Quest Journals
Journal of Research in Applied Mathematics
Volume 12 ~ Issue 7 (July 2026) pp: 31-41
ISSN (Online): 2394-0743 ISSN (Print): 2394-0735
www.questjournals.org

Research Paper

Numerical Solution of Nonlinear Systems and Boundary Value Problems Using Newton, Broyden, and Finite-Difference Methods

Mawa Adam Abd Allah Ahmed [(1)]
Dhofar Governorate ' Ministry of Education Mathematics Teacher
Budoor Mohammed Abdelati, University of Nile Valley, College of Education, Sudan

Abstract
This paper examines three fundamental numerical methods for approximating solutions to nonlinear problems. It reviews Newton's method (quadratic convergence) for multivariable systems, Broyden's method (superlinear convergence) as a computationally efficient Quasi-Newton alternative, and the Finite-Difference method for solving nonlinear boundary value problems (BVPs) in ODEs. A numerical example illustrates Newton's method, while Crout LU factorization is presented for efficiently solving the tridiagonal systems arising from the Finite-Difference approach. The study compares convergence properties, advantages, and limitations of each method.

Keywords: *Nonlinear systems, Newton's method, Broyden's method, Finite-Difference method, convergence, Jacobian matrix, Crout LU factorization, boundary value problems.*

Received 09 July., 2026; Revised 18 July, 2026; Accepted 20 July, 2026
Published with open access at ***www.questjournals.org***

I. Introduction

Over the years, we have been taught on how to solve equations using various algebraic methods. These methods include the substitution method and the elimination method. Other algebraic methods that can be executed include the quadratic formula and factorization. In Linear Algebra, we learned that solving systems of linear equations can be implemented by using row reduction as an algorithm. However, when these methods are not successful, we use the concept of numerical methods. Numerical methods are used to approximate solutions of equations when exact solutions cannot be determined via algebraic methods. They construct successive approximations that converge to the exact solution of an equation or system of equations. We focused on solving nonlinear equations involving only a single variable. We used methods such as Newton's method, the Secant method, and the Bisection method. We also examined numerical methods such as the Runge-Kutta methods, that are used to solve initial-value problems for ordinary differential equations. However, these problems only focused on solving nonlinear equations with only one variable, rather than nonlinear equations with several variables. The goal is to examine three different numerical methods that are used to solve systems of nonlinear equations in several variables. The first method we will look at is Newton's method. This will be followed by Broyden's method, which is sometimes called a Quasi-Newton method; it is derived from Newton's method. Lastly, we will study the Finite Difference method that is used to solve boundary value problems of nonlinear ordinary differential equations. For each method, a breakdown of each numerical procedure will be provided. [2]

Preliminaries (1.1) [3] We present the definitions and terms that will be used throughout the project will be presented.

Definition (1.2) [3] A function f: $R^n \to R$ is defined as being nonlinear when it does not satisfy the superposition principle that is $f(x_1 + x_2 + ...) \neq f(x_1) + f(x_2) + ...$

Now that we know what the term nonlinear refers to we can define a system of nonlinear equations.

Definition (1.3) [3] A system of nonlinear equations is a set of equations as the following:

$$f_1(x_1, x_2, ..., x_n) = 0,$$
$$f_2(x_1, x_2, ..., x_n) = 0,$$
$$\vdots$$
$$f_n(x_1, x_2, ..., x_n) = 0,$$

where $(x_1, x_2, ..., x_n) \in R^n$ and each f_i is a nonlinear real function, i = 1,2, ..., n.

Example (1.3) [3] Here is an example of a nonlinear system from Burden and Faires:

$$\begin{aligned} 3x_1 - \cos(x_2x_3) - \frac{1}{2} &= 0 \\ x_1^2 - 81(x_2 + 0.1)^2 + \sin x_3 + 1.06 &= 0 \\ e^{-x_1x_2} + 20x_3 + \frac{10\pi - 3}{3} &= 0 \end{aligned} \tag{1}$$

We will use the term root or solution frequently to describe the final result of solving the systems.

Definition (1.4) [3] A solution of a system of equations $f_1, f_2, ..., f_n$ in n variables is a point $(a_1,...,a_n) \in R^n$ such that $f_1(a_1,...,a_n) = \cdots = f_n(a_1,...,a_n) = 0$. Because systems of nonlinear equations cannot be solved as nicely as linear systems, we use procedures called iterative methods.

Definition (1.5) [3] An iterative method is a procedure that is repeated over and over again, to find the root of an equation or find the solution of a system of equations.

Definition (1.6) [3] Let F be a real function from $D \subset R^n$ to R^n. If $F(p) = p$, for some $p \in D$, then p is said to be a fixed point of F.

II. Convergence

We will discuss is the convergence of each of the numerical methods.

Definition (2.1) [5] We say that a sequence converges if it has a limit.

Definition (2.2) [5] Let p_n be a sequence that converges to p, where $p_n \neq p$. If constants $\lambda, \alpha > 0$ exist such that

$$\lim_{n\to\infty} \frac{|p_{n+1} - p|}{|p_n - p|^\alpha} = \lambda.$$

Then it is said that p_n converges to p of order α with a constant λ.

There are three different orders of convergences.

Definition (2.3) [5] A sequence p_n is said to be linearly convergent if p_n converges to p with order $\alpha = 1$, for a constant $\lambda < 1$ such that

$$\lim_{n\to\infty} \frac{|p_{n+1} - p|}{|p_n - p|^\alpha} = \lambda$$

Definition (2.4) [5] A sequence p_n is said to be quadratically convergent if p_n converges to p with order $\alpha = 2$ such that

$$\lim_{n\to\infty} \frac{|p_{n+1} - p|}{|p_n - p|^\alpha} = \lambda$$

Definition (2.5) [5] A sequence p_n is said to be super linearly convergent if

$$\lim_{n\to\infty} \frac{|p_{n+1} - p|}{|p_n - p|} = 0$$

Remark (2.6) [5] The value of α measures how fast a sequence converges. Thus, the higher the value of α is, the more rapid the convergence of the sequence is. In the case of numerical methods, the sequence of approximate solutions is converging to the root. If the convergence of an iterative method is more rapid, then a solution may be reached in less interations in comparison to another method with a slower convergence

III. Jacobian Matrix

Definition (3.1) [6] The Jacobian matrix is a matrix of first order partial derivatives

$$J(x) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(x) & \frac{\partial f_1}{\partial x_2}(x) & \cdots & \frac{\partial f_1}{\partial x_n}(x) \\ \frac{\partial f_2}{\partial x_1}(x) & \frac{\partial f_2}{\partial x_2}(x) & \cdots & \frac{\partial f_2}{\partial x_n}(x) \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial f_n}{\partial x_1}(x) & \frac{\partial f_n}{\partial x_2}(x) & \cdots & \frac{\partial f_n}{\partial x_n}(x) \end{bmatrix}.$$

Example (3.2) [6] If we take the system from Example (2.3) we are able to obtain the following Jacobian Matrix:

$$J(\mathbf{x}) = \begin{bmatrix} 3 & x_3\sin(x_2x_3) & x_2\sin(x_2x_3) \\ 2x_1 & -162(x_2+0.1) & \cos x_3 \\ -x_2e^{-x_1x_2} & -x_1e^{-x_1x_2} & 20 \end{bmatrix}$$

IV. Hessian Matrix

The Hessian matrix, will be discussed in a future proof.

Definition(4.1) [6] The Hessian matrix is a matrix of second order partial derivatives

H $= \left[\frac{\partial^2 f}{\partial x_i \partial x_j}\right]_{ij}$ such that

$$H(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f_1}{\partial x_1^2} & \frac{\partial^2 f_1}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f_1}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f_2}{\partial x_2 \partial x_1} & \frac{\partial^2 f_2}{\partial x_2^2} & \cdots & \frac{\partial^2 f_2}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial^2 f_n}{\partial x_n \partial x_1} & \frac{\partial^2 f_n}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f_n}{\partial x_n^2} \end{bmatrix}.$$

Norms of Vectors (4.2) [6] Let $x \in R^n$ where

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Definition (4.3) [6] A vector norm on R^n is a function, $\|\cdot\|$, from R^n into R that has the following properties:

(i) $\|x\| \geq 0$ for all $x \in R^n$,

(ii) $\|x\| = 0$ if and only if $x = 0$,

(iii) $\|\alpha x\| = |\alpha|\|x\|$ for all $\alpha \in R$ and $x \in R^n$,

(iv) $\|x + y\| \leq \|x\| + \|y\|$ for all $x,y \in R^n$

There are two types of vector norms we will discuss, the l_2 and l_∞ norms.

Definition (4.4) [6] The l_2 norm for the vector **x** is called the Euclidean norm because it represents the length of the vector denoted by

$$||\mathbf{x}|| = ||\mathbf{x}||_2 = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

Definition (4.5) [6] The l_∞ norm represents the absolute value of the largest component in the vector **x**. It is denoted by

$$||\mathbf{x}||_\infty = \max_{1 \leq i \leq n} |x_i|.$$

The following is an example demonstrating the vector norms.

Remark (4.6) [6] Rearranging the system in

Step 3: we get that $y^{(0)} = -J(x^{(0)})^{-1} F(x^{(0)})$. The significance of this is that, since $y^{(0)} = -J(x^{(0)})^{-1} F(x^{(0)})$, we can replace $-J(x^{(0)})^{-1} F(x^{(0)})$ in our iterative formula with $y^{(0)}$. This result will yield that, $x^{(k)} = x^{(k-1)} - J(x^{(k-1)})^{-1}F(x^{(k-1)}) = x^{(k-1)} - y^{(k-1)}$

Step 4: Once $y^{(0)}$ is found, we can now proceed to finish the first iteration by solving for $x^{(1)}$.
Thus, using the result from Step 3, we have that

$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \mathbf{y}^{(0)} = \begin{bmatrix} x_1^{(0)} \\ x_2^{(0)} \\ \vdots \\ x_n^{(0)} \end{bmatrix} + \begin{bmatrix} y_1^{(0)} \\ y_2^{(0)} \\ \vdots \\ y_n^{(0)} \end{bmatrix}$$

Step 5: Once we have calculated $\mathbf{x}^{(1)}$, we repeat the process again, until $\mathbf{x}^{(k)}$ converges to x. This indicates we have reached the solution to $\mathbf{F}(\mathbf{x}) = \mathbf{0}$, where x is the solution to the system.

Remark (4.7) [6] When a set of vectors converges, the norm $||x^{(k)} - x^{(k-1)}|| = 0$. This means that

$$||\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}|| = \sqrt{(x_1^{(k)} - x_1^{(k-1)})^2 + \cdots + (x_n^{(k)} - x_n^{(k-1)})^2} = 0$$

Convergence of Newton's Method (4.8) [6] Newton's method converges quadratically. When carrying out this method the system converges quite rapidly once the approximation is close to the actual solution of the nonlinear system. This is seen as a advantage because Newton's method may require less iterations, compared to another method with a lower rate of convergence, to reach the solution. However, when the system does not converge, this is an indicator that an error in the computations has occurred, or a solution may not exist. In the following proof, we will prove that Newton's method does indeed converge quadratically.

Advantages and Disadvantages of Newton's Method (4.9) [6] One of the advantages of Newton's method is that it's not too complicated in form and it can be used to solve a variety of problems. The major disadvantage associated with Newton's method, is that J(x), as well as its inversion has, to be calculated for each iteration. Calculating both the Jacobian matrix and its inverse can be quite time consuming depending on the size of your system is. Another problem that we may be challenged with when using Newton's method is that it may fail to converge. If Newton's method fails to converge this will result in an oscillation between points.

V. A Numerical Example of Newton's Method

The following example is a numerical application of Newton's method.
Example (5.1) [8] Solve the following nonlinear system

$$\begin{aligned} 3x_1 - \cos(x_2 x_3) - \frac{1}{2} &= 0, \\ x_1^2 - 81(x_2 + 0.1)^2 + \sin x_3 + 1.06 &= 0, \\ e^{-x_1 x_2} + 20x_3 + \frac{10\pi - 3}{3} &= 0, \end{aligned}$$

when the initial approximation is

$$\mathbf{x}^{(0)} = \begin{bmatrix} 0.1 \\ 0.1 \\ -0.1 \end{bmatrix}$$

Step 1: We have our initial vector

$$\mathbf{x}^{(0)} = \begin{bmatrix} 0.1 \\ 0.1 \\ -0.1 \end{bmatrix}$$

Step 2: Define F(x) and J(x):

$$\mathbf{F}(\mathbf{x}) = \begin{bmatrix} 3x_1 - \cos(x_2 x_3) - \frac{1}{2} \\ x_1^2 - 81(x_2 + 0.1)^2 + \sin x_3 + 1.06 \\ e^{-x_1 x_2} + 20x_3 + \frac{10\pi - 3}{3} \end{bmatrix}$$

$$J(\mathbf{x}) = \begin{bmatrix} 3 & x_3 \sin(x_2 x_3) & x_2 \sin(x_2 x_3) \\ 2x_1 & -162(x_2 + 0.1) & \cos x_3 \\ -x_2 e^{-x_1 x_2} & -x_1 e^{-x_1 x_2} & 20 \end{bmatrix}$$

Now that we have defined F(x) and J(x), we now want to calculate $F(x^{(0)})$ and $J(x^{(0)})$, where $x^{(0)} = (0.1, 0.1, -0.1)$:

$$\mathbf{F}(\mathbf{x}^{(0)}) = \begin{bmatrix} 0.3 - \cos(-0.01) - \frac{1}{2} \\ 0.01 - 3.24 + \sin(-0.1) + 1.06 \\ e^{(-0.01)} - 2 + \frac{10\pi - 3}{3} \end{bmatrix} = \begin{bmatrix} -1.19995 \\ -2.269833417 \\ 8.462025346 \end{bmatrix}$$

and

$$J(\mathbf{x}^{(0)}) = \begin{bmatrix} 3 & (-0.1)\sin(-0.01) & 0.1\sin(-0.01) \\ 0.2 & -32.4 & \cos(-0.1) \\ -0.1e^{-0.01} & -0.1e^{-0.01} & 20 \end{bmatrix} = \begin{bmatrix} 3 & 0.000999983 & -0.000999983 \\ 0.2 & -32.4 & 0.995004165 \\ -0.099004984 & -0.099004983 & 20 \end{bmatrix}$$

Step 3: Solve the system $J(x^{(0)})y^{(0)} = -F(x^{(0)})$, using Gaussian Elimination:

$$\begin{bmatrix} 3 & 0.000999983 & -0.000999983 \\ 0.2 & -32.4 & 0.995004165 \\ -0.099004984 & -0.099004983 & 20 \end{bmatrix} \begin{bmatrix} y_1^{(0)} \\ y_2^{(0)} \\ y_3^{(0)} \end{bmatrix} = - \begin{bmatrix} -1.19995 \\ -2.269833417 \\ 8.462025346 \end{bmatrix}$$

After solving the linear system above, it yields the result

$$\mathbf{y}^{(0)} = \begin{bmatrix} 0.40003702 \\ -0.08053314 \\ -0.42152047 \end{bmatrix}$$

Step 4: Using the result in Step 3, compute $x^{(1)} = x^{(0)} + y^{(0)}$:

$$\mathbf{x}^{(1)} = \begin{bmatrix} 0.1 \\ 0.1 \\ -0.1 \end{bmatrix} + \begin{bmatrix} 0.40003702 \\ -0.08053314 \\ -0.42152047 \end{bmatrix} = \begin{bmatrix} 0.50003702 \\ 0.01946686 \\ -0.52152047 \end{bmatrix}$$

We can use the results of $x^{(1)}$ to find our next iteration $x^{(2)}$ by using the same procedure.
Step 5: If we continue to repeat the process, we will get the following results:

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\|$
0	0.10000000	0.10000000	-0.10000000	–
1	0.50003702	0.01946686	-0.52152047	0.422
2	0.50004593	0.00158859	-0.52355711	0.0179
3	0.50000034	0.00001244	-0.52359845	0.00158
4	0.50000000	0.00000000	-0.52359877	0.0000124
5	0.50000000	0.00000000	-0.52359877	0

we know that when a set of vectors converges the norm, $\|x(k) - x(k-1)\| = 0$.
Thus, by our table above, the norm is equal to zero at the fifth iteration. This indicates that our system F(x) has converged to the solution, which will be denoted by x. Therefore, from our table of our results we know that

$$\overline{\mathbf{x}} = \begin{bmatrix} 0.50000000 \\ 0.00000000 \\ -0.52359877 \end{bmatrix}$$

is an approximation solution of F(x) = 0.
There are methods that are in the same family of Newton's method, identified as Quasi-Newton methods. A specific Quasi-Newton method, known as Brayden's method, will be examined next.

VI. Brayden's Method

We examined the numerical method known as Newton's method. We established that one of the major disadvantages of this method was that that J(x) and its inverse must be computed at each iteration. We, therefore want to avoid this problem. There are methods known as Quasi-Newton methods, in which Burden and Faires describe as methods that use an approximation matrix that is updated at each iteration in place of the Jacobian matrix. This implies that the form of the iterative procedure for Broyden's method is almost identical to that used in Newton's method. The only exception being that an approximation matrix Ai is implemented instead of J(x). With that said the following equation is derived: $x^{(i+1)} = x^{(i)} - A_i^{-1}F(x^{(i)})$.
This is defined as Brayden's iterative procedure.
Ai is defined as

$$A_i = A_{i-1} + \frac{\mathbf{y}_i - A_{i-1}\mathbf{s}_i}{\|\mathbf{s}_i\|_2^2}\mathbf{s}_i^t$$

$y_i = F(x^{(i)}) - F(x^{(i-1)})$ and $s_i = x^{(i)} - x^{(i-1)}$. However, in Brayden's method it involves that computationA_i^{-1}, not Ai, which brings us to the next theorem.
Theorem (6.1) [8] If A is a nonsingular matrix and x and y are vectors, then $A + xy^t$ is nonsingular provided that $y^tA^{-1}x \neq -1$ and

$$(A + \boldsymbol{x}\boldsymbol{y}^t)^{-1} = A^{-1} - \frac{A^{-1}\boldsymbol{x}\boldsymbol{y}^tA^{-1}}{1 + \boldsymbol{y}^tA^{-1}\boldsymbol{x}}.$$

The Sherman-Morrison Formula, is a matrix inversion formula. It allows A_i^{-1} to be computed directly usingA_{i-1}^{-1}, rather than computing Ai and then its inverse at each iteration. Now by using Theorem (6.1) and letting $A = A_{i-1}, x = \frac{y_i - A_{i-1}S_i}{\|S_i\|_2^2}$, and
$y = s_i$, as well as using A_i as defined above, we have that

$$A_i^{-1} = \left(A_{i-1} + \frac{\mathbf{y}_i - A_{i-1}\mathbf{s}_i}{\|\mathbf{s}_i\|_2^2}\mathbf{s}_i^t\right)^{-1}$$

$$= A_{i-1}^{-1} - \frac{A_{i-1}^{-1}\left(A_{i-1} + \frac{\mathbf{y}_i - A_{i-1}\mathbf{s}_i}{||\mathbf{s}_i||_2^2}\mathbf{s}_i^t\right)A_{i-1}^{-1}}{1 + \mathbf{s}_i^t A_{i-1}^{-1}\left(\frac{\mathbf{y}_i - A_{i-1}\mathbf{s}_i}{||\mathbf{s}_i||_2^2}\right)}$$

$$= A_{i-1}^{-1} - \frac{(A_{i-1}^{-1}y_i - \mathbf{s}_i)\mathbf{s}_i^t A_{i-1}^{-1}}{||\mathbf{s}_i||_2^2 + \mathbf{s}_i^t A_{i-1}^{-1}\mathbf{y}_i - ||\mathbf{s}_i||_2^2}$$

This leaves us with

$$A_i^{-1} = A_{i-1}^{-1} + \frac{(\mathbf{s}_i - A_{i-1}^{-1}\mathbf{y}_i)\mathbf{s}_i^t A_{i-1}^{-1}}{\mathbf{s}_i^t A_{i-1}^{-1}\mathbf{y}_i}.$$

We compute the inverse of the approximation matrix at each iteration with this equation. We know we describe the steps of Brayden's method:

Step 1: Let $x^{(0)} = (x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)})$ be the initial vector given.

Step 2: Calculate $F(x^{(0)})$.

Step 3: We compute A_0^{-1}. Because we do not have enough information to compute A_0 directly, Brayden's method permits us to let $A_0 = J(x^{(0)})$, which implies that

$$A_0^{-1} = J(\mathbf{x}^{(0)})^{-1}.$$

Step 4: Calculate $x^{(1)} = \mathbf{x}^{(0)} - A_0^{-1}\mathbf{F}(\mathbf{x}^{(0)})$.

Step 5: Calculate $F(x^{(1)})$.

Step 6: Take $F(x^{(0)})$ and $F(x^{(1)})$ and calculate $y_1 = F(x^{(1)}) - F(x^{(0)})$. Next, take the first two iterations of $x^{(i)}$ and calculate $s_1 = x^{(1)} - x^{(0)}$.

Step 7:
Calculate . $\mathbf{s}_1^t A_0^{-1}\mathbf{y}_1$

Step 8:
Compute

$$A_1^{-1} = A_0^{-1} + \left(\frac{1}{\mathbf{s}_1^t A_0^{-1}\mathbf{y}_1}\right)\left[(\mathbf{s}_1 - A_0^{-1}\mathbf{y}_1)\mathbf{s}_1^t A_0^{-1}\right]$$

Step 9:
TakeA_i^{-1} that we found in Step 8, and calculate $x^{(2)} = \mathbf{x}^{(1)} - A_1^{-1}\mathbf{F}(\mathbf{x}^{(1)})$.

Step 10:
Repeat the process until we converge to x, i.e. when x(i) = x(i+1) = x. This will indicate that we have reached the solution of the system.

Convergence of Broyden's Method (6.2) [8] Unlike Newton's method, Broyden's method as well as all of the Quasi-Newton methods converge super linearly. This means that

$$\lim_{i\to\infty}\frac{||\mathbf{x}^{(i+1)} - \mathbf{p}||}{||\mathbf{x}^{(i)} - \mathbf{p}||} = 0$$

where p is the solution to F(x) = 0, and $x^{(i)}$ and $x^{(i+1)}$ are successive approximations to p. This can be proved in a similar manner that proved the convergence of Newton's method.

Advantages and Disadvantages of Broyden's Method (6.3) [8] The main advantage of Broyden's method is the reduction of computations. More specifically, the way the inverse of the approximation matrix, A_i^{-1} can be computed directly from the previous iteration, A_i^{-1} reduces the number of computations needed for this method in comparison to Newton's Method. One thing that is seen as a disadvantage of this Quasi-Newton method is that it does not converge quadratically. This may mean that more iterations may be needed to reach the solution, when

compared to the number of iterations Newton's method requires. Another disadvantage of Broyden's method is that as described by Burden and Faires, is that it is not self-correcting. This means that in contrast to Newton's method, it does not correct itself for round off errors with consecutive interations. This may cause only a slight inaccuracy in the iterations compared to Newton's, but the final iteration will be the same. Now that we have taken a look at numerical methods for solving multivariable nonlinear equations, in the next section we will focus on a numerical method that is used to nonlinear boundary value problems for ordinary differential equations.

VII. Finite-Difference Method

We will examine a numerical method that is used to approximate the solution of a boundary-value problem. We will focus on a two-point boundary-value problem with a second order differential equation which takes the form, $y'' = f(x, y, y')$,
$a \le x \le b$, $y(a) = \alpha$, $y(b) = \beta$, where f is a function, a and b are the end points, and $y(a) = \alpha$ and $y(b) = \beta$ are the boundary conditions.

Example (7.1) [8] The following example is of a two-point boundary value problem with a second order differential equation:

$$y'' = \frac{1}{8}(32 + 2x^3 - yy'),\ 1 \le x \le 3$$

$$y(1) = 17,\ y(3) = \frac{43}{3}$$

Before we can solve a boundary value problem we have to be sure it has a unique solution. The following theorem ensures that a solution indeed does exist and is unique.

Theorem (7.2) [8] Suppose the function f in the boundary-value problem

$$y'' = f(x, y, y'), \quad a \le x \le b, \quad y(a) = \alpha, y(b) = \beta$$

is continuous on the set, $D = ((x, y, y') | a \le x \le b, -\infty < y < \infty, -\infty < y' < \infty)$
and that the partial derivatives f_y and $f_{y'}$ are also continuous in D. If
(i) $f_y(x, y, y') > 0$ for all $(x, y, y') \in D$, and
(ii) a constant M exists with $|f_{y'}(x, y, y')| \le M$ for all $(x, y, y') \in D$,
then the boundary-value problem has a unique solution.

The numerical method we will be looking at is the Finite-Difference method. This method can be used to solve both linear and nonlinear ordinary differential equations. We will just survey the nonlinear Finite-Difference method.

A nonlinear boundary-value problem takes on the form of $y'' = f(x, y, y')$, $a \le x \le b$, $y(a) = \alpha$, $y(b) = \beta$. In order for the Finite-Difference method to be carried out we have to assume f satisfies the following conditions:
(i) f and the partial derivatives f_y and $f_{y'}$ are all continuous on

$$D = ((x, y, y') | a \le x \le b, -\infty < y < \infty, -\infty < y' < \infty)$$

(ii) $f_y(x, y, y') \ge \delta$ on D, for some $\delta > 0$.
(iii) Constants k and L exist, with

$$k = \max_{(x,y,y') \in D} |f_y(x, y, y')|,\ and\ L = \max_{(x,y,y') \in D} |f_{y'}(x, y, y')|$$

With f satisfying these conditions, Theorem (7.2) implies that a unique solution exists. When solving a linear boundary-value problem using the Finite-Difference, the second-order boundary-value equation, $y'' = p(x)y' + q(x)y + r(x)$, is expanded using y in a third Taylor polynomial about x_i evaluated at x_{i+1} and x_{i-1}, where a formula called the centered-difference formula for both $y''(x_i)$ and $y'(x_i)$ is derived. Burden and Faires define the centered-difference formula for $y''(x_i)$ and $y'(x_i)$ as follows

(2)

for some ξ_i in (x_{i-1}, x_{i+1}), and

(3)

for some η_i in (x_{i-1}, x_{i+1}).

$$y'(x_i) = \frac{1}{2h}[y(x_{i+1})| - y(x_{i-1})] - \frac{h^2}{6}y'''(\eta_i)$$

Now we can begin to form the procedure for the Finite-Difference method.

$$y''(x_i) = \frac{1}{h^2}[y(x_{i+1}) - 2y(x_i) + y(x_{i-1})] - \frac{h^2}{12}y^{(4)}(\xi_i)$$

Step 1: We first want to divide the interval [a, b] into (N + 1) equal subintervals which gives us

$$h = \frac{(b-a)}{(N+1)}$$

with end points at $x_i = a + ih$ for i = 0, 1, 2, ..., N + 1.

Step 2: Next, we will take, $y''(x_i) = f(x_i, y(x_i), y'(x_i))$

and substitute equations (2) and (3) into it. This will give us:

$$\frac{y(x_{i+1}) - 2y(x_i) + y(x_{i-1})}{h^2} = f\left(x_i, y(x_i), \frac{y(x_{i+1}) - y(x_{i-1})}{2h} - \frac{h^2}{6}y'''(\eta_i)\right) + \frac{h^2}{12}y^{(4)}(\xi_i) \tag{4}$$

for some ξ_i and η_i in the interval (x_{i-1}, x_{i+1}).

Step 3: The Finite-Difference method results by using (3), and the boundary conditions to define: $w_0 = \alpha$, $w_{N+1} = \beta$

and

$$-\frac{w_{i+1} - 2w_i + w_{i-1}}{h^2} + f\left(x_i, w_i, \frac{w_{i+1} - w_{i-1}}{2h}\right) = 0$$

for each i = 1, 2, ..., N.

Step 4: Once we define the boundary conditions in Step 3, an N × N nonlinear system, F(w), is produced from the Finite Difference method defined as:

$$\begin{aligned}
2w_1 - w_2 + h^2 f\left(x_1, w_1, \frac{w_2 - \alpha}{2h}\right) - \alpha &= 0 \\
-w_1 + 2w_2 - w_3 + h^2 f\left(x_2, w_2, \frac{w_3 - w_1}{2h}\right) &= 0 \\
&\vdots \\
-w_{N-2} + 2w_{N-1} - w_N + h^2 f\left(x_{N-1}, w_{N-1}, \frac{w_N - w_{N-2}}{2h}\right) &= 0 \\
-w_{N-1} + 2w_N + h^2 f\left(x_N, w_N, \frac{\beta - w_{N-1}}{2h}\right) - \beta &= 0
\end{aligned} \tag{5}$$

Step 5: We can take F(w), and implement Newton's method to approximate the solution to this system. We can do this by taking an initial approximation $w^{(0)} = (w_1^{(0)}, w_2^{(0)}, ..., w_N^{(0)})^t$, $F(w^{(0)})$ and defining the Jacobian matrix as follows:

$$\begin{aligned}
J(w_1, w_2, ..., w_N)_{ij} &= -1 + \frac{h}{2}f_{y'}\left(x_i, w_i\frac{w_{i+1} - w_{i-1}}{2h}\right), \ for\ i = j-1\ and\ j = 2, ..., N \\
J(w_1, w_2, ..., w_N)_{ij} &= 2 + h^2 f_y\left(x_i, w_i\frac{w_{i+1} - w_{i-1}}{2h}\right), \ for\ i = j\ and\ j = 1, ..., N \\
J(w_1, w_2, ..., w_N)_{ij} &= -1 - \frac{h}{2}f_{y'}\left(x_i, w_i\frac{w_{i+1} - w_{i-1}}{2h}\right), \ for\ i = j+1\ and\ j = 1, ..., N-1
\end{aligned} \tag{6}$$

where $w_0 = \alpha$ and $w_{N+1} = \beta$.

Remark (7.3) [8] We can find the initial approximation $w^{(0)}$ by using the following equation

$$\mathbf{w}^{(0)} = \alpha + \frac{\beta - \alpha}{b - a}(x_i - a)$$

where $x_i = a + ih$ for i = 1, 2, ..., N.

In the Finite-Difference method, $J(w_1, w_2, ..., w_N)$ is tridiagonal with ijth entry. This means that there are non-zero entries on the main diagonal, non-zero entries on the diagonal directly below the main diagonal, and there are non-

zero entries on the diagonal directly above the main diagonal. If we look at Step 3 of Newton's method, we solve the system $J(x)y = -F(x)$. Now for the Finite Difference method we solve a similar system that is $J(w_1, ..., w_N)(v_1, ...,v_n)^t = -F(w_1, w_2, ..., w_N)$, where
$w_i^{(k)} = w_i^{(k-1)} + v_i$, for each i = 1, 2, ..., N. However, we do not use Gaussian Elimination to solve this system. Since the Jacobian matrix is tridiagonal, we can solve it using Crout LU factorization for matrices such that J(w) = LU.
Crout LU Factorization (7.4) [8] Since J(w) is tridiagonal, it takes on the form:

$$J(\mathbf{w}) = \begin{bmatrix} a_{11} & a_{12} & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ a_{21} & a_{22} & a_{23} & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & a_{32} & a_{33} & a_{34} & 0 & \cdots & \cdots & 0 \\ \vdots & 0 & \ddots & \ddots & \ddots & 0 & \cdots & 0 \\ \vdots & \vdots & 0 & \ddots & \ddots & \ddots & 0 & 0 \\ \vdots & \vdots & \vdots & 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & \vdots & \vdots & 0 & \ddots & \ddots & a_{i-1,j} \\ 0 & 0 & 0 & 0 & 0 & 0 & a_{i,j-1} & a_{ij} \end{bmatrix}.$$

Crout's LU Factorization factors the matrix above into two triangular matrices L and U. These two matrices can be found in the form:

$$L = \begin{bmatrix} l_{11} & 0 & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ l_{21} & l_{22} & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & l_{32} & l_{33} & 0 & \cdots & \cdots & \cdots & 0 \\ \vdots & 0 & \ddots & \ddots & \ddots & 0 & \cdots & 0 \\ \vdots & \vdots & 0 & \ddots & \ddots & \ddots & 0 & 0 \\ \vdots & \vdots & \vdots & 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & \vdots & \vdots & 0 & \ddots & \ddots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & l_{i,j-1} & l_{ij} \end{bmatrix}.$$

and

$$U = \begin{bmatrix} 1 & u_{12} & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & 1 & u_{23} & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & 0 & 1 & u_{34} & 0 & \cdots & \cdots & 0 \\ \vdots & 0 & \ddots & \ddots & \ddots & 0 & \cdots & 0 \\ \vdots & \vdots & 0 & \ddots & \ddots & \ddots & 0 & 0 \\ \vdots & \vdots & \vdots & 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & \vdots & \vdots & 0 & \ddots & \ddots & u_{i-1,j} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Once we have expressed our original matrix J(w) in terms of L and U, we need to compute the entries of each of these matrices. This procedure involves:

i. Computing the first column of L, where $l_{i1} = a_{i1}$
ii. Computing the first row of U, where $u_{1j} = \frac{a_{1j}}{l_{11}}$
iii. Alternately computing the columns of L and the rows of U, were

$$l_{ij} = a_{ij} - \sum_{k=1}^{j-1} l_{ik}u_{kj}, \ \ for\ j \leq i, \ \ i = 1, 2, ..., N$$

$$u_{ij} = \frac{a_{ij} - \sum_{k=1}^{i-1} l_{ik}u_{kj}}{l_{ii}}, \ \ for\ i \leq j, \ \ j = 2, 3, ..., N$$

Once the entries of the LU matrices are determined, we want to solve the system $J(w_1, ..., w_N)(v_1, ..., v_n)^t = -F(w_1, w_2, ..., w_N)$.
We solve this system using the following procedure:

(i) Set up and solve the system $Lz = F(w^{(k)})$, where $z \in R^n$.

(ii) Set up and solve the system Uv = z. (Remember $v = (v_1,...,v_n)^t$)

Once we are able to obtain **v**, we can proceed with computing, $w_i^{(k)} = \mathbf{w}_i^{(k-1)} + \mathbf{v}_i$, and thus repeating Newton's method for the next iteration. As a result, once we can obtain the initial approximation $w^{(0)}$ and form a N × N system, we can follow the iterative process for Newton's method, with the addition of Crout's LU factorization in place of the Gaussian Elimination, to solve the boundary-value problem, i.e. the values of $y(x_i)$, where $x_i = a + ih$ and i=0, 1, 2, ..., N+1. This implies that the procedure for the Finite-Difference method consists of converting the boundary-value problem into a nonlinear algebraic system. Once a nonlinear algebraic system is formulated, we can use Newton's method to solve this system.

VIII. Results:

For solving F(x) = 0, where $F : \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable near a solution p:

i. Newton's method converges quadratically to p if J(p) is nonsingular and the initial guess is sufficiently close.

ii. Broyden's method converges superlinearly to p under similar conditions with $A_0 = J(x^{(0)})$ nonsingular.

For a nonlinear BVP $y'' = f(x, y, y')$ satisfying Theorem (7.2), the Finite-Difference method yields approximations $w_i \to y(x_i)$ as $h \to 0$, with a tridiagonal Jacobian efficiently solved via Crout LU factorization.

IX. Conclusions

This paper provided a comparative overview of Newton's method, Broyden's method, and the Finite-Difference method for solving nonlinear problems. Newton's method offers rapid quadratic convergence but requires costly Jacobian computations. Broyden's method reduces this cost via Jacobian approximation at the expense of slower superlinear convergence. The Finite-Difference method transforms nonlinear BVPs into algebraic systems solvable by Newton's method, with Crout LU factorization exploiting the resulting tridiagonal structure. The choice of method depends on problem size, accuracy demands, and computational resources.

References

[1]. Burden, R. L., & Faires, J. D. (2011). *Numerical Analysis* (9th ed.). Brooks/Cole.
[2]. Dennis, J. E., & Schnabel, R. B. (1996). *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. SIAM.
[3]. Atkinson, K. E. (1989). *An Introduction to Numerical Analysis* (2nd ed.). John Wiley & Sons.
[4]. Heath, M. T. (2018). *Scientific Computing: An Introductory Survey* (2nd ed.). SIAM.
[5]. Ascher, U. M., & Petzold, L. R. (1998). *Computer Methods for ODEs and DAEs*. SIAM.
[6]. Kelley, C. T. (1995). *Iterative Methods for Linear and Nonlinear Equations*. SIAM.
[7]. Quarteroni, A., Sacco, R., & Saleri, F. (2007). *Numerical Mathematics* (2nd ed.). Springer.
[8]. Stoer, J., & Bulirsch, R. (2002). *Introduction to Numerical Analysis* (3rd ed.). Springer.