<td>5</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4: Numerical examples comparing the Newton method and the Generalised Newton method with a precision of $10^{-10}$ for cubic polynomial (5) where $A = 0$. 

11

<table>
<thead>
<tr>
<th>Function</th>
<th>$x^*$</th>
<th>$\lambda_N$</th>
<th>$\lambda_{gN}$</th>
<th>$x_0$</th>
<th>N Iter.</th>
<th>GN Iter.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x^3 - 7x - 6$</td>
<td>-1</td>
<td>3/4</td>
<td>7/4</td>
<td>-1.2</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>$x^3 - 7x - 6$</td>
<td>-2</td>
<td>6/5</td>
<td>7/10</td>
<td>-1.8</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>$x^3 - 7x - 6$</td>
<td>3</td>
<td>1/4</td>
<td>0</td>
<td>3.5</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Function</th>
<th>$x^*$</th>
<th>$\lambda_N$</th>
<th>$\lambda_{gN}$</th>
<th>$x_0$</th>
<th>N Iter.</th>
<th>GN Iter.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x^3 - 2x - 5$</td>
<td>2.09455</td>
<td>0.56298</td>
<td>0.12832</td>
<td>2.8</td>
<td>6</td>
<td>5</td>
</tr>
</tbody>
</table>

Case 5: $\frac{B^2}{4} + \frac{A^3}{27} > 0$ and $A, B \neq 0$. This also does not allow for simplification. So no definite results can be concluded. A specific example of this case, in Table 6, is the polynomial used by Newton in his original work on the Newton method [4], $f(x) = x^3 - 2x - 5$.

We will now look at the time taken to complete one iteration of each method.

The Newton method:

$$g_N(x) = x - \frac{x^3 + Ax + B}{3x^2 + A}$$

$$= \frac{2x^3 - B}{3x^2 + A}$$

The Generalised Newton method:

$$g_{gN}(x) = \left(\frac{x^3 - 3x^2 x^3 + Ax + B}{3x^2 + A}\right)^{1/3}$$

$$= \left(-\frac{x^2(2Ax + 3B)}{3x^2 + A}\right)^{1/3}$$

To get an idea about the time each method takes to complete one iteration of each method we will set $A = 1$, $B = 1$ and $x = -0.1$. It takes the Newton method $3.4 \cdot 10^{-7}$ seconds to complete one iteration and it takes the Generalised Newton method $3.7 \cdot 10^{-7}$ seconds to complete one iteration. This implies the Generalised Newton method is 1.1 times slower than the Newton method. Furthermore for the Generalised Newton method to be more efficient than the Newton method it must perform less than 90% of the iterations taken by the Newton method. Which it is seen to have done in the great majority of the previous numerical examples.

4.1.2 A quartic equation

In a similar way to how we formulated the depressed cubic equations, the depressed quartic can be formed. So for simple analysis we will consider the quartic equation in the form,

$$f(x) = x^4 + Ax^2 + Bx + C.$$
Hence, $f'(x) = 4x^3 + 2Ax + B$ and $f''(x) = 12x^2 + 2A$. Take $s(x) = x^4$. The asymptotic error constants at $x = x^*$ are:

$$
\lambda_N = \left| \frac{6x^2 + A}{4x^3 + 2Ax + B} \right| \\
\lambda_gN = \frac{1}{2} \left| \frac{12x^2 + 2A}{4x^3 + 2Ax + B} - \frac{3}{x} \right| = \frac{1}{2} \left| \frac{4Ax + 3B}{x(4x^3 + 2Ax + B)} \right|
$$

Next, we compute when $\lambda_gN < \lambda_N$ holds:

$$
\frac{1}{2} \left| \frac{4Ax + 3B}{x(4x^3 + 2Ax + B)} \right| < \left| \frac{6x^2 + A}{4x^3 + 2Ax + B} \right| \\
\left| 4Ax + 3B \right| < \left| 12x^3 + 2Ax \right| \tag{8}
$$

There is no formula for finding the roots of a depressed quartic, so we will look at the following general cases for this polynomial:

1. $B = 0$ and $A, C \neq 0$.
2. $B = 0$, $C = 0$ and $A \neq 0$.
3. $C = 0$ and $A, B \neq 0$.
4. $A = 0$ and $B, C \neq 0$
5. $A = 0$, $B = 0$ and $C \neq 0$.
6. $A = 0$, $C = 0$ and $B \neq 0$.

**Case 1:** $B = 0$ and $A, C \neq 0$. Substituting this into (8),

$$
|4Ax| < |12x^3 + 2Ax| \\
1 < \left| \frac{3x^2}{A} + \frac{1}{2} \right|
$$

Therefore, if a pole $x$ satisfies $x^2 > A/6$ or $x^2 < -A/6$, then the Generalised Newton method will converge in fewer iterations.

**Case 2:** $B = 0$, $C = 0$ and $A \neq 0$. Hence the roots of this equation are

$$
x = 0 \text{ (Double)} \quad x = \sqrt{-A} \quad x = -\sqrt{-A}.
$$

When $x = 0$, $s'(x) = 0$, so the Generalised Newton formula is not applicable. For $x = \sqrt{-A}$ and $x = -\sqrt{-A}$, $A$ must be negative for the root to be real. Using the requirements in (8),

$$
|2A| < |−6A + A| \\
|2A| < |5A| \\
2 < 5.
$$

This expression always holds, so the Generalised Newton method will converge in fewer iterations.

**Case 3:** $C = 0$ and $A, B \neq 0$. Hence $x = 0$ is a root of this equation and the other roots can be found through the Cardano’s formula. If $\frac{B^2}{4} + \frac{A^3}{27} = 0$ then the other two roots of the equation are $x = \frac{3B}{A}$ and the double root $x = -\frac{3B}{2A}$. Once again at $x = 0$ the Generalised
Newton formula is not applicable. At the root \( x = \frac{3B}{A} \), through (8)

\[
\begin{align*}
|2A + \frac{3B}{\frac{3B}{A}}| &< \left| \frac{54B^2}{A^2} + A \right| \\
|2A + \frac{A}{2}| &< \left| \frac{-8A^3}{A^2} + A \right| \\
\left| \frac{5A}{2} \right| &< \left| 7A \right|
\end{align*}
\]

\[5 < 14.\]

The Generalised Newton method converges in fewer iterations. For the double root, \( x = -\frac{3B}{2A} \), through (8)

\[
\begin{align*}
|2A - \frac{3B}{\frac{3B}{A}}| &< \left| \frac{27B^2}{A^2} + A \right| \\
|A| &< \left| \frac{-4A^3}{A^2} + A \right| \\
|A| &< \left| 3A \right|
\end{align*}
\]

\[1 < 3.\]

The Generalised Newton method should converge in fewer iterations.

**Case 4:** \( A = 0 \) and \( B, C \neq 0 \). By looking at the requirements imposed by the inequality (8), when

\[
\left| \frac{3B}{2x} \right| < |6x^2| \leftrightarrow |B| < |4x^3|
\]

then the Generalised Newton method converges in fewer iterations.

**Case 5:** \( A = 0, \ B = 0 \) and \( C \neq 0 \). Then for all roots other than \( x = 0 \), due to (8),

\[|0| < |4x^3|\]

the Generalised Newton method will converge in fewer iterations.

**Case 6:** \( A = 0 \) and \( C = 0 \) then the real roots are \( x = 0 \) and \( x = -\sqrt[3]{B} \) therefore by (8),

\[
\begin{align*}
|B| &< |4(-\sqrt[3]{B})^3| \\
|B| &< | -4B | \\
1 &< 4
\end{align*}
\]

Hence, for all roots other than \( x = 0 \) the Generalised Newton method will converge in fewer iterations.

From numerical testing it is seen that the Generalised Newton method only successfully finds the positive roots of a function. If we look at the iterate when \( s(x) = x^4 \),

\[
x_{n+1} = \left( x_n^4 - 4x_n^3 \frac{x_n^4 + Ax_n^2 + Bx_n + C}{4x_n^4 + 2Ax_n + B} \right)^{1/4}
\]
we can see that $x_{n+1}$ will always be a positive number due to the nature of taking a quartic root. Due to this if it is known that the root is positive and the close range of where the root lies, the Generalised Newton method should be used. Numerical examples for the cases with positive solutions can be seen in Table 7.

Let us now compare the time taken for each method to complete one iteration.

The Newton method:

$$g_N(x) = x - \frac{x^4 + Ax^2 + Bx + C}{4x^3 + 2Ax + B}$$

The Generalised Newton method:

$$g_{gN}(x) = \left(\frac{x^4 - 4x^3x^4 + Ax^2 + Bx + C}{4x^3 + 2Ax + B}\right)^{1/4}$$

To compare the time it takes to complete one iteration of each method we will set $A = 1$, $B = 1$, $C = 1$ and $x = -0.1$. It takes the Newton method $3.5 \cdot 10^{-7}$ seconds to complete one iteration and it takes the Generalised Newton method $4.6 \cdot 10^{-7}$ seconds to complete one iteration. This implies the Generalised Newton method is 1.3 times slower than the Newton method, in this particular case. Furthermore for the Generalised Newton method to be more efficient than the Newton method it must perform less than 76% of the iterations taken by the Newton method. Which it is seen to have done only a few of the previous examples. When global convergence is considered, however, the Generalised Newton method could be a more efficient method as it could have a “better fit” to the function. A repetition of previous numerical examples, with a further initial iterate can be seen in Table 8. From these tests it can be seen in a majority of the cases the Generalised Newton method is more effective than the Newton method and hence has much better global convergence properties.

### 4.1.3 A quintic equation

As any quintic equation can also be expressed in a depressed form we will use this form for our purposes,

$$f(x) = x^5 + Ax^3 + Bx^2 + Cx + D.$$
Next, we compute when $\lambda_{gN} < \lambda_N$ holds:

$$
\frac{3Ax^2 + 3Bx + 2C}{x(5x^4 + 3Ax^2 + 2Bx + C)} < \left| \frac{10x^3 + 3Ax + B}{5x^4 + 3Ax^2 + 2Bx + C} \right| < \frac{3Ax^2 + 3Bx + 2C}{x}
$$

Through the use of the inequality in (9) and the same procedure used in the cubic and quartic polynomial cases it can be identified when the Generalised Newton method converges in fewer iterations than the Newton method. Due to the repetitive nature of this we will instead just consider some numerical examples these cases which are seen in Table 9.

Let us now compare the iteration times of the methods.
Table 10: Numerical examples, with a distant \( x_0 \), comparing the Newton method and the Generalised Newton method with a precision of \( 10^{-30} \) for quintic polynomials.

<table>
<thead>
<tr>
<th>Function</th>
<th>( x^* )</th>
<th>( \lambda_N )</th>
<th>( \lambda_{gN} )</th>
<th>( x_0 )</th>
<th>N Iter.</th>
<th>GN Iter.</th>
<th>Iter. %</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x^3 + 3x^3 + x + 5 )</td>
<td>-1.0</td>
<td>1.26677</td>
<td>0.73333</td>
<td>-10</td>
<td>16</td>
<td>11</td>
<td>69%</td>
</tr>
<tr>
<td>( x^5 - 6x^3 + 4x^2 - 8x )</td>
<td>-2.88853</td>
<td>1.10930</td>
<td>0.41690</td>
<td>-13</td>
<td>15</td>
<td>8</td>
<td>53%</td>
</tr>
<tr>
<td>( x^5 - 6x^3 + 4x^2 - 8x )</td>
<td>2.39282</td>
<td>1.36029</td>
<td>0.52445</td>
<td>-1</td>
<td>15</td>
<td>8</td>
<td>53%</td>
</tr>
<tr>
<td>( x^5 - 5x^2 - 2x - 7 )</td>
<td>1.98236</td>
<td>1.31613</td>
<td>0.30723</td>
<td>12</td>
<td>15</td>
<td>7</td>
<td>47%</td>
</tr>
<tr>
<td>( x^5 - 9x^3 - 7x^2 + 63 )</td>
<td>-3</td>
<td>0.96078</td>
<td>0.29412</td>
<td>-13</td>
<td>14</td>
<td>8</td>
<td>57%</td>
</tr>
<tr>
<td>( x^5 - 9x^3 - 7x^2 + 63 )</td>
<td>1.91293</td>
<td>0.19360</td>
<td>0.123912</td>
<td>0.94</td>
<td>9</td>
<td>8</td>
<td>89%</td>
</tr>
<tr>
<td>( x^5 - 9x^3 - 7x^2 + 63 )</td>
<td>3</td>
<td>1.51667</td>
<td>0.850</td>
<td>13</td>
<td>15</td>
<td>9</td>
<td>60%</td>
</tr>
</tbody>
</table>

The Newton method:

\[
g_N(x) = x - \frac{x^5 + Ax^3 + Bx^2 + Cx + D}{5x^4 + 3Ax^2 + 2Bx + C} = \frac{4x^5 + 2Ax^3 + Bx^2 - D}{5x^4 + 3Ax^2 + 2Bx + C}
\]

The Generalised Newton method:

\[
g_{gN}(x) = \left(\frac{x^5 - 5x^2 + Ax^3 + Bx^2 + Cx + D}{5x^4 + 3Ax^2 + 2Bx + C}\right)^{1/5} = \left(\frac{-x^4(2Ax^3 + 3Bx^2 + 4Cx + 5D)}{5x^4 + 3Ax^2 + 2Bx + C}\right)^{1/5}
\]

To get an idea about the time it takes to complete one iteration of each method we will set \( A = 1, B = 1, C = 1, D = 1 \) and \( x = -0.1 \). It takes the Newton method \( 4.7 \cdot 10^{-7} \) seconds to complete one iteration and it takes the Generalised Newton method \( 6.3 \cdot 10^{-7} \) seconds to complete one iteration. This implies the Generalised Newton method is 1.3 times slower than the Newton method. Furthermore for the Generalised Newton method to be more efficient than the Newton method it must perform less than 74% of the iterations taken by the Newton method. Which it is seen to have been the case in a few of the previous numerical examples. When global convergence is considered the Generalised Newton method could be a more efficient method as it could have a better fit to the function. Numerical examples with further initial iterates can be seen in Table 10. It can be seen that in the majority of the cases tested the Generalised Newton method has better global convergence properties.

### 4.2 An equation involving sine

Consider the function \( f(x) = \sin(x) + Ax + B \). Hence, \( f'(x) = \cos(x) + A \) and \( f''(x) = -\sin(x) \).

Take \( s(x) = \sin(x) \). The asymptotic error constants at \( x = x^* \) are:

\[
\lambda_N = \frac{1}{2} \begin{vmatrix} -\sin(x) \\ \cos(x) + A \end{vmatrix}, \quad \lambda_{gN} = \frac{1}{2} \begin{vmatrix} -\sin(x) \\ \cos(x) + A + \tan(x) \end{vmatrix}
\]

17
Table 11: Numerical examples comparing the Newton method and the Generalised Newton method for \( f(x) = \sin(x) + Ax + B = 0 \) with a precision of \( 10^{-10} \).

Next, we compute when \( \lambda_{gN} < \lambda_N \) holds:

\[
\frac{1}{2} \left| \frac{-\sin(x)}{\cos(x) + A} + \tan(x) \right| < \frac{1}{2} \left| \frac{-\sin(x)}{\cos(x) + A} \right|
\]

\[
\frac{A \sin(x)}{\cos(x)(\cos(x) + A)} < \frac{\sin(x)}{\cos(x) + A}
\]

\[
|A \tan(x)| \left| \frac{1}{\cos(x) + A} \right| < |\sin(x)| \left| \frac{1}{\cos(x) + A} \right|
\]

\[
|\cos(x)| > |A| \tag{10}
\]

Hence due to the inequality (10), if \( |A| \geq 1 \), then \( \lambda_{gN} \geq \lambda_N \) and the Newton method converges just as fast or in fewer iterations than the Generalised Newton method. If instead \( |A| < 1 \), then \( |\cos(x)| > A \) and \( |\cos(x)| > -A \) and the Generalised Newton method converges in fewer iterations. This though does not take into account the operations used in the Generalised Newton method and their restrictions when using \( s(x) = \sin(x) \),

\[
x_{n+1} = \arcsin \left( \sin(x) - \cos(x) \frac{\sin(x) + Ax + B}{\cos(x) + A} \right).
\]

Therefore \( \arcsin(x) \) is defined on \([-1, 1]\), the argument of \( \arcsin \) is restricted to this region otherwise the Generalised Newton iteration is not defined. So,

\[
-1 \leq A(\sin(x) - x \cos(x)) - B \cos(x) \leq 1.
\]

Numerical examples can be seen in Table 11.

To identify which method is less costly to compute we must look at the time taken for each method to calculate one iterate.

The Newton method:

\[
g_N(x) = x - \frac{\sin(x) + Ax + B}{\cos(x) + A}
\]

\[
= \frac{x \cos x - \sin x - B}{\cos x + A}
\]
| Function       | $x^*$        | $|\cos(x)|$ | $\lambda_N$ | $\lambda_{gN}$ | $x_0$ | N Iter. | GN Iter. | Iter. % |
|----------------|-------------|-----------|-------------|---------------|------|---------|----------|--------|
| $\sin(x) + x/2 + 1/2$ | $-0.33758$  | $0.94356$ | $1/2$       | $0.11472$     | $0.06079$ | $5$  | $41$    | $6$    | $15\%$ |
| $\sin(x) - 4x/5 + 1/20$ | $-0.96521$  | $0.56924$ | $1/4$       | $1.78147$     | $2.50363$ | $-2$ | $7$     | $11$   | $157\%$ |
| $\sin(x) - 4x5 + 1/20$ | $-0.26555$  | $0.96495$ | $1/4$       | $0.79552$     | $0.65954$ | $0.4$ | $9$     | $7$    | $78\%$ |
| $\sin(x) - 4x/5 + 1/20$ | $1.24790$   | $0.31731$ | $1/4$       | $0.98234$     | $2.47663$ | $2$  | $15$    | $-1$   |

Table 12: Numerical examples, with distant $x_0$, comparing the Newton method and the Generalised Newton method for $f(x) = \sin(x) + Ax + B = 0$ with a precision of $10^{-10}$.

The Generalised Newton method:

$$g_{gN}(x) = \arcsin \left( \frac{\sin(x) - \cos(x) \sin(x) + Ax + B}{\cos(x) + A} \right)$$

To compare the time it takes to complete one iteration of each method we will set $A = 1$, $B = 1$ and $x = -0.1$. It takes the Newton method $1.6 \cdot 10^{-7}$ seconds to complete one iteration and it takes the Generalised Newton method $3.0 \cdot 10^{-7}$ seconds to complete one iteration. This implies the Generalised Newton method is 1.9 times slower than the Newton method. Furthermore for the Generalised Newton method to be more efficient than the Newton method it must perform less than 53% of the iterations taken by the Newton method. Which never happened in the examples taken though when global convergence is considered the Generalised Newton method could be a more efficient method as it could have a better fit to the function. The same numerical examples with further starting points can be seen in Table 12. Due to the nature of $f(x) = \sin(x) + Ax + B$ there are multiple intervals which find each root can be found. This causes the Generalised Newton method and the Newton method to find different roots at the same initial starting point. It can be seen that global convergence of the Generalised Newton method is not more effective than the Newton method. This could possibly be fixed if a new function was assigned to $s(x)$.

4.3 An equation involving natural logarithm

Consider the function $f(x) = \ln(x) + Ax + B$ where $x > 0$. Hence, $f'(x) = x^{-1} + A$ and $f''(x) = -x^{-2}$. Take $s(x) = \ln(x)$. The asymptotic error constants at $x = x^*$ are:

$$\lambda_N = \frac{1}{2} \left| \frac{-x^{-2}}{x^{-1} + A} \right|$$

$$\lambda_{gN} = \frac{1}{2} \left| \frac{-x^{-2}}{x^{-1} + A + x^{-1}} \right|$$
Table 13: Numerical examples comparing the Newton method and the Generalised Newton method for \( f(x) = \ln(x) + Ax + B = 0 \) with a precision of \( 10^{-10} \).

| Function          | \( x^* \) | \( \frac{1}{|A|} \) | \( \lambda_N \) | \( \lambda_{gN} \) | \( x_0 \) | N Iter. | GN Iter. |
|-------------------|-----------|------------------|---------------|------------------|--------|--------|--------|
| \( \ln(x) + 8x + 1 \) | 0.1300    | 1/8              | 1.88537       | 1.96078          | 0.2    | 5      | 5      |
| \( \ln(x) + 12x - 19 \) | 1.54697   | 1/12             | 0.01652       | 0.30669          | 1.3    | 4      | 5      |
| \( \ln(x) + x/7 - 1 \) | 2.03309   | 7                | 0.19058       | 0.05535          | 2.2    | 4      | 4      |

Next, we compute when \( \lambda_{gN} < \lambda_N \) holds:

\[
\frac{\lambda_{gN}}{\lambda_N} < \frac{1}{2} \left| \frac{-x^{-2} + x^{-1}}{x^{-1} + A} \right| < \frac{1}{2} \left| \frac{x^{-2}}{x^{-1} + A} \right| < \frac{A}{x(x^{-1} + A)} < \frac{1}{x^{-1} + A} < \frac{1}{|A|} \frac{1}{x^{-1} + A} \]

\[
x < \frac{1}{|A|}
\]

Hence when the inequality in (11) hold, the Generalised Newton method will take fewer iterations. This can be confirmed through the numerical examples in Table 13.

The iterative formula of each method is given bellow.

The Newton method:

\[
g_N(x) = x - \frac{\ln(x) + Ax + B}{\frac{x}{x} + A} = x(1 - \ln x - B) \frac{1}{1 + Ax}
\]

The Generalised Newton method:

\[
g_{gN}(x) = \exp \left( \ln x - \frac{1}{x} \ln(x) + Ax + B \right) = \exp \left( -\frac{(\ln x + A - B)}{1 + A} \right)
\]

To compare the time it takes to complete one iteration of each method we will set \( A = 1, \ B = 1 \) and \( x = -0.1 \). It takes the Newton method \( 3.9 \cdot 10^{-7} \) seconds to complete one iteration and it takes the Generalised Newton method \( 4.7 \cdot 10^{-7} \) seconds to complete one iteration. This implies the Generalised Newton method is 1.2 times slower than the Newton method. Furthermore for the Generalised Newton method to be more efficient than the Newton method it must perform less than 82% of the iterations taken by the Newton method. Which never happened in the examples taken though when global convergence is considered the Generalised Newton method could be a more efficient method as it could have a better fit to the function. The same numerical examples repeated with further starting points can be seen in Table 14. In only one of these examples the Generalised Newton method was more efficient than the Newton method. Maybe as more distant initial points are taken this could change.
Consider the function \( f(x) = x \ln(x) + Ax + B \). Hence, \( f'(x) = \ln(x) + 1 + A \) and \( f''(x) = x^{-1} \).

Take \( s(x) = \ln(x) \). The asymptotic error constants at \( x = x^* \) are:

\[
\lambda_N = \frac{1}{2} \left| \frac{x^{-1}}{\ln(x) + 1 + A} \right| \quad \lambda_gN = \frac{1}{2} \left| \frac{x^{-1}}{\ln(x) + 1 + A} + x^{-1} \right|
\]

Next, we compute when \( \lambda_gN < \lambda_N \) holds:

\[
\frac{1}{2} \left| \frac{x^{-1}}{\ln(x) + 1 + A} \right| < \frac{1}{2} \left| \frac{x^{-1}}{\ln(x) + 1 + A} + x^{-1} \right| < \frac{1}{2} \left| \frac{x^{-1}}{\ln(x) + 1 + A} \right|
\]

\[
\left| \frac{\ln(x) + 2 + A}{x} \right| < \left| \frac{1}{\ln(x) + 1 + A} \right| < \left| \frac{1}{\ln(x) + 1 + A} \right| < 1 \quad |\ln(x) + 2 + A| < 1
\]

\[
-1 < \ln(x) + 2 + A < 1
\]

\[
-3 - A < \ln(x) < -1 - A
\]

\[
e^{-3-A} < x < e^{-(1+A)}
\]

Therefore, when the inequality in (12) holds, the Generalised Newton method will take fewer iterations. Numerical examples to confirm this requirement can be seen in Table 15.

The iterative formulas can be seen below.
Table 16: Numerical examples, with a distant $x_0$, comparing the Newton method and the Generalised Newton method for $f(x) = x \ln(x) + Ax + B = 0$ with a precision of $10^{-10}$.

<table>
<thead>
<tr>
<th>Function</th>
<th>$x^*$</th>
<th>In Region</th>
<th>$\lambda_N$</th>
<th>$\lambda_{gN}$</th>
<th>$x_0$</th>
<th>N Iter.</th>
<th>GN Iter.</th>
<th>Iter. %</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x \ln(x) - 6x + 12$</td>
<td>2.32780</td>
<td>×</td>
<td>0.05169</td>
<td>0.16310</td>
<td>20</td>
<td>7</td>
<td>7</td>
<td>100 %</td>
</tr>
<tr>
<td>$x \ln(x) + 2x - 2$</td>
<td>1</td>
<td>×</td>
<td>1/6</td>
<td>2/3</td>
<td>20</td>
<td>7</td>
<td>9</td>
<td>129 %</td>
</tr>
<tr>
<td>$x \ln(x) - 4x + 20$</td>
<td>18.26060</td>
<td>√</td>
<td>0.28746</td>
<td>0.26007</td>
<td>3</td>
<td>9</td>
<td>6</td>
<td>67 %</td>
</tr>
<tr>
<td>$x \ln(x) - 4x + 20$</td>
<td>21.96751</td>
<td>×</td>
<td>0.25413</td>
<td>0.27689</td>
<td>40</td>
<td>8</td>
<td>9</td>
<td>123 %</td>
</tr>
</tbody>
</table>

The Newton method:

$$g_N(x) = x - \frac{x \ln(x) + Ax + B}{\ln(x) + 1 + A} = \frac{x - B}{\ln(x) + 1 + A}$$

The Generalised Newton method:

$$g_{gN}(x) = \exp \left( \ln x - \frac{1}{x} \cdot \frac{x \ln(x) + Ax + B}{\ln(x) + 1 + A} \right)$$

$$= \exp \left( \frac{(\ln x + A)x \ln x - Ax - B}{x \ln(x) + x + Ax} \right)$$

To compare the time it takes to complete one iteration of each method we will set $A = 1$, $B = 1$ and $x = -0.1$. It takes the Newton method $3.7 \cdot 10^{-7}$ seconds to complete one iteration and it takes the Generalised Newton method $8.3 \cdot 10^{-7}$ seconds to complete one iteration. This implies the Generalised Newton method is 2.2 times slower than the Newton method. Furthermore for the Generalised Newton method to be more efficient than the Newton method it must perform less than 44% of the iterations taken by the Newton method. Which never happened in the examples taken though when global convergence is considered the Generalised Newton method could be a more efficient method as it could have a better fit to the function. The same numerical examples repeated with further starting points can be seen in Table 16. These results show for this function, the Generalised Newton method does not seem to have a more efficient global convergence. Perhaps with a different $s(x)$ we can achieve better global convergence.

### 4.5 An equation involving exponential

Consider the function $f(x) = e^x + Ax + B$. Hence, $f'(x) = e^x + A$ and $f''(x) = e^x$. Take $s(x) = e^x$. The asymptotic error constants at $x = x^*$ are:

$$\lambda_N = \frac{1}{2} \left| \frac{e^x}{e^x + A} \right| \quad \lambda_{gN} = \frac{1}{2} \left| \frac{e^x}{e^x + A} - 1 \right|$$
Next, we compute when $\lambda_{gN} < \lambda_N$ holds:

\[
\frac{\lambda_{gN}}{\lambda_N} < \frac{\left| e^x \right|}{\left| e^x + A \right|} \quad \text{for} \quad e^x > |A|, \quad x > \ln |A|
\]

Table 17: Numerical examples comparing the Newton method and the Generalised Newton method for $f(x) = e^x + Ax + B = 0$ with a precision of $10^{-10}$.

| Function | $x^*$ | $\ln |A|$ | $\lambda_N$ | $\lambda_{gN}$ | $x_0$ | N Iter. | GN Iter. |
|----------|-------|----------|-------------|--------------|------|---------|----------|
| $e^x - 3x - 2$ | -0.45523 | 1.09861 | 0.13406 | 0.63406 | -1 | 5 | 6 |
| $e^x - 3x - 2$ | 2.12539 | 1.09861 | 0.77901 | 0.27901 | 3 | 6 | 5 |
| $e^x + 2x - 5$ | 1.05870 | 0.69315 | 0.29519 | 0.20481 | 2 | 6 | 5 |
| $e^x + 4x + 9$ | -2.27568 | 1.38629 | 0.02443 | 0.47557 | -2 | 4 | 5 |

The requirement stated in (13) can be used after finding an approximate region of the root with the bisection method and once determining where it lies in relation to the $\ln |A|$ the more efficient method can be chosen.

Next we will look at the iterative formula for each method.

The Newton method:

\[
g_N(x) = x - \frac{e^x + Ax + B}{e^x + A} = \frac{(x - 1)e^x - B}{e^x + A}
\]

The Generalised Newton method:

\[
g_{gN}(x) = \ln \left( \frac{e^x - e^x e^x + Ax + B}{e^x + A} \right) = \ln \left( \frac{e^x (A - Ax - B)}{e^x + A} \right) = x + \ln \left( \frac{A - Ax - B}{e^x + A} \right)
\]

To compare the time it takes to complete one iteration of each method we will set $A = 1$, $B = 1$ and $x = -0.1$. It takes the Newton method $1.5 \cdot 10^{-7}$ seconds to complete one iteration and it takes the Generalised Newton method $1.8 \cdot 10^{-7}$ seconds to complete one iteration. This implies the Generalised Newton method is 1.2 times slower than the Newton method. Furthermore for the Generalised Newton method to be more efficient than the Newton method it must perform less than 83% of the iterations taken by the Newton method. The same numerical examples with further initial iterates can be seen in Table 18. These experiments show in most cases...
Table 18: Numerical examples, with distant \( x_0 \), comparing the Newton method and the Generalised Newton method for \( f(x) = e^x + Ax + B = 0 \) with a precision of \( 10^{-10} \).

| Function    | \( x^* \)    | \( \ln |A| \) | \( \lambda_N \) | \( \lambda_{gN} \) | \( x_0 \) | N Iter. | GN Iter. | Iter. % |
|-------------|--------------|-------------|----------------|----------------|---------|---------|----------|--------|
| \( e^x - 3x - 2 \) | -0.45523     | 1.09861    | 0.13406       | 0.63406       | -9      | 5       | 11       | 220%   |
| \( e^x - 3x - 2 \) | 2.12539      | 1.09861    | 0.77901       | 0.27901       | 12      | 16      | 7        | 44%    |
| \( e^x + 2x - 5 \) | 1.05870      | 0.69315    | 0.29519       | 0.20481       | 10      | 14      | 8        | 57%    |
| \( e^x + 4x + 9 \) | -2.27568     | 1.38629    | 0.02443       | 0.47557       | -10     | 4       | 10       | 250%   |

the Generalised Newton method to be more effective. In two cases the Newton method greatly outperformed the Generalised Newton method. This is seen to happen when the initial iterate is taken to be less than the solution. This fact would be interesting to look into in relation to the interval of convergence of the Generalised Newton method.

5 The Lambert W Function

Consider the function \( f(x) = xe^x + A \). \( x \) here is the (implicit) Lambert W function and \( A \) is a real constant. The Lambert W function is widely encountered in mathematical models such as those for epidemic processes, including spread of rumours, signal processing and population growth [5]. Both MATLAB and MAPLE have commands to compute Lambert W functions using iterative techniques.

Now, \( f'(x) = e^x + xe^x \) and \( f''(x) = 2e^x + xe^x \). Take \( s(x) = e^x \). The asymptotic error constants at \( x = x^* \) are:

\[
\lambda_{gN} = \frac{1}{2} \left| \frac{2e^x + xe^x}{e^x + xe^x} - 1 \right| \quad \lambda_N = \frac{1}{2} \left| \frac{2e^x + xe^x}{e^x + xe^x} \right|
\]

Next, we compute when \( \lambda_{gN} < \lambda_N \) holds:

\[
\lambda_{gN} < \lambda_N \quad \left| \frac{1}{1 + x} \right| < \left| 2 + x \right| \quad \frac{1}{1 + x} < \frac{1}{1 + x}
\]

Thus, when \( x < -3 \) or \( x > -1 \), the Generalised Newton method will take fewer iterations.

The Generalised Newton method converges in fewer iterations than the Newton method outside the interval between the two triangle markers on the Figure 1 of the Lambert-W function. As we are finding the root of the equation Figure 2, is a more relevant graph which shows the less general case where \( A = 0 \). Numerical examples for this case can be seen in Table 19.

To determine which method is more efficient the cost of the operations performed each iteration must also be compared. To find the roots of the equation \( f(x) = xe^x + A \), the Newton
method carries out the iterations,

\[ x_{n+1} = x_n - \frac{x_ne^{x_n} + A}{(1 + x_n)e^{x_n}} \]

\[ = x_n - \frac{x_n + Ae^{-x_n}}{1 + x_n} \]

and the Generalised Newton method computes the iterations,

\[ x_{n+1} = \ln(e^{x_n} - e^{x_n} \frac{x_ne^{x_n} + A}{(1 + x_n)e^{x_n}}) \]

\[ = \ln\left(\frac{e^{x_n} - A}{1 + x_n}\right). \]

It can be easily identified that the Generalised Newton method is more expensive to compute than the Newton method, due to the natural log being computed. To determine how much more expensive the Generalised Newton method is, a numerical experiment was done in MATLAB where

\[ \text{Newton Iter.: } \quad x - \frac{x + e^{-x}}{1 + x} \quad \text{and} \quad \ln\left(\frac{e^{x_n} - 1}{1 + x_n}\right) : \text{Generalised Newton Iter.} \]
Figure 2: $f(x) = xe^x$

<table>
<thead>
<tr>
<th>Function</th>
<th>$x^*$</th>
<th>$\lambda_N$</th>
<th>$\lambda_{gN}$</th>
<th>$x_0$</th>
<th>N Iter.</th>
<th>GN Iter.</th>
<th>Iter. %</th>
</tr>
</thead>
<tbody>
<tr>
<td>$xe^x - 5$</td>
<td>1.32672</td>
<td>0.71489</td>
<td>0.21489</td>
<td>15</td>
<td>21</td>
<td>11</td>
<td>52%</td>
</tr>
<tr>
<td>$xe^x + 0.1$</td>
<td>-3.57715</td>
<td>0.30599</td>
<td>0.19401</td>
<td>-5.5</td>
<td>8</td>
<td>6</td>
<td>75%</td>
</tr>
<tr>
<td>$xe^x + 0.1$</td>
<td>-0.11183</td>
<td>1.06296</td>
<td>0.56296</td>
<td>12</td>
<td>21</td>
<td>12</td>
<td>57%</td>
</tr>
<tr>
<td>$xe^x + 0.3$</td>
<td>-1.78134</td>
<td>0.13993</td>
<td>0.63993</td>
<td>-3.3</td>
<td>8</td>
<td>6</td>
<td>63%</td>
</tr>
<tr>
<td>$xe^x + 0.3$</td>
<td>-0.48940</td>
<td>1.47924</td>
<td>0.97924</td>
<td>12</td>
<td>22</td>
<td>13</td>
<td>59%</td>
</tr>
</tbody>
</table>

Table 20: Numerical examples, for a distant $x_0$, comparing the Newton method and the Generalised Newton method with a precision of $10^{-10}$.

for $x = -0.1$ were looped one billion times each and the times taken to complete these loops were recorded. It took the Newton method $1.8 \cdot 10^{-7}$s per iteration and the Generalised Newton method took $3 \cdot 10^{-7}$s per iteration. This implies that one Generalised Newton iteration is roughly 1.7 time slower than one Newton iteration. So the Generalised Newton method must take at most 60% of the iterations the Newton method takes to make it comparably efficient.

In the previous numerical examples in this section it is seen that the Generalised takes at least 80% of the number of iterations of the Newton method, which is a too small of a reduction. Suppose we change the starting points and retest the previous functions for distant $x_0$. This can be seen in Table 20. It can be seen for most of the roots the global convergence of the Generalised Newton method is more efficient. The times were it failed to be more efficient were in times were there were two roots. This is due to the small range of convergence to the smaller of the to roots. Due to the a distant enough initial iterate can not be taken for it to balance out the expense of a Generalised Newton iteration. These numerical experiments can be seen graphically in Figure 3, 4, 5, 6 and 7, where this iterations of the two methods are plotted.

From looking at these numerical examples we were able to develop Proposition 2 and 3.

**Proposition 2.** The update in the Newton iteration is

$$|x_{n+1} - x_n| = \left| \frac{x_n + Ae^{-x_n}}{1 + x_n} \right| < 1.$$

for all $x_n > x^* > 0$. 

26
Figure 3: Graph of the function $f(x) = xe^x - 5$ with the Newton method iterations points plotted in red and the Generalised Newton iteration points plotted in black to find root $x = 1.32672$

Figure 4: Graph of the function $f(x) = xe^x + 0.1$ with the Newton method iterations points plotted in red and the Generalised Newton iteration points plotted in black to find root $x = -3.57715$
Figure 5: Graph of the function $f(x) = xe^x + 0.1$ with the Newton method iterations points plotted in red and the Generalised Newton iteration points plotted in black to find root $x = -0.11183$

Figure 6: Graph of the function $f(x) = xe^x + 0.3$ with Newton iteration points plotted with the Newton method iterations points plotted in red and the Generalised Newton iteration points plotted in black to find root $x = -1.78134$
Figure 7: Graph of the function $f(x) = xe^x + 0.3$ with the Newton method iterations points plotted in red and the Generalised Newton iteration points plotted in black to find root $x = -0.48940$

**Proposition 3.** If $x_n > 0$ then, Newton update is less for $f(x) = xe^x = 0$ $(A = 0)$ than the Generalised Newton update,

$$\lambda_{gN} = \frac{1}{2} \frac{2e^x + xe^x}{e^x + xe^x} \quad \lambda_N = \frac{1}{2} \frac{2e^x + xe^x}{e^x + xe^x}$$

In this section we have only considered the assignment of $s(x) = e^x$. This selection has been seen to perform better than the Newton method in various cases. Suppose instead we chose another function which is similar to the exponential function.

Take $s(x) = \tan x$. Hence, $s'(x) = \sec^2 x$ and $s''(x) = 2\sec^2 x \tan x$ The asymptotic error constants at $x = x^*$ are:

$$\lambda_{gN} = \frac{1}{2} \frac{2 + x - 2 \tan x}{1 + x} \quad \lambda_N = \frac{1}{2} \frac{2 + x}{1 + x}$$

Next, we compute when $\lambda_{gN} < \lambda_N$ holds:

$$\lambda_{gN} < \lambda_N \quad |2 + x - 2 \tan x - 2x \tan x| \left|\frac{1}{1 + x}\right| < |2 + x| \left|\frac{1}{1 + x}\right|$$

Hence when the inequality in (14) holds the Generalised Newton method will take fewer iterations locally compared the Newton method.
To determine if $\tan x$ is a better choice of an auxiliary function in the Generalised Newton method than $e^x$, let $s_1(x) = e^x$ and $s_2(x) = \tan x$.

\[
\lambda_{gN1} > \lambda_{gN2}
\]
\[
\left| \frac{1}{1+x} \right| > |2 + x - 2\tan x - 2x\tan x| \left| \frac{1}{1+x} \right|
\]
\[
1 > |2 + x - 2\tan x - 2x\tan x|
\]

Therefore when the inequality in (15) holds, $\tan(x)$ should be assigned to $s(x)$. Some of the values of $x$ where $\tan x$ has a better convergence rate than $e^x$ can be seen in Figure 8. It can be observed that at certain points in Figure 8, $|2 + x - 2\tan x - 2x\tan x|$ is equal to zero. Hence this implies that the asymptotic error constant at these points are also equal to zero. This could probably indicate instances where the order of convergence is greater than two.

Take $s(x) = \sinh x$. Hence, $s'(x) = \cosh x$ and $s''(x) = \sinh x$ The asymptotic error constants at $x = x^*$ are:

\[
\lambda_{gN} = \frac{1}{2} \left| \frac{2e^x + xe^x}{e^x + xe^x} - \frac{\sinh x}{\cosh x} \right|
\]
\[
\lambda_N = \frac{1}{2} \left| \frac{2e^x + xe^x}{e^x + xe^x} \right|
\]
\[
\lambda_{gN} = \frac{1}{2} \left| \frac{2 + x}{1 + x} - \tanh x \right|
\]
\[
\lambda_N = \frac{1}{2} \left| \frac{2 + x}{1 + x} \right|
\]
\[
\lambda_{gN} = \frac{1}{2} \left| \frac{2 + x - \tanh x - x\tanh x}{1 + x} \right|
\]
\[
\lambda_N = \frac{1}{2} \left| \frac{2 + x}{1 + x} \right|
\]
Next, we compute when $\lambda_{gN} < \lambda_N$ holds:

$$\lambda_{gN} < \lambda_N \quad \iff \quad \left| 2 + x + 1 - \frac{2}{1 + e^{-2x}} + x - \frac{2x}{1 + e^{-2x}} \right| \left| \frac{1}{1 + x} \right| < \left| 2 + x \right| \left| \frac{1}{1 + x} \right|$$

$$\left| 3 + 2x - \frac{2 + 2x}{1 + e^{-2x}} \right| < \left| 2 + x \right|$$ (16)

Hence, from (16), the Generalised Newton method will take fewer iterations locally when $x > 0$ or $x \in (-1.68187, -1)$. This is a smaller range of values than taking $s(x) = e^x$. To determine when $\sinh x$ has a better local convergence rate let $s_1(x) = e^x$ and $s_2(x) = \sinh x$

$$\lambda_{gN1} > \lambda_{gN2} \quad \iff \quad \left| \frac{1}{1 + x} \right| > \left| 3 + 2x - \frac{2 + 2x}{1 + e^{-2x}} \right| \left| \frac{1}{1 + x} \right|$$

$$1 > \left| 3 + 2x - \frac{2 + 2x}{1 + e^{-2x}} \right|$$ (17)

Hence, from (17), when $x \in (-2.01768, -1)$ this can be seen in Figure 9. It can be observed that

Figure 9: Graph of $f(x) = \left| 3 + 2x - (2 + 2x)/(1 + e^{-2x}) \right|$ and $g(x) = 1$

at $x = -1.52474$ in Figure 9, $\left| 3 + 2x - (2 + 2x)/(1 + e^{-2x}) \right|$ is equal to zero. Hence this implies that the asymptotic error constant at this point is also equal to zero. This could probably indicate an instance where the order of convergence is greater than two.

6 Further Examples With Higher Precision

The following examples in Table 21 were taken from a paper which proposed an alternative Newton method [6].
It can be seen that is it not always obvious which function should be assigned to $s(x)$ when dealing with equations with contain a mixture of polynomial, trigonometric, logarithm and exponential variables. Though the Generalised Newton method is seen to perform better than the Newton method in most cases when the most appropriate $s(x)$ is chosen.

### 7 Multivariate Generalised Newton Method

Consider the multivariate problem,

$$f_1(x_1, \ldots, x_m) = 0$$

$$\vdots$$

$$f_m(x_1, \ldots, x_m) = 0.$$  

This can be written as

$$\mathbf{f}(\mathbf{x}) = 0,$$

where

$$\mathbf{f} = \begin{bmatrix} f_1(\mathbf{x}) \\ \vdots \\ f_m(\mathbf{x}) \end{bmatrix} \quad \text{and} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}.$$  

Through a similar procedure as the single variable case, the Generalised Newton method is given as

$$x_{n+1} = s^{-1} \left( s(x_n) - J_s(x_n) [J_f(x_n)]^{-1} f(x_n) \right) =: g(x).$$
where
\[
\begin{bmatrix}
s_1(x) \\
\vdots \\
s_m(x)
\end{bmatrix}, \quad
\begin{bmatrix}
\frac{\partial f_1}{\partial x_1} \cdots \frac{\partial f_1}{\partial x_m} \\
\vdots \\
\frac{\partial f_m}{\partial x_1} \cdots \frac{\partial f_m}{\partial x_m}
\end{bmatrix},
\]
\[
J_s(x) = \begin{bmatrix}
\frac{\partial s_1}{\partial x_1} \cdots \frac{\partial s_1}{\partial x_m} \\
\vdots \\
\frac{\partial s_m}{\partial x_1} \cdots \frac{\partial s_m}{\partial x_m}
\end{bmatrix},
\]
and
\[
J_f(x) = \begin{bmatrix}
\frac{\partial f_1}{\partial x_1} \cdots \frac{\partial f_1}{\partial x_m} \\
\vdots \\
\frac{\partial f_m}{\partial x_1} \cdots \frac{\partial f_m}{\partial x_m}
\end{bmatrix},
\]
\[
J_g(x^*) = I - J_{s(x^*)}^{-1} f(x).
\]

**Lemma 7.** Suppose that \( f, s \in C^2[0, 1] \), \( f(x^*) = 0 \), and \( J_s(x^*), J_f(x^*) \) are nonsingular. Then \( J_g(x^*) = 0 \).

**Proof.**
\[
s(g(x)) = s(x) - J_s(x) [J_f(x)]^{-1} f(x)
\]
\[
J_s(x^*)J_g(x^*) = J_s(x^*) - \frac{\partial}{\partial x} \left[ J_s(x) [J_f(x)]^{-1} f(x) \right]_{x=x^*}
\]
\[
J_g(x^*) = I - J_s^{-1}(x^*) \left( \left[ \frac{\partial}{\partial x} J_s(x^*) \right] [J_f(x^*)]^{-1} f(x^*) \)
\]+J_s(x^*) \left[ \frac{\partial}{\partial x} [J_f(x^*)]^{-1} f(x^*) + J_s(x^*) [J_f(x^*)]^{-1} J_f(x^*) \right)
\]
\[
J_g(x^*) = I - I = 0.
\]

This proves the necessary condition for quadratic convergence. Further work needs to be done to prove quadratic convergence.

Setting \( s(x) = x \), one gets the Newton method:
\[
x_{n+1} = x_n - [J_f(x_n)]^{-1} f(x_n)
\]

There was not sufficient time to develop a formula for an asymptotic error constant for the multivariate case for the Generalised Newton method. Due to this, proper analysis on when the Generalised Newton method is the preferable method could not be done. Instead, some numerical experiments were done to compare the methods, which can be seen in Table 22. The first three of these functions were taken from a paper containing test problems [7] and the last three were examples constructed for this research.

Let us observe a specific example from Table 22. Let us consider the system of functions
\[
\begin{bmatrix}
x_2 e^{x_1} - 0.5 \\
x_1 e^{x_2} - 1.5
\end{bmatrix}
\]

This system can be seen plotted in Figure 10 and 11.

Unlike the single variable case, the multivariate Generalised Newton method does not have an easy to determine range of convergence. So when we considered a distant initial iterate, it
<table>
<thead>
<tr>
<th>Functions</th>
<th>$x^*$</th>
<th>$x_0$</th>
<th>$s(x)$</th>
<th>N Iter.</th>
<th>GN Iter.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Rosenbrock Function</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\begin{bmatrix} 10(x_2 - x_1^2) \ 1 - x_1 \end{bmatrix} = 0$</td>
<td>$\begin{bmatrix} 1 \ 1 \end{bmatrix}$</td>
<td>$\begin{bmatrix} -1.2 \ 1 \end{bmatrix}$</td>
<td>$\begin{bmatrix} x_1^2 \ x_2 \end{bmatrix}$</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td><strong>Freudenstein and Roth Function</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\begin{bmatrix} x_1 - x_2(2 - x_2(5 - x_2)) - 13 \ x_1 - x_2(14 - x_2(1 + x_2)) - 29 \end{bmatrix} = 0$</td>
<td>$\begin{bmatrix} 5 \ 4 \end{bmatrix}$</td>
<td>$\begin{bmatrix} 0.5 \ 2 \end{bmatrix}$</td>
<td>$\begin{bmatrix} x_1 \ x_2^3 \end{bmatrix}$</td>
<td>43</td>
<td>10</td>
</tr>
<tr>
<td><strong>Brown almost linear Function</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\begin{bmatrix} x_1 + x_2 - 3 \ x_1 x_2 - 1 \end{bmatrix} = 0$</td>
<td>$\begin{bmatrix} 2.61803 \ 0.38197 \end{bmatrix}$</td>
<td>$\begin{bmatrix} 4 \ 1 \end{bmatrix}$</td>
<td>$\begin{bmatrix} x_1^2 \ x_2 \end{bmatrix}$</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>$\begin{bmatrix} x_2 e^{x_1} - x_2 \ \ln(x_2) + x_1 \end{bmatrix} = 0$</td>
<td>$\begin{bmatrix} 0 \ 1 \end{bmatrix}$</td>
<td>$\begin{bmatrix} -1 \ 2 \end{bmatrix}$</td>
<td>$\begin{bmatrix} e^{x_1} \ \ln(x_2) \end{bmatrix}$</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>$\begin{bmatrix} e^{x_2} + \ln(x_1) + x_1 \ e^{x_1} + \ln(x_2) + x_2 \end{bmatrix} = 0$</td>
<td>$\begin{bmatrix} 0.22715 \ 0.22715 \end{bmatrix}$</td>
<td>$\begin{bmatrix} 0.8 \ 0.8 \end{bmatrix}$</td>
<td>$\begin{bmatrix} \ln(x_1) \ \ln(x_2) \end{bmatrix}$</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>$\begin{bmatrix} x_2 e^{x_1} - 0.5 \ x_1 e^{x_2} - 1.5 \end{bmatrix} = 0$</td>
<td>$\begin{bmatrix} 1.31084 \ 0.13480 \end{bmatrix}$</td>
<td>$\begin{bmatrix} 1 \ 0 \end{bmatrix}$</td>
<td>$\begin{bmatrix} e^{x_1} \ e^{x_2} \end{bmatrix}$</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

Table 22: Numerical examples comparing the Newton method and the Generalised Newton method with a precision of $10^{-10}$ for the multivariate case.

Figure 10: Iterates clustered at solution

Figure 11: Rotated view
Table 23: Multivariate system solved by both methods by various $x_0$

<table>
<thead>
<tr>
<th>$x_0$</th>
<th>Newton Iterations</th>
<th>Gen. Newton Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.5</td>
<td>13</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>18</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>14</td>
<td>17</td>
</tr>
</tbody>
</table>

Figure 12: Contour plot with initial iterate $x_1 = 4.5$ and $x_2 = 4$

cannot always be said that one method will converge in fewer iterations globally. This can be seen in Table 23, which considers the system 18 which has the solution

$$\begin{bmatrix} 1.31084 \\ 0.13480 \end{bmatrix}.$$

For this numerical experiment,

$$s(x) = \begin{bmatrix} e^{x_1} \\ e^{x_2} \end{bmatrix}.$$

These examples can be shown pictorially through contour plots of the infinity norm of the function $f$,

$$h(p) = ||f(p)||_\infty = \max\{|x_2e^{x_1} - 0.5|, \ |x_1e^{x_2} - 1.5|\},$$

in Figures 12, 13 and 14 with the Newton method plotted in black and the Generalised Newton method in magenta.
Figure 13: Contour plot with initial iterate $x_1 = 5$ and $x_2 = 2$

Figure 14: Contour plot with initial iterate $x_1 = 5$ and $x_2 = 5$
8 Discussion and Conclusion

We have devised conditions under which the Generalised Newton method converges in fewer iterations than the Newton method for a comprehensive list of nonlinear equations. Though through comparison of computational times we have seen that the Generalised Newton method is, in most cases, only more efficient in global convergence due to it being more expensive to compute. We have also developed a multivariate version of the Generalised Newton method and have shown that it can converge in fewer iterations than the multivariate Newton method.

Further work needs to be done on the multivariate method. Only the necessary condition of quadratic convergence has been proved, more work is needed to be done to complete the convergence rate proof. The asymptotic error constant should also be found so proper comparison of the two methods in the multivariate case can be undertaken. This would allow one to identify conditions under which the Generalised Newton method is preferable to the Newton method. Further numerical experiments must also be done as we only looked at two variable examples.

A further line of research for both the single and multivariate cases would be to identify the region of convergence of the Generalised Newton method.

Yet another line of research would be to investigate the benefits of the Generalised Newton method on equations which have a nonlinear term followed by terms with very small coefficients plus a large coefficient (e.g. \( x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0 = 0 \) where \( a_0 \) is much greater than \( a_1, \ldots, a_{n-1} \)). This would be interesting to investigate as in these cases the conditions found by comparing the asymptotic error constants showed the Generalised Newton method to converge in far fewer iterations.
References


