UNIVERSITA DELL'INSUBRIA

Doctoral Thesis

Equations and Systems of Nonlinear Equations: from high order Numerical Methods to fast Eigensolvers for Structured Matrices and Applications

Author:
Fayyaz Ahmad

Supervisor:
Prof. Dr. Stefano Serra-Capizzano

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

in the

DIPARTIMENTO DI SCIENZA E ALTA TECNOLOGIA

December 2018

UNIVERSITA DELL'INSUBRIA

Doctoral Thesis

Equations and Systems of Nonlinear Equations: from high order Numerical Methods to fast Eigensolvers for Structured Matrices and Applications

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

in the

DIPARTIMENTO DI SCIENZA E ALTA TECNOLOGIA

Fayyaz Ahmad

Supervisor: Prof. Dr. Stefano Serra-Capizzano
Signed: ....

Coordinator: Prof. Dr. Marco Donatelli
Signed: ....

December 2018

UNIVERSITA DELL'INSUBRIA

Abstract

DIPARTIMENTO DI SCIENZA E ALTA TECNOLOGIA

Doctor of Philosophy

Equations and Systems of Nonlinear Equations: from high order Numerical Methods to fast Eigensolvers for Structured Matrices and Applications

by Fayyaz Ahmad

The narrative of our thesis concerns the construction of numerical methods for the solution of nonlinear equations and systems of nonlinear equations, in order to compute simple roots or roots with multiplicities and in several context, ranging from differential problems to the eigenvalues of structured matrices.

After an introduction containing the description and a brief history of the problems with pointers to the relevant literature, we present eight chapters containing our new findings.

First, a parametrized multi-step Newton method is constructed for widening the region of convergence of classical multi-step Newton method. The second improvement is proposed in the context of multi-step Newton methods, by introducing preconditioners to enhance their accuracy, without disturbing their original order of convergence and the related computational cost (in most of the cases).

In this proposal, we cover both cases of multi-step Newton methods with and without derivatives aimed at computing simple roots. To find roots with unknown multiplicities preconditioners are also effective when they are applied to the Newton method for roots with unknown multiplicities. Frozen Jacobian higher order multi-step iterative method for the solution of systems of nonlinear equations are developed and the related results better than those obtained when employing the classical frozen Jacobian multi-step Newton method.

To get benefit from the past information that is produced by the iterative method, we constructed iterative methods with memory for solving systems of nonlinear equations. Iterative methods with memory have a greater rate of convergence, if compared with the iterative method without memory. In terms of computational cost, iterative methods with memory are marginally superior comparatively. Numerical methods are also introduced for approximating all the eigenvalues of banded symmetric Toeplitz and preconditioned Toeplitz matrices.

Our proposed numerical methods work very efficiently, when the generating symbols of the considered Toeplitz matrices are bijective: the remarkable feature of the presented methods is the computational cost which is negligible since we do not even need to store the considered matrices, thanks to specific extrapolation techniques.

We finally stress that all the algorithms presented in the current thesis are accompanied by numerical tests taking into account different applications.

Acknowledgements

> Recite in the name of your Lord who created. Created man from a clinging substance. Recite, and your Lord is the most Generous. Who taught by the pen. Taught man that which he knew not. No! [But] indeed, man transgresses. Because he sees himself self-sufficient. Indeed, to your Lord is the return. Have you seen the one who forbids. A servant when he prays? Have you seen if he is upon guidance Or enjoins righteousness? Have you seen if he denies and turns away. Does he not know that Allah sees? No! If he does not desist, We will surely drag him by the forelock. A lying, sinning forelock. Then let him call his associates; We will call the angels of Hell. No! Do not obey him. But prostrate and draw near [to Allah].

I would like to thank my supervisor, Prof. Dr. Stefano Serra Capizzano, for the patient guidance, encouragement and advice he has provided throughout my time as his student. I have been extremely lucky to have a supervisor who cared so much about my work, and who responded to my questions and queries so promptly.

Special thanks to Prof. Dr. Eman Salem Alaidarous for her moral support and scholarship to support my studies. My friend Dr. Malik Zaka Ullah always helps and supports me in my good and bad times. My younger brother Dr. Shamshad Ahmad who always stands behind me and gives the confidence to chase my aims in life.

I would like to thank my fellow doctoral student Isabella Furci for her cooperation. In addition, I would like to express my gratitude to the staff.

Last but not the least, I would like to thank my family: my parents, to my brothers, sisters, wife and son for supporting me spiritually throughout my course of study and my life in general.

Contents

List of Figures

List of Tables

To my late mother Sakina Bibi.

Chapter 1

Introduction

Most of the phenomena in nature are nonlinear. The mathematical models that represent them are nonlinear models. Some of the nonlinear models can be expressed directly in terms of nonlinear equations or systems of nonlinear equations. In most of the cases the models lead to nonlinear differential or integrodifferential equations or systems of equations: when this scenario occurs we end up to large systems of nonlinear equations after a proper numerical approximation. In general, it is hard to find an analytical solution in closed form for a nonlinear equation and systems of nonlinear equations. The solution of the general quintic equation cannot be expressed algebraically which is demonstrated by Abel's Theorem [12]. In such a scenario numerical iterative methods play an important role to find numerical solutions of nonlinear equations or systems of nonlinear equations. A wide class of nonlinear boundary value problems (BVPs) can be written as

$$\mathcal{L}(u) + f(u) = p(x) \quad \text{under some boundary conditions}\,, \tag{1.1}$$

where $\mathcal{L}$ is a linear differential operator, $f(\cdot)$ is a nonlinear function and $p(x)$ is a source term. A suitable discretization of (1.1) leads to

$$L(\boldsymbol{u}) + f(\boldsymbol{u}) = p(\mathbf{x})\,, \tag{1.2}$$

where L is a discrete approximation of linear differential operator $\mathcal{L}$ and $\boldsymbol{u}$, $\mathbf{x}$ are vectors of appropriate dimensions. Equation (1.2) represents compactly a system of nonlinear equations. This system of nonlinear equations has a linear part $L(\boldsymbol{u})$ and a nonlinear part $f(\boldsymbol{u})$. Due to this form, it can be considered a system of mildly nonlinear equations. Solving this system of nonlinear equations (1.2) means that we are indirectly solving a suitable nonlinear boundary value problem. In the theory of iterative methods to solve nonlinear equations Ostrowski [2] and Traub [1] contributed in a significant manner. Iterative methods can be divided into two classes namely the single-point method and multi-point [1]. Other classifications are

- single-point iterative methods with- and without-memory
- multi-point iterative methods with- and without-memory.

Higher order ($\geq$ 3) single-point iterative methods often suffer from numerical stability. While multi-point iterative methods give the higher order of convergence and numerical stability. Some one-point iterative methods without-memory of different orders, are:

$$\begin{cases} x_{n+1} = x_n - p_1 & \text{(2nd order Newton-Raphson)} \\ x_{n+1} = x_n - p_1 - p_2 & \text{(3rd order)} \\ x_{n+1} = x_n - p_1 - p_2 - p_3 & \text{(4th order)} \\ x_{n+1} = x_n - p_1 - p_2 - p_3 - p_4 & \text{(5th order)} \\ x_{n+1} = x_n - p_1 - p_2 - p_3 - p_4 - p_5 & \text{(6th order),} \end{cases} \tag{1.3}$$

where $p_1 = f(x_n)/f'(x_n)$, $p_2 = c_2 p_1^2$, $p_3 = (-c_3 + 2c_2^2)p_1^3$, $p_4 = (-5c_2c_3 + 5c_2^3 + c_4)p_1^4$, $p_5 = (-c_5 + 3c_3^2 + 14c_2^4 - 21c_3c_2^2 + 6c_2c_4)p_1^5$, $c_2 = f''(x_n)/2!f'(x_n)$, $c_3 = f^{(3)}(x_n)/3!f'(x_n)$, $c_4 = f^{(4)}(x_n)/4!f'(x_n)$, $c_5 = f^{(5)}(x_n)/5!f'(x_n)$, and $f(x) = 0$ is a nonlinear equation.

According to Kung-Traub conjecture [1] if a multi-point iterative method without memory uses n function evaluations its order of convergence is bounded from above by 2^{n-1}. Iterative methods achieving 2^{n-1} convergence order with n function evaluations are called optimal iterative methods. The classical iterative methods due to Newton and Ostrowski are examples of optimal iterative methods for finding a simple root of a nonlinear equation. The efficiency of an the iterative method depends on

- the order of convergence,
- the computational cost,

- the region of convergence.

The efficiency index of the considered iterative method to find the solution of nonlinear equations can be defined as

$$\text{Efficiency index} = \rho^{1/n},$$

where ρ is the convergence order of the iterative method and n is the total number of function evaluations. Usually, the factor of the computational cost of binary operations is not included in the efficiency index in the case of nonlinear equations. But computational cost becomes important when we deal with systems of nonlinear equations. The convergence radius of a given iterative method can be observed by plotting the dynamics in the complex plane [13–30].

The well-known Newton-Raphson iterative method for the system of nonlinear equations can be expressed as

$$\begin{cases} \mathbf{F}'(\mathbf{x}_n)\boldsymbol{\phi}_1 = \mathbf{F}(\mathbf{x}_n), \\ \mathbf{x}_{n+1} = \mathbf{x}_n - \boldsymbol{\phi}_1, \end{cases} \tag{1.4}$$

where $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ is the system of nonlinear equations. The computational cost of a single step of the Newton method is given by the total number of function evaluations that is $n^2 + n$ and the solution of a system of linear equations. The considered iteration has quadratic convergence order. The fourth-order Jarratt [31] iterative scheme can be written as

$$\begin{cases} \mathbf{F}'(\mathbf{x}_n)\boldsymbol{\phi}_1 = \mathbf{F}(\mathbf{x}_n), \\ \mathbf{x}_n = \mathbf{x}_n - \frac{2}{3}\boldsymbol{\phi}_1, \\ (3\mathbf{F}'(\mathbf{x}_n) - \mathbf{F}'(\mathbf{x}_n))\boldsymbol{\phi}_2 = 3\mathbf{F}'(\mathbf{x}_n) + \mathbf{F}(\mathbf{x}_n), \\ \mathbf{x}_{n+1} = \mathbf{x}_n - \frac{1}{2}\boldsymbol{\phi}_2\boldsymbol{\phi}_1. \end{cases} \tag{1.5}$$

H. Montazeri et al. [102] proposed an efficient iterative method with four convergence-order that is

$$\begin{cases} \mathbf{F}'(\mathbf{x}_n)\boldsymbol{\phi}_1 = \mathbf{F}(\mathbf{x}_n), \\ \mathbf{x}_n = \mathbf{x}_n - \frac{2}{3}\boldsymbol{\phi}_1, \\ \mathbf{F}'(\mathbf{x}_n)\mathbf{T} = \mathbf{F}'(\mathbf{x}_n), \\ \boldsymbol{\phi}_2 = \mathbf{T}\boldsymbol{\phi}_1, \\ \boldsymbol{\phi}_3 = \mathbf{T}\boldsymbol{\phi}_2, \\ \mathbf{x}_{n+1} = \mathbf{x}_n - \frac{23}{8}\boldsymbol{\phi}_1 + 3\boldsymbol{\phi}_2 - \frac{9}{8}\boldsymbol{\phi}_3. \end{cases} \tag{1.6}$$

A huge source of information regarding iterative methods for solving systems of nonlinear equation can be found in [2, 32–37, 39, 40, 40, 51, 52, 56, 57, 62–65, 70, 71] and references therein.

Multi-step iterative methods are fascinating because of their low computational cost. The classical Newton multi-step method (NR) can be written as

$$
\mathrm{NR} = \begin{cases}
\text{Number of steps} & = m \geq 1 \\
\text{CO} & = m+1 \\
\text{Function evaluations} & = m \\
\text{Jacobian evaluations} & = 1 \\
\text{Number of LU-factors} & = 1 \\
\text{Vector-vector multiplications} & = 0 \\
\text{Number of solutions of lower} & \\
\text{ and upper triangular systems} & = m
\end{cases}
\begin{cases}
\text{Base method} \rightarrow \begin{cases} \mathbf{F}'(\mathbf{x}_0)\,\boldsymbol{\phi}_1 = \mathbf{F}(\mathbf{x}_0) \\ \mathbf{x}_1 = \mathbf{x}_0 - \boldsymbol{\phi}_1 \end{cases} \\
\text{Multi-step part} \rightarrow \begin{cases} \text{for } s = 1, m-1 \\ \qquad \mathbf{F}'(\mathbf{x}_0)\,\boldsymbol{\phi}_{s+1} = \mathbf{F}(\mathbf{x}_s) \\ \qquad \mathbf{x}_{s+1} = \mathbf{x}_s - \boldsymbol{\phi}_{s+1} \\ \ \text{end} \end{cases}
\end{cases},
$$

Multi-step iterative methods can be divided into two parts, i.e., base method and multi-step part. In the multi-step part we use the information contained in the frozen Jacobian that was computed in the base method. We use the LU-factors of the frozen Jacobian over and over to solve the system of linear equations in the multi-step part, which makes the method computationally economical. In the NR method there is an increment of one in the convergence order per multi-step. Similarly, a derivative-free Newton multi-step method can be expressed as

$$
\begin{array}{l}
\text{Base method} \longrightarrow \begin{cases} \mathbf{x}_0 = \text{initial guess} \\ \boldsymbol{u} = \mathbf{x}_0 + \beta\,\mathbf{F}(\mathbf{x}_0) \\ [\boldsymbol{u}, \mathbf{x}_0; \mathbf{F}]\,\boldsymbol{\phi}_1 = \mathbf{F}\left(\mathbf{x}_0\right) \\ \mathbf{x}_1 = \mathbf{x}_0 - \boldsymbol{\phi}_1 \end{cases} \\
\text{Multi-step part} \rightarrow \begin{cases} \text{for } \ j = 2, m \\ \qquad [\boldsymbol{u}, \mathbf{x}_0; \mathbf{F}]\,\boldsymbol{\phi}_j = \mathbf{F}\left(\mathbf{x}_{j-1}\right) \\ \qquad \mathbf{x}_j = \mathbf{x}_{j-1} - \boldsymbol{\phi}_j \\ \text{end} \\ \mathbf{x}_0 = \mathbf{x}_m \end{cases}
\end{array}, \tag{1.7}
$$

where β is a scalar parameter and the $[.,.;\mathbf{F}] : D \times D \subset \mathbb{R}^n \times \mathbb{R}^n \longrightarrow L(\mathbb{R}^n)$ is divided difference operator of $\mathbf{F}$. We remind that the divided difference operator is defined as

$$[\mathbf{x}+\mathbf{h},\mathbf{x};\mathbf{F}] = \int_0^1 \mathbf{F}'(\mathbf{x}+t\,\mathbf{h})\,dt, \quad \forall \mathbf{x},\mathbf{h}\in\mathbb{R}^n = \mathbf{F}'(\mathbf{x}) + \frac{1}{2}\mathbf{F}''(\mathbf{x})\,\mathbf{h} + \frac{1}{6}\,\mathbf{F}'''(\mathbf{x})\,\mathbf{h}^2 + O(\mathbf{h}^3), \tag{1.8}$$

where $\mathbf{h}^i = \overbrace{(\mathbf{h},\mathbf{h},\cdots,\mathbf{h})}^{i\,times}$. The convergence order of (1.7) is $m+1$ and m is the number of multi-steps. This derivative-free method (1.7) is computationally more expensive compared with the NR method because the number of function evaluations in divided difference operator are more compare to that of Jacobian. There are higher order multi-step methods. Taking note of the convergence order and of the computational cost, multi-step higher order method (HJ) [63, 102] can be described as

$$\text{HJ} = \left\{\begin{array}{ll} \text{Number of steps} & = m \geq 2 \\ \text{Convergence order} & = 2m \\ \text{Function evaluations} & = m-1 \\ \text{Jacobian evaluations} & = 2 \\ \text{LU decomposition} & = 1 \\ \text{Matrix vector multiplications} & = m \\ \text{Vector vector multiplications} & = 2m \\ \text{Number of solutions of systems} & \\ \text{of lower and upper triangular} & \\ \text{systems of equations} & = 2m-1 \end{array}\right\} \begin{array}{ll} \text{Base method} \longrightarrow & \left\{\begin{array}{l} \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_1 = \mathbf{F}(\mathbf{x}_k) \\ \mathbf{x}_1 = \mathbf{x}_k - \frac{2}{3}\,\boldsymbol{\phi}_1 \\ \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_2 = \mathbf{F}'(\mathbf{x}_1)\,\boldsymbol{\phi}_1 \\ \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_3 = \mathbf{F}'(\mathbf{x}_1)\,\boldsymbol{\phi}_2 \\ \mathbf{x}_2 = \mathbf{x}_k - \frac{23}{8}\,\boldsymbol{\phi}_1 + 3\,\boldsymbol{\phi}_2 - \frac{9}{8}\,\boldsymbol{\phi}_3 \end{array}\right. \\ \text{Multi-step part} \rightarrow & \left\{\begin{array}{l} \text{for } s = 1, m-2 \\ \quad \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_{2s+2} = \mathbf{F}(\mathbf{x}_{s+1}) \\ \quad \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_{2s+3} = \mathbf{F}'(\mathbf{x}_1)\,\boldsymbol{\phi}_{2s+2} \\ \quad \mathbf{x}_{s+2} = \mathbf{x}_{s+1} - \frac{5}{2}\,\boldsymbol{\phi}_{2s+2} + \frac{3}{2}\,\boldsymbol{\phi}_{2s+3} \\ \text{end} \end{array}\right. \end{array}$$

The base method in a HJ method has a convergence order of four and involves one LU decomposition, one function evaluation, and two Jacobian evaluations. Each step of the multi-step part increases the convergence order by two. In 2015, Malik et al. [70] developed another efficient multi-step iterative method (MSF) for solving nonlinear systems arising from particular ODEs which can be described as

$$\text{MSF} = \left\{\begin{array}{ll} \text{Number of steps} & = m \\ \text{Convergence-order} & = 3m \\ \text{Function evaluations} & = m \\ \text{Jacobian evaluations} & = 2 \\ \text{Second-order Fréchet derivative} & = 1 \\ \text{LU decomposition} & = 1 \\ \text{Matrix vector multiplications} & = 2m-2 \\ \text{Vector vector multiplications} & = m+2 \\ \text{Number of solutions of systems} & \\ \text{of lower and upper triangular} & \\ \text{systems of equations} & = 3m-1 \end{array}\right\} \begin{array}{ll} \text{Base method} \longrightarrow & \left\{\begin{array}{l} \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_1 = \mathbf{F}(\mathbf{x}_k) \\ \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_2 = \mathbf{F}''(\mathbf{x}_k)\,\boldsymbol{\phi}_1^2 \\ \mathbf{x}_1 = \mathbf{x}_k - \boldsymbol{\phi}_1 - \frac{1}{2}\,\boldsymbol{\phi}_2 \end{array}\right. \\ \text{Multi-step part} \rightarrow & \left\{\begin{array}{l} \text{for } s = 1, m-1 \\ \quad \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_{3s} = \mathbf{F}(\mathbf{x}_s) \\ \quad \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_{3s+1} = \mathbf{F}'(\mathbf{x}_1)\,\boldsymbol{\phi}_{3s} \\ \quad \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_{3s+2} = \mathbf{F}'(\mathbf{x}_1)\,\boldsymbol{\phi}_{3s+1} \\ \quad \mathbf{x}_{s+1} = \mathbf{x}_s - 3\,\boldsymbol{\phi}_{3s} + 3\,\boldsymbol{\phi}_{3s+1} \\ \qquad\quad -\boldsymbol{\phi}_{3s+2} \\ \text{end} \end{array}\right. \end{array}$$

Further improvements in multi-step method are possible in the direction of reusing the available information. There are a few works related to iterative methods with memory of high orders for nonlinear systems. Researchers have introduced iterative methods with memory to solve systems of nonlinear equations. In 2014, Sharma et al. [51] proposed a derivative-free iterative method with fourth order of convergence. In detail we have

$$\begin{cases} \mathbf{y}_k = \mathbf{x}_k - [\mathbf{x}_k, \mathbf{w}_k, \mathbf{F}]^{-1}\mathbf{F}(\mathbf{x}_k), \\ \mathbf{x}_{k+1} = \mathbf{y}_k - (a\mathbf{I} + \mathbf{G}^{(k)}((3-2a)\mathbf{I} + (a-2)\mathbf{G}^{(k)}))[\mathbf{x}_k, \mathbf{w}_k, \mathbf{F}]^{-1}\mathbf{F}(\mathbf{x}_k), \end{cases} \tag{1.9}$$

where $\mathbf{w}_k = (\mathbf{x}_k + b\,\mathbf{F}(\mathbf{x}_k)$, $\mathbf{z}_k = \mathbf{y}_k + c\,\mathbf{F}(\mathbf{x}_k)$, $a \in \mathbb{R}$, $b, c \in \mathbb{R}\backslash\{0\}$, $\mathbf{G}^{(k)} = [\mathbf{x}_k, \mathbf{w}_k, \mathbf{F}]^{-1}[\mathbf{z}_k, \mathbf{y}_k, \mathbf{F}]$, and $\mathbf{I}$ is the identity matrix of appropriate size.

Recently, the authors in [48] improved the convergence speed of (1.9) to $2 + \sqrt{5}$ when $a \neq 3$ and $2 + \sqrt{6}$, when $a = 3$, by considering

$$\begin{cases} \mathbf{B}^{(k)} = -[\mathbf{w}_{k-1}, \mathbf{x}_{k-1}; \mathbf{F}]^{-1}, & k \geq 1, \\ \mathbf{y}_k = \mathbf{x}_k - [\mathbf{x}_k, \mathbf{w}_k, \mathbf{F}]^{-1}\mathbf{F}(\mathbf{x}_k), & k \geq 0, \\ \mathbf{x}_{k+1} = \mathbf{y}_k - \\ \qquad (a\mathbf{I} + \mathbf{G}^{(k)}((3-2a)\mathbf{I} + (a-2)\mathbf{G}^{(k)}))[\mathbf{x}_k, \mathbf{w}_k, \mathbf{F}]^{-1}\mathbf{F}(\mathbf{x}_k), \end{cases} \tag{1.10}$$

where $\mathbf{w}_k = \mathbf{x}_k + \mathbf{B}^{(k)}\mathbf{F}(\mathbf{x}_k)$.

In the direction of preconditioning, Aslam and his co-researcher [112] proposed a preconditioned double Newton method with quartic convergence order for solving a system of nonlinear equations. We now describes shortly the considered algorithmic proposal. Let

$$\mathbf{F}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_n(\mathbf{x})]^T = \mathbf{0} \tag{1.11}$$

be the system of nonlinear equations and let us suppose that only simple roots are present. Here $\mathbf{x} = [x_1, x_2, \cdots, x_n]^T$. Assume $\mathbf{G}(\mathbf{x}) = [g_1(\mathbf{x}), g_2(\mathbf{x}), \cdots, g_n(\mathbf{x})]^T$ is a function which is non-zero everywhere in its definition domain. We define a new function

$$\mathbf{Q}(\mathbf{x}) = \mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}) = [\![\mathbf{G}(\mathbf{x})]\!]\,\mathbf{F}(\mathbf{x}) = [\![\mathbf{F}(\mathbf{x})]\!]\,\mathbf{G}(\mathbf{x}), \tag{1.12}$$

where $\odot$ is the element-wise multiplication and $[\![\cdot]\!]$ represent the diagonal matrix, having as main diagonal its argument. The first order Fréchet derivative of (1.12) can be computed as

$$\begin{aligned}\mathbf{Q}'(\mathbf{x}) &= [\![\mathbf{F}(\mathbf{x})]\!]\,\mathbf{G}'(\mathbf{x}) + [\![\mathbf{G}(\mathbf{x})]\!]\,\mathbf{F}'(\mathbf{x})\\ &= [\![\mathbf{G}(\mathbf{x})]\!]\,\big(\mathbf{F}'(\mathbf{x}) + [\![\mathbf{F}(\mathbf{x})]\!]\,[\![\mathbf{G}(\mathbf{x})]\!]^{-1}\,\mathbf{G}'(\mathbf{x})\big).\end{aligned} \tag{1.13}$$

The application of the Newton method to (1.12) leads to

$$\begin{aligned}\mathbf{x}_{k+1} &= \mathbf{x}_k - \mathbf{Q}'(\mathbf{x}_k)^{-1}\,\mathbf{Q}(\mathbf{x}_k)\\ &= \mathbf{x}_k - \big(\mathbf{F}'(\mathbf{x}_k) + [\![\mathbf{F}(\mathbf{x}_k)]\!]\,[\![\mathbf{G}(\mathbf{x}_k)]\!]^{-1}\,\mathbf{G}'(\mathbf{x}_k)\big)^{-1}\,\mathbf{F}(\mathbf{x}_k).\end{aligned} \tag{1.14}$$

The convergence order of (1.14) is quadratic, because the considered scheme is the Newton method for solving the preconditioned system of nonlinear equations $\mathbf{Q}(\mathbf{x}) = \mathbf{0}$. If we take $\mathbf{G}(\mathbf{x}) = exp(\boldsymbol{\beta} \odot \mathbf{x})$ then (1.14) can be written as

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + [\![\boldsymbol{\beta} \odot \mathbf{F}(\mathbf{x}_k)]\!]\right)^{-1}\,\mathbf{F}(\mathbf{x}_k), \tag{1.15}$$

where $\boldsymbol{\beta} = [\beta_1, \beta_2, \cdots, \beta_n]^T$.

When the system of nonlinear equations have unknown multiplicities, a modified Newton method for finding zeros with multiplicity greater than one for nonlinear equations can be written as

$$\begin{cases} x_0 = \text{initial guess} \\ x_{k+1} = x_k - m\,\frac{\phi(x_k)}{\phi'(x_k)}, \quad k = 0, 1, \cdots . \end{cases} \tag{1.16}$$

Hueso et al. [110] proposed the multidimensional version of of (1.16) as

$$\begin{cases} \mathbf{x}_0 = \text{initial guess} \\ \mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{F}'(\mathbf{x}_k)^{-1}\,[\![\mathbf{m}]\!]\,\mathbf{F}(\mathbf{x}_k), \quad k = 0, 1, \cdots , \end{cases} \tag{1.17}$$

where $\mathbf{m} = [m_1, m_2, \cdots, m_n]^T$ is a vector of multiplicities for the system of nonlinear equations $\mathbf{F}(\mathbf{x}) = \mathbf{0}$. The proof of quadratic convergence of (1.17) is provided in [110]. W. Wu [109] proposed a variant of the Newton method with the help of an auxiliary exponential function used as preconditionor. We define a new system of nonlinear equation with a nonlinear preconditioner function, having the same root

$$\mathbf{U}(\mathbf{x}) = e^{\mathbf{v}\odot\mathbf{x}} \odot \mathbf{F}(\mathbf{x}) = \mathbf{0}\,. \tag{1.18}$$

The application of the Newton method for (1.18) implies

$$\begin{aligned}
\mathbf{x}_{k+1} &= \mathbf{x}_k - \mathbf{U}'(\mathbf{x}_k)^{-1}\,\mathbf{U}(\mathbf{x}_k)\,,\\
\mathbf{x}_{k+1} &= \mathbf{x}_k - \left(\llbracket e^{\mathbf{v}\odot\mathbf{x}_k}\rrbracket \left(\mathbf{F}'(\mathbf{x}_k) + \llbracket \mathbf{v}\odot\mathbf{F}(\mathbf{x}_k)\rrbracket\right)\right)^{-1}\, e^{\mathbf{v}\odot\mathbf{x}_k}\odot\mathbf{F}(\mathbf{x}_k)\,,\\
\mathbf{x}_{k+1} &= \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + \llbracket \mathbf{v}\odot\mathbf{F}(\mathbf{x}_k)\rrbracket\right)^{-1}\,\llbracket e^{\mathbf{v}\odot\mathbf{x}_k}\rrbracket^{-1} e^{\mathbf{v}\odot\mathbf{x}_k}\odot\mathbf{F}(\mathbf{x}_k)\,,\\
\mathbf{x}_{k+1} &= \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + \llbracket \mathbf{v}\odot\mathbf{F}(\mathbf{x}_k)\rrbracket\right)^{-1}\,\mathbf{F}(\mathbf{x}_k).
\end{aligned} \tag{1.19}$$

The rate of convergence of (1.19) is quadratic. A modification [110] in (1.16) is proposed by using an exponential preconditioner

$$\mathbf{U}(\mathbf{x}) = e^{\mathbf{v}\odot\mathbf{x}} \odot \mathbf{F}(\mathbf{x})^{1/\mathbf{m}} = \mathbf{0}, \tag{1.20}$$

where $1/\mathbf{m} = [1/m_1, 1/m_2, \cdots, 1/m_n]^T$ and power of $\mathbf{F}(\mathbf{x})$ is component-wise. The application of the Newton method to (1.20) leads to

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + \llbracket \mathbf{v}\odot\mathbf{F}(\mathbf{x}_k)\rrbracket\right)^{-1}\,\llbracket\mathbf{m}\rrbracket\,\mathbf{F}(\mathbf{x}_k). \tag{1.21}$$

We remind that the original idea of a nonlinear preconditioner function was proposed in [109]. Noor et al. [112] have proposed Newton method with general preconditioners. They defined a preconditioned system of nonlinear equations as follows

$$\mathbf{U}(\mathbf{x}) = \mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}) = \mathbf{0}, \tag{1.22}$$

where $\mathbf{G}(\mathbf{x}) \neq \mathbf{0}$. Notice that the roots of $\mathbf{U}(\mathbf{x}) = \mathbf{0}$ and $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ coincide because $\mathbf{G}(\mathbf{x}) \neq \mathbf{0}$ for all $\mathbf{x}$. The first order Fréchet derivative of (1.22) can be computed as

$$\begin{aligned}
&\Psi_i(\mathbf{x}) = \Phi_i(\mathbf{x})\;\Lambda_i(\mathbf{x}),\\
&\nabla\Psi_i(\mathbf{x})^T = \Phi_i(\mathbf{x})\,\nabla\Lambda_i(\mathbf{x})^T + \Lambda_i(\mathbf{x})\,\nabla\Phi_i(\mathbf{x})^T,\;\; i = 1, 2, \cdots, n,
\end{aligned} \tag{1.23}$$

$$
\begin{aligned}
\begin{bmatrix} \nabla\Psi_1(\mathbf{x})^T \\ \nabla\Psi_2(\mathbf{x})^T \\ \nabla\Psi_3(\mathbf{x})^T \\ \vdots \\ \nabla\Psi_n(\mathbf{x})^T \end{bmatrix}
&= \begin{bmatrix} \Phi_1(\mathbf{x}) & 0 & \cdots & 0 \\ 0 & \Phi_2(\mathbf{x}) & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \Phi_n(\mathbf{x}) \end{bmatrix}
\begin{bmatrix} \nabla\Lambda_1(\mathbf{x})^T \\ \nabla\Lambda_2(\mathbf{x})^T \\ \nabla\Lambda_3(\mathbf{x})^T \\ \vdots \\ \nabla\Lambda_n(\mathbf{x})^T \end{bmatrix} \\
&+ \begin{bmatrix} \Lambda_1(\mathbf{x}) & 0 & \cdots & 0 \\ 0 & \Lambda_2(\mathbf{x}) & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \Lambda_n(\mathbf{x}) \end{bmatrix}
\begin{bmatrix} \nabla\Phi_1(\mathbf{x})^T \\ \nabla\Phi_2(\mathbf{x})^T \\ \nabla\Phi_3(\mathbf{x})^T \\ \vdots \\ \nabla\Phi_n(\mathbf{x})^T \end{bmatrix}.
\end{aligned}
$$

From (1.23), the Fréchet derivative of $\mathbf{F}(\mathbf{x}) \odot \mathbf{G}(\mathbf{x})$ is

$$
\begin{aligned}
&(\mathbf{F}(\mathbf{x}) \odot \mathbf{G}(\mathbf{x}))' = \llbracket \mathbf{F}(\mathbf{x}) \rrbracket \mathbf{G}'(\mathbf{x}) + \llbracket \mathbf{G}(\mathbf{x}) \rrbracket \mathbf{F}'(\mathbf{x}) \,, \\
&\mathbf{U}'(\mathbf{x}) = \llbracket \mathbf{G}(\mathbf{x}) \rrbracket \, \mathbf{F}'(\mathbf{x}) + \llbracket \mathbf{F}(\mathbf{x}) \rrbracket \, \mathbf{G}'(\mathbf{x}) \,, \\
&\mathbf{U}'(\mathbf{x}) = \llbracket \mathbf{G}(\mathbf{x}) \rrbracket \left(\mathbf{F}'(\mathbf{x}) + \llbracket \mathbf{F}(\mathbf{x}) \rrbracket \, \llbracket \mathbf{G}(\mathbf{x}) \rrbracket^{-1} \mathbf{G}'(\mathbf{x}) \right).
\end{aligned} \tag{1.24}
$$

If we apply the Newton method to (1.22), we obtain

$$
\begin{aligned}
\mathbf{x}_{k+1} &= \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + \llbracket \mathbf{F}(\mathbf{x}_k) \rrbracket \, \llbracket \mathbf{G}(\mathbf{x}_k) \rrbracket^{-1} \mathbf{G}'(\mathbf{x}_k) \right)^{-1} \llbracket \mathbf{G}(\mathbf{x}_k) \rrbracket^{-1} \mathbf{G}(\mathbf{x}_k) \odot \mathbf{F}(\mathbf{x}_k) \,, \\
\mathbf{x}_{k+1} &= \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + \llbracket \mathbf{F}(\mathbf{x}_k) \rrbracket \, \llbracket \mathbf{G}(\mathbf{x}_k) \rrbracket^{-1} \mathbf{G}'(\mathbf{x}_k) \right)^{-1} \mathbf{F}(\mathbf{x}_k).
\end{aligned} \tag{1.25}
$$

The convergence order of (1.25) is two. The iterative method (1.21) with a general preconditioner can be written as

$$
\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + \llbracket \mathbf{F}(\mathbf{x}_k) \rrbracket \, \llbracket \mathbf{G}(\mathbf{x}_k) \rrbracket^{-1} \mathbf{G}'(\mathbf{x}_k) \right)^{-1} \llbracket \mathbf{m} \rrbracket \, \mathbf{F}(\mathbf{x}_k). \tag{1.26}
$$

The convergence order of (1.26) is also two. The modified Newton method [1, 7, 8] for solving nonlinear equations with unknown multiplicity can be developed in this way. We define a new function

$$
s(x) = \frac{\phi(x)}{\phi'(x)}. \tag{1.27}
$$

The application of the Newton method to (1.27) gives

$$\begin{aligned} x_{k+1} &= x_k - \frac{s(x_k)}{s'(x_k)}\,, \\ x_{k+1} &= x_k - \frac{\phi'(x_k)\phi(x_k)}{\phi'(x_k)^2 - \phi''(x_k)\,\phi(x_k)}. \end{aligned} \tag{1.28}$$

The order of convergence of (1.28) is two. Noor and his co-researchers [111] have constructed a family of methods for solving nonlinear equations with unknown multiplicity by introducing a preconditioner. They defined a further function

$$q(x) = \frac{\phi(x)\,\lambda(x)}{\phi'(x)} \tag{1.29}$$

and the application of the Newton method to (1.29) leads to

$$\begin{aligned} x_{k+1} &= x_k - \frac{q(x_k)}{q'(x_k)}\,, \\ x_{k+1} &= x_k - \frac{\phi'(x_k)\,\phi(x_k)\,\lambda(x_k)}{\phi'(x_k)(\phi(x_k)\,\lambda(x_k))' - \phi''(x_k)\,\phi(x_k)\,\lambda(x_k)}, \end{aligned} \tag{1.30}$$

where $\lambda(x)$ is a non-zero function. The order of convergence of (1.30) is again two.

Eigenvalues are the roots of characteristics polynomials. Recently Ekström et al. [137] have introduced a numerical method to extrapolate the eigenvalues of small-size structured matrices in order to compute the eigenvalues of much larger structured matrices. Our new contribution goes along the same lines but concerns a further class of matrices with hidden structure, that is a special class of "preconditioned" Toeplitz matrices.

A matrix of size n, having a fixed entry along each diagonal, is called Toeplitz and enjoys the expression

$$[a_{i-j}]_{i,j=1}^{n} = \begin{bmatrix} a_0 & a_{-1} & a_{-2} & \cdots & \cdots & a_{-(n-1)} \\ a_1 & \ddots & \ddots & \ddots & & \vdots \\ a_2 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & a_{-2} \\ \vdots & & \ddots & \ddots & \ddots & a_{-1} \\ a_{n-1} & \cdots & \cdots & a_2 & a_1 & a_0 \end{bmatrix}.$$

Given a complex-valued Lebesgue integrable function $\phi : [-\pi, \pi] \to \mathbb{C}$, the n-th Toeplitz matrix generated by ϕ is defined as

$$T_n(\phi) = \left[\hat{\phi}_{i-j}\right]_{i,j=1}^{n},$$

where the quantities $\hat{\phi}_k$ are the Fourier coefficients of ϕ, which means

$$\hat{\phi}_k = \frac{1}{2\pi} \int_{-\pi}^{\pi} \phi(\theta)\, \mathrm{e}^{-\mathbf{i}k\theta} \mathrm{d}\theta, \qquad k \in \mathbb{Z}.$$

We refer to $\{T_n(\phi)\}_n$ as the Toeplitz sequence generated by ϕ, which in turn is called the generating function of $\{T_n(\phi)\}_n$. In the case where ϕ is real-valued, all the matrices $T_n(\phi)$ are Hermitian and much is known about their spectral properties, from the localization of the eigenvalues to the asymptotic spectral distribution in the Weyl sense: in particular ϕ is the spectral symbol of $\{T_n(\phi)\}_n$, see [9, 11] and the references therein.

More in detail, if ϕ is real-valued and not identically constant, then any eigenvalue of $T_n(\phi)$ belongs to the open set (m_ϕ, M_ϕ), with m_ϕ, M_ϕ being the essential infimum, the essential supremum of ϕ, respectively. The case of a constant ϕ is trivial: in that case if $\phi = m$ almost everywhere then $T_n(\phi) = m\mathbb{I}_n$ with $\mathbb{I}_n$ denoting the identity of size n. Hence if $M_\phi > 0$ and ϕ is nonnegative almost everywhere, then $T_n(\phi)$ is Hermitian positive definite.

In the current work we focus our attention on the following setting.

- We consider two real-valued cosine trigonometric polynomials (RCTPs) f, g, that is

$$f(\theta) = \hat{f}_0 + 2\sum_{k=1}^{m_1} \hat{f}_k \cos(k\theta), \qquad \hat{f}_0, \hat{f}_1, \ldots, \hat{f}_{m_1} \in \mathbb{R}, \qquad m_1 \in \mathbb{N},$$

$$g(\theta) = \hat{g}_0 + 2\sum_{k=1}^{m_2} \hat{g}_k \cos(k\theta), \qquad \hat{g}_0, \hat{g}_1, \ldots, \hat{g}_{m_2} \in \mathbb{R}, \qquad m_2 \in \mathbb{N},$$

  so that $T_n(f)$, $T_n(g)$ are both real symmetric.

- We assume that $M_g = \max g > 0$ and $m_g = \min g \geq 0$, so that $T_n(g)$ is positive definite.

- We consider $\mathcal{P}_n(f, g) = T_n^{-1}(g)T_n(f)$ the "preconditioned" matrix and we define the new symbol $r = f/g$.

The n-th Toeplitz matrix generated by $\phi \in \{f, g\}$ is the real symmetric banded matrix of bandwidth $2m+1$, $m \in \{m_1, m_2\}$ ($m = m_1$ if $\phi = f$ and $m = m_2$ if $\phi = g$), given by

$$T_n(\phi) = \left[\begin{array}{ccccccccccc}
\hat{\phi}_0 & \hat{\phi}_1 & \cdots & \hat{\phi}_m & & & & & & & \\
\hat{\phi}_1 & \ddots & \ddots & & \ddots & & & & & & \\
\vdots & \ddots & \ddots & \ddots & & \ddots & & & & & \\
\hat{\phi}_m & & \ddots & \ddots & \ddots & & \ddots & & & & \\
 & \ddots & & \ddots & \ddots & \ddots & & \ddots & & & \\
 & & \multicolumn{7}{c}{\boxed{\hat{\phi}_m \quad \cdots \quad \hat{\phi}_1 \quad \hat{\phi}_0 \quad \hat{\phi}_1 \quad \cdots \quad \hat{\phi}_m}} & & \\
 & & & \ddots & & \ddots & \ddots & \ddots & & \ddots & \\
 & & & & \ddots & & \ddots & \ddots & \ddots & & \hat{\phi}_m \\
 & & & & & \ddots & & \ddots & \ddots & \ddots & \vdots \\
 & & & & & & \ddots & & \ddots & \ddots & \hat{\phi}_1 \\
 & & & & & & & \hat{\phi}_m & \cdots & \hat{\phi}_1 & \hat{\phi}_0
\end{array}\right].$$

Matrices of the form $\mathcal{P}_n(f, g)$ are important for the fast solution of large Toeplitz linear systems (in connection with the preconditioned conjugate gradient method [134–136, 142] or of more general preconditioned Krylov methods [139, 140]). Furthermore, up to low rank corrections, they appear in the context of the spectral approximation of differential operators in which a low rank correction of $T_n(g)$ is the mass matrix and a low rank correction of $T_n(f)$ is the stiffness matrix.

Their spectral features have been studied in detail. More precisely, under the assumption that $r = m$ identically $\mathcal{P}_n(f, g) = r\mathbb{I}_n$, while if $m_r < M_r$, then any eigenvalue of $\mathcal{P}_n(f, g)$ belongs to the open set (m_r, M_r), see [136], and the whole sequence $\{\mathcal{P}_n(f, g)\}_n$ is spectrally distributed in the Weyl sense as $r = f/g$ (see [143]).

In our context, we say that a function is monotone if it is either increasing or decreasing over the interval $[0, \pi]$.

Under the assumption that $r = f/g$ is monotone, we show experimentally that for every integer $\alpha \geq 0$, every n and every $j = 1, \ldots, n$, the following asymptotic expansion holds:

$$\lambda_j(\mathcal{P}_n(f,g)) = r(\theta_{j,n}) + \sum_{k=1}^{\alpha} c_k(\theta_{j,n})h^k + E_{j,n,\alpha}, \tag{1.31}$$

where:

- the eigenvalues of $\mathcal{P}_n(f,g)$ are arranged in nondecreasing or nonincreasing order, depending on whether r is increasing or decreasing;
- $\{c_k\}_{k=1,2,\ldots}$ is a sequence of functions from $[0,\pi]$ to $\mathbb{R}$ which depends only on r;
- $h = \frac{1}{n+1}$ and $\theta_{j,n} = \frac{j\pi}{n+1} = j\pi h$;
- $E_{j,n,\alpha} = O(h^{\alpha+1})$ is the remainder (the error), which satisfies the inequality $|E_{j,n,\alpha}| \leq C_\alpha h^{\alpha+1}$ for some constant C_α depending only on α and r.

In the pure Toeplitz case, that is for $g = 1$ identically, so that $\mathcal{P}_n(f,g) = T_n(f)$ and $r = f$, the result is proven in [131–133], if the RCTP f is monotone and satisfies certain additional assumptions, which include the requirements that $f'(\theta) \neq 0$ for $\theta \in (0,\pi)$ and $f''(\theta) \neq 0$ for $\theta \in \{0,\pi\}$. The symbols

$$f_q(\theta) = (2 - 2\cos\theta)^q, \qquad q = 1, 2, \ldots, \tag{1.32}$$

arise in the discretization of differential equations and are therefore of particular interest. Unfortunately, for these symbols the requirement that $f''(0) \neq 0$ is not satisfied if $q \geq 2$. In [138] several numerical evidences are reported, showing that the higher order approximation (8.1) holds even in this “degenerate case”.

Furthermore, in [138], the authors employed the asymptotic expansion (8.1) for computing an accurate approximation of $\lambda_j(T_n(f))$ for very large n, provided that the values $\lambda_{j_1}(T_{n_1}(f)), \ldots, \lambda_{j_s}(T_{n_s}(f))$ are available for moderate sizes $n_1, \ldots, n_s$ with $\theta_{j_1,n_1} = \cdots = \theta_{j_s,n_s} = \theta_{j,n}$, $s \geq 2$.

Overview of thesis

The first chapter encloses the introduction. The second chapter covers the construction of a parametric multi-step Newton method. Parametric multi-step Newton method is the generalization of the classical Newton method to widen the region of convergence. Widening the convergence regions gives a richer choice for setting the initial guess. The inclusion of multi-steps with help of frozen Jacobian provide a computationally economical iterative method. A set of nonlinear boundary value problems is solved by using parametric multi-step Newton method. Results show that the use of a parameter gives high accuracy and convergence when the classical Newton method fails. The effect of preconditioners on frozen multi-step Newton method is explored in the third chapter. The preconditioners are introduced in a way that they do not affect the order of convergence of the frozen Jacobian multi-step Newton method. The computational cost of both methods is almost the same. The simple roots of original systems of nonlinear equations are not affected by preconditioners, because in the domain of definition of the considered systems of nonlinear equations, the preconditioners do not have any root. Many examples are solved numerically in order to show the influence of preconditioners. The fourth chapter is concerned with the study of preconditioners for derivative-free frozen Jacobian multi-step iterative methods, when solving a system of nonlinear equations in the case of simple roots. In this study, we found that the considered preconditioners make the method more efficient in terms of accuracy. The computational cost is almost the same, when sparse preconditioners are used. The numerical examples clearly exhibit the benefits of preconditioners. All the above methods are constructed for the computation of simple roots i.e., roots with multiplicity one. The preconditioners are also equally effective when we are solving a system of nonlinear equations with unknown multiplicities. The design of preconditioners guarantees a similar computational cost and computation of roots with multiplicities of the original system of nonlinear equations. This topic is covered in the fifth chapter. Chapter six presents the construction of higher order frozen Jacobian multi-step methods for solving a system of nonlinear equations for finding simple roots. The order of convergence of this method is $3m - 4$ and here $m(\geq 2)$ is the number of steps. The inversion of Jacobian is not performed directly, instead we compute the LU-factors and we use them in the multi-step part, to solve the system of linear equations over and over for making the method computationally economical. Different types of nonlinear boundary value problems are solved with the help of our proposed method to highlight related efficiencys. Iterative methods for solving a system of nonlinear equations produce a sequence of approximations to the roots. To collect benefit from the previous information, researchers have developed iterative methods with memory to solve nonlinear equations and systems of nonlinear equations. Chapter seven covers derivative-free iterative methods with memory to find simple roots of systems of nonlinear equations. This chapter also includes the construction and the proof of the convergence order. Dense and mildly nonlinear systems of equations are solved in order to show the efficiency and accuracy of our proposed methods. Chapters eight and nine deal with computations of eigenvalues of structured Toeplitz matrices. Chapter eight contains the expansion of the eigenvalues of preconditioned banded symmetric Toeplitz matrices. With the help of this expansion, we can extrapolate the eigenvalues of the large size of preconditioned banded symmetric Toeplitz matrices, by using the computed eigenvalues of much smaller matrices. This numerical method works well orredwhen the generating symbol is bijective. In non-bijective cases, the efficiency of this method degrades. In chapter nine, a robust numerical method is presented to extrapolate and interpolate the eigenvalues of banded symmetric Toeplitz matrices without any eigenvalue expansion. The numerical method is equally valid for the extrapolation and interpolation of eigenvalues of preconditioned banded symmetric Toeplitz matrices, but we do not cover this topic in this thesis. Again the method is efficient in the bijective part of the generating symbol. Many numerical examples are presented to present the numerical accuracy and stability of numerical methods covered in chapters eight and nine.

The last chapter encloses the summary of the thesis and future work.

Papers:

1. F. Ahmad, E. Tohidi, J. A. Carrasco, A parameterized multi-step Newton method for solving systems of nonlinear equations, Numerical Algorithms, 71(3) (2016) 631–653.

2. F. Ahmad, M. Z. Ullah, S. Ahmad, A. S. Alshomrani, A. M. Alqahtani, L. Alzaben, Multi-step preconditioned Newton methods for solving systems of nonlinear equations, SeMA Journal, 75(1) (2018) 127–137.

3. F. Ahmad, Multi-step derivative-free preconditioned Newton method for solving systems of nonlinear equations, SeMA Journal, 75(1) (2018) 45–56.

4. F. Ahmad. E. Tohidi, M. Z. Ullah, J. A. Carrasco, Higher order multi-step Jarratt-like method for solving systems of nonlinear equations: Application to PDEs and ODEs, Computers & Mathematics with Applications, 70(4) (2015) 624–636.

5. F. Ahmad. F. Soleymani, F. K. Haghani, S. Serra-Capizzano, Higher order derivative-free iterative methods with and without memory for systems of nonlinear equations, Applied Mathematics and Computation, 314(1) (2017) 199–211.

6. F. Ahmad, E. S. Al-Aidarous, D. A. Alrehaili, S.-E. Ekström, I. Furci, S. Serra-Capizzano, Are the eigenvalues of preconditioned banded symmetric Toeplitz matrices known in almost closed form? Numerical Algorithms, 78(2) (2018) 867–893.

The following chapters report in a faithful manner the published papers, with the exception of chapter 9 which contains unpublished findings. In chapters 2-8 only minor language or notation changes occur with respect to the published papers, in order to make the presentation as uniform as possible.

Chapter 2

A Parameterized Multi-step Newton Method for Solving Systems of Nonlinear Equations

We construct a novel family of multi-step iterative methods, for solving systems of nonlinear equations, by introducing a parameter θ to generalize the multi-step Newton method, while keeping its order of convergence and computational cost. By an appropriate selection of θ, the new method can both have faster convergence and have larger radius of convergence. The new iterative method only requires one Jacobian inversion per iteration, and, therefore, can be efficiently implemented using Krylov subspace methods. The new method can be used for solving nonlinear systems of partial differential equations, such as complex generalized Zakharov systems of partial differential equations. The idea is to transform them into systems of nonlinear equations by discretizing approaches in both spatial and temporal dimensions such as, for instance, the Chebyshev pseudo-spectral discretizing method. Quite extensive tests show that the new method can have significantly faster convergence and sensibly larger radius of convergence than that exhibited by the multi-step Newton method.

2.1 Introduction

Numerical methods for solving nonlinear systems of equations are an important research topic. Nonlinear systems of equations usually arise when discretizing ordinary differential equations (ODEs) and partial differential equations (PDEs). The classical Newton-Raphson method [1] is a basic iterative method for solving nonlinear systems of equations. A large number of papers have considered that method and variants. For instance, Cruz et al. [59] have proposed some gradient-free inexact forms of Newton-Raphson. Moreover, An and Bai [60] have discussed a globally convergent iterative scheme using the GMRES method. It should be noted that they assumed that the Jacobian matrix associated with the considered nonlinear system of equations had a sparse form. In all those methods, LU decomposition or an efficient iterative linear system solver such as the NSCGNR algorithm [61] can be used to avoid the calculation of the inverse of the Jacobian matrix.

Since in multi-step methods the inverse of the Jacobian matrix is computed several times, robust iterative schemes such as Krylov subspace methods [62–64] should be considered. For instance, the authors of [65] introduced a class of multi-step iterative methods for solving nonlinear systems of equations which avoid the computation of high order Fréchet derivatives. In summary, multi-step iterative methods are computationally attractive. It should be noted that those iterative methods provide an effective way of constructing highly accurate solutions with low computational cost. As a typical iterative method, one can mention the multi-step variant of Newton method [1, 66]. That variant will be called here NR. The NR method for solving a nonlinear system $\mathbf{F}(\mathbf{x}) = 0$ can be described as

$$
\begin{aligned}
&\text{Base method} \rightarrow \begin{cases} \mathbf{F}'(\mathbf{x}_0)\,\boldsymbol{\phi}_1 = \mathbf{F}(\mathbf{x}_0) \\ \mathbf{x}_1 = \mathbf{x}_0 - \boldsymbol{\phi}_1 \end{cases} \\
&\text{Multi-step part} \rightarrow \begin{cases} \text{for } s = 1, m-1 \\ \qquad \mathbf{F}'(\mathbf{x}_0)\,\boldsymbol{\phi}_{s+1} = \mathbf{F}(\mathbf{x}_s) \\ \qquad \mathbf{x}_{s+1} = \mathbf{x}_s - \boldsymbol{\phi}_{s+1} \\ \text{end} \end{cases}
\end{aligned}
$$

where $\mathbf{F}'(\cdot)$ is the Fréchet derivative [3, 70] or Jacobian of $\mathbf{F}(\cdot)$, $\mathbf{x}_0$ is the initial approximation vector $\mathbf{x}$ for the solution of $\mathbf{F}(\mathbf{x}) = 0$, and $\mathbf{x}_m$ is the approximation vector $\mathbf{x}$ for the solution of $\mathbf{F}(\mathbf{x})$ after an iteration of NR. The NR method uses m (≥ 1) steps to obtain a $m+1$ convergence order, makes m function evaluations and one Jacobian evaluation, and requires only one LU decomposition and m solutions of lower and upper triangular systems. We will construct a new multi-step method which enhances the radius of convergence and the speed of convergence of NR. We will develop the new method by introducing a parameter in NR. A similar idea for scalar algebraic equations has been suggested in [71]. Although we use LU decompositions for solving linear systems in both the base method and the multi-step part, iterative methods such as restarted GMRES could also be used.

2.2 New multi-step iterative method

Our new iterative method came out by an attempt to increase the convergence radius in NR without changing its convergence rate and its computational cost. The resulting method (ATC) can be described as

$$\text{Base method} \rightarrow \begin{cases} \mathbf{F}'(\mathbf{x}_0)\,\boldsymbol{\phi}_1 = \mathbf{F}(\mathbf{x}_0) \\ \mathbf{y}_1 = \mathbf{x}_0 - (1+\theta-\theta^2)\,\boldsymbol{\phi}_1 \\ \mathbf{F}'(\mathbf{x}_0)\,\boldsymbol{\phi}_2 = \mathbf{F}\left(\mathbf{x}_0 - \frac{1}{\theta}\boldsymbol{\phi}_1\right) \\ \mathbf{x}_2 = \mathbf{x}_1 - \theta^2\,\boldsymbol{\phi}_2 \end{cases}$$

$$\text{Multi-step part} \rightarrow \begin{cases} \text{For } s = 1,\, m-2 \\ \qquad \mathbf{F}'(\mathbf{x}_0)\,\boldsymbol{\phi}_{s+2} = \mathbf{F}(\mathbf{x}_{s+1}) \\ \qquad \mathbf{x}_{s+2} = \mathbf{x}_{s+1} - \boldsymbol{\phi}_{s+2} \\ \text{end} \end{cases}$$

where $\theta \neq 0$, $\mathbf{x}_0$ is the initial aproximation vector $\mathbf{x}$ for the solution of $\mathbf{F}(\mathbf{x}) = 0$ and $\mathbf{x}_m$ is the approximation vector $\mathbf{x}$ for the solution of $\mathbf{F}(\mathbf{x}) = 0$ after an iteration of the method. The ATC method needs m (≥ 2) steps to obtain a $m+1$ convergence order, makes m function evaluations and one Jacobian evaluation, and requires one LU decomposition, 3 vector-vector multiplications and m solutions of lower and upper triangular systems. The more computationally expensive operations are the LU factorization of the Jacobian and the solutions of the upper and lower triangular systems. Picking up $\theta = 1$ reduces the new method ATC to NR, so the new method can be seen as a generalization of NR keeping the same convergence order. It is clear that by an appropriate selection for the θ parameter the new method can be made to have faster convergence than NR and to have larger convergence radius than NR. While we don't currently have a strategy for picking up a good value for θ, it is possible that such strategies can be developed in the future for particular instances or classes of functions $\mathbf{F}(\cdot)$ such as functions $\mathbf{F}(\cdot)$ arising when solving the Poisson partial differential equation. We will verify that faster convergence is achieved in ATC with respect to NR. That faster convergence must be attributed to the fact that the leading term of the error has smaller value in norm in ATC.

2.3 Convergence analysis

In this section we first prove that the order of convergence of ATC is four when $m = 3$. Later, we will prove via induction that the order of convergence of ATC is $m + 1$. In the constructed proof, we require that the function $\mathbf{F}(\cdot)$ should have at least three Fréchet derivatives. The function $\mathbf{F} : \Gamma \subseteq \mathbb{R}^n \to \mathbb{R}^r$ is Fréchet differentiable [3] at $\mathbf{x} \in \text{interior}(\Gamma)$ if there is an $\mathbf{A} \in \mathbb{L}(\mathbb{R}^n, \mathbb{R}^r)$ such that

$$\lim_{\mathbf{h}\to\mathbf{0}} \frac{||\mathbf{F}(\mathbf{x}+\mathbf{h}) - \mathbf{F}(\mathbf{x}) - \mathbf{A}\mathbf{h}||}{||\mathbf{h}||} = 0\,.$$

The linear operator $\mathbf{A}$ is denoted by $\mathbf{F}'(\mathbf{x})$ and is called the Fréchet derivative of $\mathbf{F}(\cdot)$ at $\mathbf{x}$. The higher-order Fréchet derivative of $\mathbf{F}(\mathbf{x})$ with respect to $\mathbf{x}$ can be calculated recursively

$$\begin{aligned}
&\mathbf{F}'(\mathbf{x}) = \text{Jacobian}\left(\mathbf{F}(\mathbf{x})\right),\\
&\mathbf{F}^s(\mathbf{x})\mathbf{v}^{s-1} = \text{Jacobian}\left(\mathbf{F}^{s-1}(\mathbf{x})\mathbf{v}^{s-1}\right), \quad s \geq 2\,,
\end{aligned}$$

where $\mathbf{v}$ is vector.

Theorem 2.1. *Let $\mathbf{F} : \Gamma \subseteq \mathbb{R}^n \to \mathbb{R}^n$ be a function with up to third order Fréchet derivative on an open convex neighborhood Γ of $\mathbf{x}^* \in \mathbb{R}^n$ with $\mathbf{F}(\mathbf{x}^*) = 0$ and $\det(\mathbf{F}'(\mathbf{x}^*)) \neq 0$, where $\mathbf{F}'(\mathbf{x})$ denotes the Fréchet derivative of $\mathbf{F}(\mathbf{x})$. Let $\mathbf{A}_1 = \mathbf{F}'(\mathbf{x}^*)$ and $\mathbf{A}_s = \frac{1}{s!}\mathbf{F}'(\mathbf{x}^*)^{-1}\mathbf{F}^{(s)}(\mathbf{x}^*)$, for $s \geq 2$, where $\mathbf{F}^{(s)}(\mathbf{x})$ denotes s-order Fréchet derivative of $\mathbf{F}(\mathbf{x})$. Then, for $m = 3$, with an initial guess in the neighborhood of $\mathbf{x}^*$, the sequence $\{\mathbf{x}_k\}$ generated by ATC converges to $\mathbf{x}^*$ with local order of convergence at least four and error*

$$\mathbf{e}_{k+1} = \mathbf{L}{\mathbf{e}_k}^4 + O\left({\mathbf{e}_k}^5\right)\,,$$

where $\mathbf{e}_k = \mathbf{x_k} - \mathbf{x}^*$, $\mathbf{e}_k{}^p = \overbrace{(\mathbf{e}_k, \mathbf{e}_k, \ldots, \mathbf{e}_k)}^{p\ times}$, *and* $\mathbf{L} = -\left(2\,(1 - 1/\theta)\ \mathbf{A}_2\mathbf{A}_3 - 4\,\mathbf{A}_2^3\right)$ *is a 4-linear function, i.e.* $\mathbf{L} \in \mathbb{L}(\mathbb{R}^n, \mathbb{R}^n, \mathbb{R}^n, \mathbb{R}^n)$ *with* $\mathbf{L}\mathbf{e}_k{}^4 \in \mathbb{R}^n$*.*

Proof. Let $\mathbf{F} : \Gamma \subseteq \mathbb{R}^n \to \mathbb{R}^n$ be a function with up to third order Fréchet derivative in Γ. The qth Fréchet derivative of $\mathbf{F}$ at $v \in \mathbb{R}^n$, $q \geq 1$, is a q-linear function $\mathbf{F}^{(q)}(v) : \overbrace{\mathbb{R}^n\mathbb{R}^n \cdots \mathbb{R}^n}^{q \text{ times}}$ with $\mathbf{F}^{(q)}(v)(u_1, u_2, \cdots, u_q) \in \mathbb{R}^n$. Taylor's series expansion of $\mathbf{F}(\mathbf{x}_k)$ around $\mathbf{x}^*$ is

$$\begin{aligned}\mathbf{F}\left(\mathbf{x}_k\right) &= \mathbf{F}\left(\mathbf{x}^* + \mathbf{x}_k - \mathbf{x}^*\right) = \mathbf{F}(\mathbf{x}^* + \mathbf{e}_k)\\ &= \mathbf{F}\left(\mathbf{x}^*\right) + \mathbf{F}'(\mathbf{x}^*)\,\mathbf{e}_k + \frac{1}{2!}\,\mathbf{F}''(\mathbf{x}^*)\,\mathbf{e}_k^2 + \frac{1}{3!}\,\mathbf{F}^{(3)}(\mathbf{x}^*)\,\mathbf{e}_k^3 +\\ &\frac{1}{4!}\,\mathbf{F}^{(4)}(\mathbf{x}^*)\,\mathbf{e}_k^4 + \cdots\\ &= \mathbf{F}'(\mathbf{x}^*)\Big(\mathbf{e}_k + \frac{1}{2!}\,\mathbf{F}'(\mathbf{x}^*)^{-1}\,\mathbf{F}''(\mathbf{x}^*)\,\mathbf{e}_k^2 + \frac{1}{3!}\,\mathbf{F}'(\mathbf{x}^*)^{-1}\\ &\mathbf{F}^{(3)}(\mathbf{x}^*)\,\mathbf{e}_k^3 + \frac{1}{4!}\,\mathbf{F}'(\mathbf{x}^*)^{-1}\,\mathbf{F}^{(4)}(\mathbf{x}^*)\,\mathbf{e}_k^4 + \cdots\Big)\\ &= \mathbf{A}_1\Bigg(\mathbf{e}_k + \mathbf{A}_2\,\mathbf{e}_k^2 + \mathbf{A}_3\,\mathbf{e}_k^3 + \mathbf{A}_4\,\mathbf{e}_k^4 + O\left(\mathbf{e}_k{}^5\right)\Bigg).\end{aligned} \tag{2.1}$$

Computing the Fréchet derivative of $\mathbf{F}$ with respect to $\mathbf{e}_k$, we get

$$\mathbf{F}'(\mathbf{x}_k) = \mathbf{A}_1\Bigg(\mathbf{I} + 2\mathbf{A}_2\mathbf{e}_k + 3\mathbf{A}_3\mathbf{e}_k{}^2 + 4\mathbf{A}_4\mathbf{e}_k{}^3 + O\left(\mathbf{e}_k{}^4\right)\Bigg),$$

where $\mathbf{I}$ is the identity matrix. Computing its inverse using a symbolic mathematical package Maple, we obtain

$$\begin{aligned}\mathbf{F}'(\mathbf{x}_k)^{-1} = \Bigg(&\mathbf{I} - 2\mathbf{A}_2\mathbf{e}_k + \Big(4\mathbf{A}_2^2 - 3\mathbf{A}_3\Big)\mathbf{e}_k^2 + \Big(6\mathbf{A}_3\mathbf{A}_2 + 6\mathbf{A}_2\mathbf{A}_3 -\\ &8\mathbf{A}_2^3 - 4\mathbf{A}_4\Big)\mathbf{e}_k^3 + \Big(8\mathbf{A}_4\mathbf{A}_2 + 9\mathbf{A}_3^2 + 8\mathbf{A}_2\mathbf{A}_4 - 5\mathbf{A}_5 - 12\mathbf{A}_3\mathbf{A}_2^2\\ &- 12\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 - 12\mathbf{A}_2^2\mathbf{A}_3 + 16\mathbf{A}_2^4\Big)\mathbf{e}_k^4 + O\Big(\mathbf{e}_k{}^5\Big)\Bigg)\,\mathbf{A}_1^{-1}.\end{aligned} \tag{2.2}$$

To clarify the notation in the rest of the proof, we note that $\mathbf{x}_k$ is the vector $\mathbf{x}_0$ used in the description of ATC and that $\mathbf{x}_{k+1}$ is the vector $\mathbf{x}_3$ in the description of ATC. The vectors $\boldsymbol{\phi}_1$, $\boldsymbol{\phi}_2$, $\boldsymbol{\phi}_3$, $\mathbf{x}_1$, and $\mathbf{x}_2$ will denote the vectors with same names in the description of ATC which allow to go from $\mathbf{x}_k$ to $\mathbf{x}_{k+1}$ when ATC is applied.

Using $\boldsymbol{\phi}_1 = \mathbf{F}'(\mathbf{x}_k)^{-1}\mathbf{F}(\mathbf{x}_k)$, we get

$$\begin{aligned}
\boldsymbol{\phi}_1 = {} & \big(\mathbf{I} - 2\mathbf{A}_2\mathbf{e}_k + \big(4\mathbf{A}_2^2 - 3\mathbf{A}_3\big)\mathbf{e}_k^2 + \big(6\mathbf{A}_3\mathbf{A}_2 + 6\mathbf{A}_2\mathbf{A}_3 - 8\mathbf{A}_2^3 - 4\mathbf{A}_4\big)\mathbf{e}_k^3 + \\
& \big(8\mathbf{A}_4\mathbf{A}_2 + 9\mathbf{A}_3^2 + 8\mathbf{A}_2\mathbf{A}_4 - 5\mathbf{A}_5 - 12\mathbf{A}_3\mathbf{A}_2^2 - 12\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 - 12\mathbf{A}_2^2\mathbf{A}_3 \\
& + 16\mathbf{A}_2^4\big)\mathbf{e}_k^4 + O\big({\mathbf{e}_k}^5\big)\big)\big(\mathbf{e}_k + \mathbf{A}_2\,\mathbf{e}_k^2 + \mathbf{A}_3\,\mathbf{e}_k^3 + O\big({\mathbf{e}_k}^4\big)\big) \\
= {} & \mathbf{e}_k - \mathbf{A}_2\,\mathbf{e}_k^2 + \big(2\mathbf{A}_2^2 - 2\mathbf{A}_3\big)\,\mathbf{e}_k^3 + \big(4\mathbf{A}_2\mathbf{A}_3 + 3\mathbf{A}_3\mathbf{A}_2 - 4\mathbf{A}_2^3 - 3\mathbf{A}_4\big)\,\mathbf{e}_k^4 \\
& + \big(4\mathbf{A}_4\mathbf{A}_2 + 6\mathbf{A}_3^2 + 6\mathbf{A}_2\mathbf{A}_4 + 8\mathbf{A}_2^4 - 6\mathbf{A}_3\mathbf{A}_2^2 - 6\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 - 8\mathbf{A}_2^2\mathbf{A}_3 \\
& - 4\mathbf{A}_5\big)\,\mathbf{e}_k^5 + O\big({\mathbf{e}_k}^6\big)\,.
\end{aligned} \tag{2.3}$$

Using $\mathbf{x}_1 = \mathbf{x}_k - \big(1 + \theta - \theta^2\big)\,\boldsymbol{\phi}_1$ and plugging (2.3) we get

$$\begin{aligned}
\mathbf{y}_1 - \mathbf{x}^* &= \mathbf{x}_k - \mathbf{x}^* - \big(1 + \theta + \theta^2\big)\,\boldsymbol{\phi}_1 \\
&= \mathbf{e}_k - \big(1 + \theta + \theta^2\big)\big(\mathbf{e}_k - \mathbf{A}_2\,\mathbf{e}_k^2 + \big(2\mathbf{A}_2^2 - 2\mathbf{A}_3\big)\,\mathbf{e}_k^3 + \big(4\mathbf{A}_2\mathbf{A}_3 \\
&\quad + 3\mathbf{A}_3\mathbf{A}_2 - 4\mathbf{A}_2^3 - 3\mathbf{A}_4\big)\,\mathbf{e}_k^4 + \big(4\mathbf{A}_4\mathbf{A}_2 + 6\mathbf{A}_3^2 + 6\mathbf{A}_2\mathbf{A}_4 + \\
&\quad 8\mathbf{A}_2^4 - 6\mathbf{A}_3\mathbf{A}_2^2 - 6\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 - 8\mathbf{A}_2^2\mathbf{A}_3 - 4\mathbf{A}_5\big)\,\mathbf{e}_k^5 + O\big({\mathbf{e}_k}^6\big)\big) \\
&= \big(\theta^2 - \theta\big)\,\mathbf{e}_k - \big(\theta^2 - \theta - 1\big)\mathbf{A}_2\,\mathbf{e}_k^2 + \Bigg(\big(\theta^2 - \theta - 1\big)\big(\mathbf{A}_2^2 \\
&\quad - \mathbf{A}_3\big)\Bigg)\,\mathbf{e}_k^3 + \Bigg(\big(\theta^2 - \theta - 1\big)\big(-4\mathbf{A}_2^3 + 3\mathbf{A}_3\mathbf{A}_2 + 4\mathbf{A}_2\mathbf{A}_3 \\
&\quad - 3\mathbf{A}_4\big)\Bigg)\,\mathbf{e}_k^4 + O\big({\mathbf{e}_k}^5\big)\,.
\end{aligned} \tag{2.4}$$

Using $\boldsymbol{\phi}_2 = \mathbf{F}'(\mathbf{x}_k)^{-1}\,\mathbf{F}(\mathbf{x}_k - \boldsymbol{\phi}_1/\theta)$ and substituting (2.2) and (2.1) we get

$$\begin{aligned}
\boldsymbol{\phi}_2 = {} & \big(\mathbf{I} - 2\mathbf{A}_2\mathbf{e}_k + \big(4\mathbf{A}_2^2 - 3\mathbf{A}_3\big)\mathbf{e}_k^2 + \big(6\mathbf{A}_3\mathbf{A}_2 + 6\mathbf{A}_2\mathbf{A}_3 - 8\mathbf{A}_2^3 - 4\mathbf{A}_4\big)\mathbf{e}_k^3 \\
& + \big(8\mathbf{A}_4\mathbf{A}_2 + 9\mathbf{A}_3^2 + 8\mathbf{A}_2\mathbf{A}_4 - 5\mathbf{A}_5 - 12\mathbf{A}_3\mathbf{A}_2^2 - 12\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 - 12\mathbf{A}_2^2\mathbf{A}_3 \\
& + 16\mathbf{A}_2^4\big)\mathbf{e}_k^4 + O\big({\mathbf{e}_k}^5\big)\big)\Bigg(\mathbf{e}_k - \frac{\boldsymbol{\phi}_1}{\theta} + \mathbf{A}_2\Big(\mathbf{e}_k - \frac{\boldsymbol{\phi}_1}{\theta}\Big)^2 + \mathbf{A}_3\Big(\mathbf{e}_k - \frac{\boldsymbol{\phi}_1}{\theta}\Big)^3 \\
& + \mathbf{A}_4\Big(\mathbf{e}_k - \frac{\boldsymbol{\phi}_1}{\theta}\Big)^4 + O(\Big(\mathbf{e}_k - \frac{\boldsymbol{\phi}_1}{\theta}\Big)^5)\Bigg) \\
= {} & \Big(1 - \frac{1}{\theta}\Big)\,\mathbf{e}_k + \Big(-1 + \frac{1}{\theta} + \frac{1}{\theta^2}\Big)\mathbf{A}_2\,\mathbf{e}_k^2 + \Bigg(2\Big(1 - \frac{1}{\theta} - \frac{2}{\theta^2}\Big)\mathbf{A}_2^2 + \\
& \Big(-2 + \frac{2}{\theta} + \frac{3}{\theta^2} - \frac{1}{\theta^3}\Big)\mathbf{A}_3\Bigg)\,\mathbf{e}_k^3 + \Bigg(3\Big(1 - \frac{1}{\theta} - \frac{3}{\theta^2} - \frac{1}{\theta^3}\Big)\mathbf{A}_3\mathbf{A}_2 + \\
& \Big(-4 + \frac{4}{\theta} + \frac{13}{\theta^2}\Big)\mathbf{A}_2^3 + 2\Big(2 - \frac{2}{\theta} - \frac{5}{\theta^2} + \frac{1}{\theta^3}\Big)\mathbf{A}_2\mathbf{A}_3 \\
& + (-3 + \frac{3}{\theta} + \frac{6}{\theta^2} - \frac{4}{\theta^3} + \frac{1}{\theta^4})\mathbf{A}_4\Bigg)\,\mathbf{e}_k^4 + O\big({\mathbf{e}_k}^5\big)\,.
\end{aligned} \tag{2.5}$$

Using $\mathbf{x}_2 = \mathbf{x}_1 - \theta^2 \boldsymbol{\phi}_2$ and substituting (2.4) and (2.5),

$$
\begin{aligned}
\mathbf{x}_2 - \mathbf{x}^* &= \mathbf{x}_1 - \mathbf{x}^* - \theta^2 \boldsymbol{\phi}_2 \\
&= \Big(\theta^2 - \theta\Big)\mathbf{e}_k - \Big(\theta^2 - \theta - 1\Big)\mathbf{A}_2\,\mathbf{e}_k^2 + \Big(\Big(\theta^2 - \theta - 1\Big) \\
&\quad \Big(\mathbf{A}_2^2 - \mathbf{A}_3\Big)\Big)\mathbf{e}_k^3 + \Big(\Big(\theta^2 - \theta - 1\Big)\Big(-4\mathbf{A}_2^3 + 3\mathbf{A}_3\mathbf{A}_2 + \\
&\quad 4\mathbf{A}_2\mathbf{A}_3 - 3\mathbf{A}_4\Big)\Big)\mathbf{e}_k^4 + O\Big({\mathbf{e}_k}^5\Big) - \theta^2\Big(\Big(1 - \frac{1}{\theta}\Big)\mathbf{e}_k + \Big(-1 + \frac{1}{\theta} + \\
&\quad \frac{1}{\theta^2}\Big)\mathbf{A}_2\,\mathbf{e}_k^2 + \Big(2\Big(1 - \frac{1}{\theta} - \frac{2}{\theta^2}\Big)\mathbf{A}_2^2 + \Big(-2 + \frac{2}{\theta} + \frac{3}{\theta^2} - \frac{1}{\theta^3}\Big)\mathbf{A}_3\Big)\mathbf{e}_k^3 \\
&\quad + \Big(3\Big(1 - \frac{1}{\theta} - \frac{3}{\theta^2} - \frac{1}{\theta^3}\Big)\mathbf{A}_3\mathbf{A}_2 + \Big(-4 + \frac{4}{\theta} + \frac{13}{\theta^2}\Big)\mathbf{A}_2^3 + 2\Big(2 - \frac{2}{\theta} - \\
&\quad \frac{5}{\theta^2} + \frac{1}{\theta^3}\Big)\mathbf{A}_2\mathbf{A}_3 + \Big(-3 + \frac{3}{\theta} + \frac{6}{\theta^2} - \frac{4}{\theta^3} + \frac{1}{\theta^4}\Big)\mathbf{A}_4\Big)\mathbf{e}_k^4 + O\Big({\mathbf{e}_k}^5\Big)\Big)
\end{aligned}
$$

$$
\begin{aligned}
&= \Big(2\mathbf{A}_2^2 + \Big(1 - \frac{1}{\theta}\Big)\mathbf{A}_3\Big)\mathbf{e}_k^3 + \Big(3\Big(2 - \frac{1}{\theta}\Big)\mathbf{A}_3\mathbf{A}_2 - 9\mathbf{A}_2^3 + 2\Big(3 - \frac{1}{\theta}\Big) \\
&\mathbf{A}_2\mathbf{A}_3 + \Big(-3 + \frac{4}{\theta} - \frac{1}{\theta^2}\Big)\mathbf{A}_4\Big)\mathbf{e}_k^4 + O\Big({\mathbf{e}_k}^5\Big).
\end{aligned}
\tag{2.6}
$$

Substituting $\mathbf{x}_k$ by $\mathbf{x}_2$ in (2.1) we get, replacing $\mathbf{x}_k$ by the previous expression for $\mathbf{x}_2 - \mathbf{x}^*$, and with $\mathbf{z} = \mathbf{x}^* + (\theta^2 - \theta)\mathbf{e}_k - (\theta^2 - \theta - 1)\mathbf{A}_2\,\mathbf{e}_k^2 + ((\theta^2 - \theta - 1)(\mathbf{A}_2^2 - \mathbf{A}_3))\mathbf{e}_k^3 + ((\theta^2 - \theta - 1)(-4\mathbf{A}_2^3 + 3\mathbf{A}_2\mathbf{A}_2 + 4\mathbf{A}_2\mathbf{A}_3 - 3\mathbf{A}_4))\mathbf{e}_k^4 + O(\mathbf{e}_k^5)$,

$$
\begin{aligned}
\mathbf{F}(\mathbf{x}_2) &= \mathbf{F}\Big(\mathbf{x}^* + \Big(\theta^2 - \theta\Big)\mathbf{e}_k - \Big(\theta^2 - \theta - 1\Big)\mathbf{A}_2\,\mathbf{e}_k^2 + \Big(\Big(\theta^2 - \theta - 1\Big) \\
&\quad \Big(\mathbf{A}_2^2 - \mathbf{A}_3\Big)\Big)\mathbf{e}_k^3 + \Big(\Big(\theta^2 - \theta - 1\Big)\Big(-4\mathbf{A}_2^3 + 3\mathbf{A}_3\mathbf{A}_2 + 4\mathbf{A}_2\mathbf{A}_3 - \\
&\quad 3\mathbf{A}_4\Big)\Big)\mathbf{e}_k^4 + O\Big({\mathbf{e}_k}^5\Big)\Big) \\
&= \mathbf{A}_1\Big(\mathbf{z} + \mathbf{A}_2\,\mathbf{z}^2 + \mathbf{A}_3\,\mathbf{z}^3 + \mathbf{A}_4\,\mathbf{z}^4 + O\Big(\mathbf{z}^5\Big)\Big) \\
&= \mathbf{A}_1\Big(\Big(2\mathbf{A}_2^2 + \Big(-1 + \frac{1}{\theta}\Big)\mathbf{A}_3\Big)\mathbf{e}_k^3 + \Big(\Big(-3 + \frac{4}{\theta} - \frac{1}{\theta^2}\Big)\mathbf{A}_4 + \\
&\quad 2\Big(3 - \frac{1}{\theta}\Big)\mathbf{A}_2\mathbf{A}_3 + 3\Big(2 - \frac{1}{\theta}\Big)\mathbf{A}_3\mathbf{A}_2 - 9\mathbf{A}_2^3\Big)\mathbf{e}_k^4 + O\Big({\mathbf{e}_k}^5\Big)\Big).
\end{aligned}
\tag{2.7}
$$

Using $\boldsymbol{\phi}_3 = \mathbf{F}'(\mathbf{x}_k)^{-1}\mathbf{F}(\mathbf{x}_2)$ and substituting (2.2) and (2.7),

$$\begin{aligned}\boldsymbol{\phi}_3 = \mathbf{F}'(\mathbf{x}_k)^{-1}\mathbf{F}(\mathbf{x}_2) = \Bigg(&\mathbf{I} - 2\mathbf{A}_2\mathbf{e}_k + \Big(4\mathbf{A}_2^2 - 3\mathbf{A}_3\Big)\mathbf{e}_k^2 + \Big(6\mathbf{A}_3\mathbf{A}_2 + \\ &6\mathbf{A}_2\mathbf{A}_3 - 8\mathbf{A}_2^3 - 4\mathbf{A}_4\Big)\mathbf{e}_k^3 + \Big(8\mathbf{A}_4\mathbf{A}_2 + 9\mathbf{A}_3^2 + 8\mathbf{A}_2\mathbf{A}_4 - 5\mathbf{A}_5 - \\ &12\mathbf{A}_3\mathbf{A}_2^2 - 12\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 - 12\mathbf{A}_2^2\mathbf{A}_3 + 16\mathbf{A}_2^4\Big)\mathbf{e}_k^4 + O\Big({\mathbf{e}_k}^5\Big)\Bigg) \\ &\Bigg(\Bigg(2\mathbf{A}_2^2 + \Big(-1 + \frac{1}{\theta}\Big)\mathbf{A}_3\Bigg)\mathbf{e}_k^3 + \Bigg(\Big(-3 + \frac{4}{\theta} - \frac{1}{\theta^2}\Big)\mathbf{A}_4 + \\ &2\Big(3 - \frac{1}{\theta}\Big)\mathbf{A}_2\mathbf{A}_3 + 3\Big(2 - \frac{1}{\theta}\Big)\mathbf{A}_3\mathbf{A}_2 - 9\mathbf{A}_2^3\Bigg)\mathbf{e}_k^4 + O\Big({\mathbf{e}_k}^5\Big)\Bigg)\end{aligned}$$

$$\begin{aligned}= &\Bigg(2\mathbf{A}_2^2 + \Big(-1 + \frac{1}{\theta}\Big)\mathbf{A}_3\Bigg)\mathbf{e}_k^3 + \Bigg(\Big(2 - \frac{1}{\theta}\Big)\Big(3\mathbf{A}_3\mathbf{A}_2 + 4\mathbf{A}_2\mathbf{A}_3\Big) - \\ &13\mathbf{A}_2^3 + \Big(-3 - \frac{1}{\theta} + \frac{4}{\theta^2}\Big)\mathbf{A}_4\Bigg)\mathbf{e}_k^4 + O\Big({\mathbf{e}_k}^5\Big).\end{aligned} \tag{2.8}$$

Using $\mathbf{x}_3 = \mathbf{x}_2 - \boldsymbol{\phi}_3$ and substituting (2.6) and (2.8),

$$\begin{aligned}\mathbf{x}_3 - \mathbf{x}^* = \mathbf{x}_2 - \mathbf{x}^* - \boldsymbol{\phi}_3 = &\Bigg(\Bigg(2\mathbf{A}_2^2 + \left(1 - \frac{1}{\theta}\right)\mathbf{A}_3\Bigg)\mathbf{e}_k^3 + \Bigg(3\left(2 - \frac{1}{\theta}\right)\mathbf{A}_3\mathbf{A}_2 - \\ &9\mathbf{A}_2^3 + 2\left(3 - \frac{1}{\theta}\right)\mathbf{A}_2\mathbf{A}_3 + \left(-3 + \frac{4}{\theta} - \frac{1}{\theta^2}\right)\mathbf{A}_4\Bigg)\mathbf{e}_k^4 + O\left({\mathbf{e}_k}^5\right)\Bigg) \\ &\Bigg(\Bigg(2\mathbf{A}_2^2 + \left(-1 + \frac{1}{\theta}\right)\mathbf{A}_3\Bigg)\mathbf{e}_k^3 + \Bigg(\left(2 - \frac{1}{\theta}\right)(3\mathbf{A}_3\mathbf{A}_2 + 4\mathbf{A}_2\mathbf{A}_3) - \\ &13\mathbf{A}_2^3 + \left(-3 - \frac{1}{\theta} + \frac{4}{\theta^2}\right)\mathbf{A}_4\Bigg)\mathbf{e}_k^4 + O\left({\mathbf{e}_k}^5\right)\Bigg) \\ &= 2\mathbf{A}_2\Bigg(\left(\frac{1}{\theta} - 1\right)\mathbf{A}_3 + 2\mathbf{A}_2^2\Bigg)\mathbf{e}_k^4 + O\left({\mathbf{e}_k}^5\right).\end{aligned}$$

□

Note that to work with non-commutative algebra one may use following commands in Maple.

```
with(Physics):
```

```
Setup(mathematicalnotation = true):
quantumOperators := {A1, A2, A3, A4}:
Setup(quantumoperators = quantumOperators):
```

Theorem 2.2. *Let* $\mathbf{F} : \Gamma \subseteq \mathbb{R}^n \to \mathbb{R}^n$ *has at least third order Fréchet derivative on an open convex neighborhood* Γ *of* $\mathbf{x}^* \in \mathbb{R}^n$ *with* $\mathbf{F}(\mathbf{x}^*) = 0$ *and* $\det(\mathbf{F}'(\mathbf{x}^*)) \neq 0$*. Then, the multi-step ATC iterative method has, for* $m \geq 2$*, local convergence order at least* $m + 1$*.*

Proof. The proof can be obtained via mathematical induction as done in [70]. □

The error equation for the m-step iterative method ATC is calculated by using the Maple symbolic toolbox that can be written as

$$\mathbf{x}_m - \mathbf{x}^* = (2\mathbf{A}_2)^{m-2} \left(\left(\frac{1}{\theta} - 1 \right) \mathbf{A}_3 + 2\mathbf{A}_2^2 \right) \mathbf{e}^{m+1} + O\left(\mathbf{e}^{m+2}\right), \quad m \geq 2. \tag{2.9}$$

The highest Fréchet derivative in the error equation (2.9) is third order. So, the m-step iterative method ATC has $m + 1$ convergence order and it requires that the nonlinear function $\mathbf{F}(\cdot)$ should have at least three Fréchet derivatives. Note that wide classes of important ODEs and PDEs, such as those arising in the Bratu problem, the Frank-Kamenetzkii problem [72], the Lene-Emden equation [73], the Burgers equation [74], the Klein-Gordon equation [75], the two-dimensional sinh-Poisson equation [76], and the three-dimensional nonlinear Poisson equation [77], heat equation, wave equation, Euler's beam equation etc., give rise to $\mathbf{F}(\mathbf{x})$ functions with high order Fréchet derivatives. Then, the multi-step iterative method ATC is applicable to wide classes of important problems. The real parameter $\theta(\neq 0)$ in ATC can be replaced by a vector of non-zero real numbers when $\mathbf{A}_j$ for $j \geq 2$ are diagonal matrices and usually it is the case in the systems of nonlinear equations associated with ODEs and PDEs. The diagonal matrices can be treated as vectors and we define binary and unary operations for them as

$$\begin{gathered}
\begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} = \begin{bmatrix} v_1\, u_1 \\ v_2\, u_2 \\ \vdots \\ v_n\, u_n \end{bmatrix}, \quad \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}^{-1} = \begin{bmatrix} 1/v_1 \\ 1/v_2 \\ \vdots \\ 1/v_n \end{bmatrix} \\
\begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \pm \text{constant} = \begin{bmatrix} v_1 \pm \text{constant} \\ v_2 \pm \text{constant} \\ \vdots \\ v_n \pm \text{constant} \end{bmatrix},
\end{gathered} \tag{2.10}$$

and the error equation (2.9) is verified as

$$\mathbf{x}_m - \mathbf{x}^* = \left(\left(\frac{1}{\theta} - 1 \right) (2\mathbf{A}_2)^{m-2}\, \mathbf{A}_3 + 2\,(2\mathbf{A}_2)^m \right) \mathbf{e}^{m+1} + O\left(\mathbf{e}^{m+2}\right), \quad m \geq 2.$$

In that error equation, $(2\mathbf{A}_2)^{m-2}\, \mathbf{A}_3\mathbf{e}^{m+1}$ and $(2\mathbf{A}_2)^m\, \mathbf{e}^{m+1}$ are vectors and $\left(\frac{1}{\theta} - 1\right)\left((2\mathbf{A}_2)^{m-2}\, \mathbf{A}_3\mathbf{e}^{m+1}\right)$ is calculated using (2.10).

2.4 A real test problem

In this section, to illustrate the application of the multi-step ATC iterative method, we will consider the nonlinear complex generalized Zakharov system (GZS) of one dimensional PDEs with the Chebyshev pseudo-spectral method for discretize it in spatial and temporal dimensions to reduce it to a nonlinear system of algebraic equations.

2.4.1 The nonlinear complex generalized Zakharov system

The nonlinear complex Zakharov system has importance in plasma physics [78]. The system includes two coupled nonlinear PDEs which can be written as

$$\begin{aligned} & i\; \partial_t\, \psi(x,t) + \delta_1 \partial_{tt}\, \psi(x,t) - \delta_2\, \psi(x,t)\, w(x,t) + \delta_3\, |\psi(x,t)|^2 \psi(x,t) = 0 \\ & \partial_{tt}\, w(x,t) - c_s^2 \partial_{xx}\, w(x,t) - \delta_4\, \partial_{xx}\, |\psi(x,t)|^2 = 0 \\ & \text{for } (x,\ t) \in (a_x,\ b_x) \times (a_t,\ b_t), \end{aligned} \tag{2.11}$$

subject to the initial and boundary conditions

$$\begin{array}{ll} \psi(a_x,\ t) = \psi_1(t)\,, & \psi(b_x,\ t) = \psi_2(t) \\ \psi(x,\ 0) = \psi_0(x)\,, & w(a_x,\ t) = w_1(t) \\ w(b_x,\ t) = w_2(x)\,, & w(x,\ 0) = w_3(x) \end{array} \quad , \quad (x,\ t) \in [a_x,\ b_x] \times [a_t,\ b_t]\,. \tag{2.12}$$

Several numerical methods have been proposed recently for approximating the solution of (2.11)–(2.12) such as the homotopy method [79], the finite difference method [80, 81], and the variational iteration method [82]. Also, Bao et al. [83]

suggested some high-accurate numerical methods for solving numerically (2.11)–(2.12). Bao and Sun [84] applied a new technique based on time-splitting discretization for approximating the solution of a variant of (2.11)–(2.12).

One can split (2.11) using the real and imaginary parts of $\psi(x,t)$, $u(x,t)$ and $v(x,t)$, as

$$\begin{aligned}
&\partial_t\, u(x,t) + \delta_1 \partial_{xx}\, v(x,t) - \delta_2 v(x,t) w(x,\ t) + \delta_3 \left(u^2(x,t) + v^2(x,t)\right)\\
&v(x,t) = 0\,,\\
&-\partial_t\, v(x,t) + \delta_1 \partial_{xx}\, u(x,t) - \delta_2 u(x,t) w(x,t) + \delta_3 \left(u^2(x,t) + v^2(x,t)\right)\\
&u(x,t) = 0\,,\\
&\partial_{tt}\, w(x,t) - c_s^2 \partial_{xx}\, w(x,t) - 2\delta_4 \big(u(x,t)\partial_{xx}\, u(x,t) + \left(\partial_x\, u(x,t)\right)^2\\
&+ v(x,t)\partial_{xx}\, v(x,t) + \left(\partial_x\, v(x,t)\right)^2\big) = 0\,,
\end{aligned} \tag{2.13}$$

with the initial and boundary conditions

$$\left.\begin{array}{ll}
u(a_x,t) = \alpha_1(t)\,, & u(b_x,t) = \alpha_2(t)\\
v(a_x,t) = \alpha_3(t)\,, & v(b_x,t) = \alpha_4(t)\\
w(a_x,t) = \alpha_5(t)\,, & w(b_x,t) = \alpha_6(t)\\
u(x,a_t) = \beta_1(x)\,, & v(x,a_t) = \beta_2(x)\\
w(x,a_t) = \beta_3(x)\,, & w_t(x,a_t) = \beta_4(x)
\end{array}\right. ,\quad (x,\ t) \in [a_x,\ b_x] \times [a_t,\ b_t]\,. \tag{2.14}$$

The matrix form of the nonlinear system (2.13) is

$$\begin{bmatrix} \partial_t & \delta_1\partial_{xx} & 0\\ \delta_1\partial_{xx} & -\partial_t & 0\\ 0 & 0 & \partial_{tt} - c_s^2\partial_{xx} \end{bmatrix} \begin{bmatrix} u\\ v\\ w \end{bmatrix} + \begin{bmatrix} q_1(u,v,w)\\ q_2(u,v,w)\\ q_3(u,v,w) \end{bmatrix} = \begin{bmatrix} 0\\ 0\\ 0 \end{bmatrix}, \tag{2.15}$$

where

$$\begin{aligned}
q_1 &= -\delta_2 v(x,t) w(x,\ t) + \delta_3 \left(u^2(x,t) + v^2(x,t)\right) v(x,t)\,,\\
q_2 &= -\delta_2 u(x,t) w(x,t) + \delta_3 \left(u^2(x,t) + v^2(x,t)\right) u(x,t)\,,\\
q_3 &= -2\delta_4 \left(u(x,t)\partial_{xx}\, u(x,t) + (\partial_x\, u(x,t))^2 + v(x,t)\partial_{xx}\, v(x,t) + (\partial_x\, v(x,t))^2\right),
\end{aligned}$$

the constants δ_1, δ_2, δ_3, δ_4 and c_s are given, and the functions $\alpha_i(t)$, $1 \le i \le 6$ and $\beta_j(x)$, $1 \le j \le 4$ are known.

In the next section, we will use the Chebyshev pseudo-spectral method for discretizing (2.15) subject to the initial and boundary conditions (2.14) to reduce (2.15) to a system of nonlinear algebraic equations.

2.4.2 The Chebyshev pseudo-spectral method

Spectral methods are the best methods for approximating the solutions of problems in applied mathematics and engineering when the solutions are smooth and their domains are simple. Many researchers have used those methods for the numerical solution of nonlinear PDEs [85], fractional ODEs [86], high-order boundary value problems [87], systems of Volterra integral equations [88], optimal control problems governed by Volterra integral equations [89], Quasi Bang-Bang optimal control problems [90], and ODEs of degenerate types [91]. In relation to many other methods, spectral methods give highly accurate results.

To discretize (2.15) subject to the initial and boundary conditions (2.14) using the Chebyshev pseudo-spectral method, we define the following transformations

$$\begin{aligned} y &= \frac{2}{b_x - a_x}\,x - \frac{a_x + b_x}{b_x - a_x}\,, \\ \tau &= \frac{2}{b_t - a_t}\,t - \frac{a_t + b_t}{b_t - a_t}\,, \end{aligned}$$

where $(y,\ \tau) \in [-1,\ 1] \times [-1,\ 1]$. The partial derivatives with respect to the variables associated with the new domain are related to the partial derivatives with respect to the variables associated with the previous domain as

$$\begin{aligned} \partial_x &= \left(\frac{2}{b_x - a_x}\right)\partial_y\,, \quad \partial_{xx} = \left(\frac{2}{b_x - a_x}\right)^2\partial_{yy}\,, \\ \partial_t &= \left(\frac{2}{b_t - a_t}\right)\partial_\tau\,, \quad \partial_{tt} = \left(\frac{2}{b_t - a_t}\right)^2\partial_{\tau\tau}\,. \end{aligned}$$

Let n_x and n_t be the number of grid points in, respectively, the spatial and temporal domains associated with the variables y and τ. The partitions of $[-1,\ 1]$ in the space and time directions are performed using Chebyshev-Gauss-Lobatto

(CGL) points. The number of grid points is $n = n_x n_t$. Let

$$\begin{aligned}
\mathbf{U} &= \left[u_{1,1}, u_{1,2}, \cdots, u_{1,n_t}, u_{2,1}, u_{2,2}, \cdots, u_{2,n_t}, \cdots, u_{n_x,1}, u_{n_x,2}, \cdots, u_{n_x,n_t}\right]^T, \\
\mathbf{V} &= \left[v_{1,1}, v_{1,2}, \cdots, v_{1,n_t}, v_{2,1}, v_{2,2}, \cdots, v_{2,n_t}, \cdots, v_{n_x,1}, u_{n_x,2}, \cdots, v_{n_x,n_t}\right]^T, \\
\mathbf{W} &= \left[w_{1,1}, w_{1,2}, \cdots, w_{1,n_t}, w_{2,1}, w_{2,2}, \cdots, w_{2,n_t}, \cdots, w_{n_x,1}, w_{n_x,2}, \cdots, w_{n_x,n_t}\right]^T
\end{aligned}$$

be vectors collecting the values of the functions $u(y,\tau)$, $v(y,\tau)$ and $w(y,\tau)$ at the grid points. The discrete approximations for the partial derivatives are

$$\begin{aligned}
\partial_y &\approx \mathbf{D}_y \otimes \mathbf{I}_\tau, \quad \partial_{yy} \approx \mathbf{D}_y^2 \otimes \mathbf{I}_\tau, \\
\partial_\tau &\approx \mathbf{I}_y \otimes \mathbf{D}_\tau, \quad \partial_{\tau\tau} \approx \mathbf{I}_y \otimes \mathbf{D}_\tau^2,
\end{aligned} \tag{2.16}$$

where $\mathbf{D}_y$, $\mathbf{D}_\tau$, $\mathbf{I}_y$, $\mathbf{I}_\tau$ are, respectively, the Chebyshev differentiation and identity matrices for variables y and τ, the dimensions for subscripts y and τ are, respectively, n_x and n_t, and $\otimes$ denotes the Kronecker product. Finally, we define the partial derivative operators as

$$\begin{aligned}
\mathbf{A}_x &= \frac{2}{b_x - a_x}\mathbf{D}_y \otimes \mathbf{I}_\tau, \quad \mathbf{A}_{xx} = \frac{2}{b_x - a_x}\mathbf{D}_y^2 \otimes \mathbf{I}_\tau, \\
\mathbf{A}_t &= \frac{2}{b_t - a_t}\mathbf{I}_y \otimes \mathbf{D}_\tau, \quad \mathbf{A}_{tt} = \frac{2}{b_t - a_t}\mathbf{I}_y \otimes \mathbf{D}_\tau^2.
\end{aligned}$$

The discrete form of (2.15) is, using $\mathbf{x}_{m\times n}$ to denote an $m \times n$ matrix $\mathbf{x}$,

$$\begin{bmatrix} \mathbf{A}_t & \delta_1\mathbf{A}_{xx} & \mathbf{O} \\ \delta_1\mathbf{A}_{xx} & -\mathbf{A}_t & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{A}_{tt} - c_s^2\mathbf{A}_{xx} \end{bmatrix} \begin{bmatrix} \mathbf{U} \\ \mathbf{V} \\ \mathbf{W} \end{bmatrix} + \begin{bmatrix} \mathbf{Q}_1(\mathbf{U},\mathbf{V},\mathbf{W}) \\ \mathbf{Q}_2(\mathbf{U},\mathbf{V},\mathbf{W}) \\ \mathbf{Q}_3(\mathbf{U},\mathbf{V},\mathbf{W}) \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{n\times 1} \\ \mathbf{0}_{n\times 1} \\ \mathbf{0}_{n\times 1} \end{bmatrix}, \tag{2.17}$$

where

$$\begin{aligned}
\mathbf{Q}_1 &= -\delta_2\mathbf{V} \odot \mathbf{W} + \delta_3\left(\mathbf{U} \odot \mathbf{U} + \mathbf{V} \odot \mathbf{V}\right) \odot \mathbf{V}, \\
\mathbf{Q}_2 &= -\delta_2\mathbf{U} \odot \mathbf{W} + \delta_3\left(\mathbf{U} \odot \mathbf{U} + \mathbf{V} \odot \mathbf{V}\right) \odot \mathbf{U}, \\
\mathbf{Q}_3 &= -2\delta_4\Big(\mathbf{U} \odot (\mathbf{A}_{xx}\mathbf{U}) + (\mathbf{A}_x\mathbf{U}) \odot (\mathbf{A}_x\mathbf{U}) + \mathbf{V} \odot (\mathbf{A}_{xx}\mathbf{V}) + (\mathbf{A}_x\mathbf{V}) \odot (\mathbf{A}_x\mathbf{V})\Big),
\end{aligned}$$

with $\odot$ denoting the point-wise multiplication between vectors.

The compact form of (2.17) is

$$\mathbf{F}(\mathbf{S}) \equiv \mathbf{A}\mathbf{S} + \mathbf{Q} - \mathbf{B} = \mathbf{0}, \tag{2.18}$$

where

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_t & \delta_1\mathbf{A}_{xx} & \mathbf{O} \\ \delta_1\mathbf{A}_{xx} & -\mathbf{A}_t & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{A}_{tt} - c_s^2\mathbf{A}_{xx} \end{bmatrix}_{3n\times 3n}, \mathbf{S} = \begin{bmatrix} \mathbf{U} \\ \mathbf{V} \\ \mathbf{W} \end{bmatrix}_{3n\times 1},$$

$$\mathbf{Q} = \begin{bmatrix} \mathbf{Q}_1(\mathbf{U},\mathbf{V},\mathbf{W}) \\ \mathbf{Q}_2(\mathbf{U},\mathbf{V},\mathbf{W}) \\ \mathbf{Q}_3(\mathbf{U},\mathbf{V},\mathbf{W}) \end{bmatrix}_{3n\times 1}, \mathbf{B} = \begin{bmatrix} \mathbf{0}_{n\times 1} \\ \mathbf{0}_{n\times 1} \\ \mathbf{0}_{n\times 1} \end{bmatrix}_{3n\times 1}.$$

Our aim is to solve the nonlinear system of algebraic equations (2.18) by the proposed new multi-step ATC method presented in Section 2.2. We have to adapt the structure of $\mathbf{F}(\mathbf{S}) = \mathbf{0}$ to the initial and boundary conditions. Using Matlab-like notation, the initial and boundary conditions can be written as

Initial conditions

```
for  $i = 1 : n_x$
    $indx_1 = (i-1)n_t + 1$,    $indx_2 = (i-1)n_t + 2$,
    $A(indx_1, :) = 0$,    $A(n + indx_1, :) = 0$,
    $A(indx_1, 1:n) = D_1(i, :)$,    $A(n + indx_1, n+1:2n) = D_1(i, :)$,
    $A(2n + indx_1, :) = 0$,    $A(2n + indx_2, :) = 0$,
    $A(2n + indx_1, 2n+1:3n) = D_1(i, :)$,    $A(2n + indx_2, 2n+1:3n) = D_2(i, :)$,
    $B(indx_1) = \beta_1(i)$,    $B(n + indx_1) = \beta_2(i)$,
    $B(2n + indx_1) = \beta_3(i)$,    $B(2n + indx_2) = \beta_4(i)$,
end
```
(2.19)

where $\mathbf{D}_1 = \mathbf{I}_x \otimes \mathbf{I}_t(1,:)$ and $\mathbf{D}_2 = \mathbf{I}_x \otimes \left(\frac{2}{b_t - a_t}\right)\mathbf{D}_\tau$, and

Boundary conditions

$$\begin{aligned}
&A(1:n_t, :) = 0\,, && B = 0\,, \\
&A(1:n_t, 1:n_t) = I_t\,, && B(1:n_t) = \alpha_1(1:n_t)\,, \\
&A(n-n_t+1:n, :) = 0\,, && B(n-n_t+1:n) = \alpha_2(1:n_t)\,, \\
&A(n-n_t+1:n, n-n_t+1:n) = I_t\,, && B(n+1:n+n_t) = \alpha_3(1:n_t)\,, \\
&A(n+1:n+n_t, :) = 0\,, && B(2n-n_t+1:2n) = \alpha_4(1:n_t)\,, \\
&A(n+1:n+n_t, n+1:n+n_t) = I_t\,, && B(2n+1:2n+n_t) = \alpha_5(1:n_t)\,, \\
&A(2n-n_t+1:2n, :) = 0\,, && B(3n-n_t+1:3n) = \alpha_6(1:n_t)\,, \\
&A(2n-n_t+1:2n, 2n-n_t+1:2n) = I_t\,, && A(2n+1:2n+n_t, :) = 0\,, \\
&A(2n+1:2n+n_t, 2n+1:2n+n_t) = I_t\,, && A(3n-n_t+1:3n, :) = 0\,, \\
&A(3n-n_t+1:3n, 3n-n_t+1:3n) = I_t\,,
\end{aligned} \tag{2.20}$$

Finally the rows of $\mathbf{Q}$ and Jacobian of $\mathbf{Q}$ get zeros where $\mathbf{B}$ gets values from initial and boundary conditions. After these modifications, nonlinear system of algebraic equations will be updated and can be solved by any iterative methods such as ATC or NR.

2.5 Numerical analysis

In this section we show the accuracy and performance of the multi-step iterative method ATC when used to solve a system of nonlinear equations obtained by using the Chebyshev pseudo-spectral method to discretize the nonlinear complex generalized Zakharov system (GZS) of partial differential equations. In [78], two test problems concerning GZS have been solved with good accuracy by using the Jacobi pseudo-spectral collocation method. As a comparison we will solve the same two test problems with higher accuracies than those reported in [78]. The errors will be computed using the $||\cdot||_\infty$ norm over the entire grid as

$$\begin{aligned} E_u &= \max_{(x,t)\in\Lambda} |u(x,t) - u_{\text{num}}(x,t)| \ , \\ E_v &= \max_{(x,t)\in\Lambda} |v(x,t) - v_{\text{num}}(x,t)| \ , \\ E_w &= \max_{(x,t)\in\Lambda} |w(x,t) - w_{\text{num}}(x,t)| \ , \end{aligned} \tag{2.21}$$

where Λ is the grid of values for (x,t) used in the discretization, and $u_{\text{num}}(x,t)$, $v_{\text{num}}(x,t)$ and $w_{\text{num}}(x,t)$ are the computed numerical values of the functions $u(x,t)$, $v(x,t)$ and $w(x,t)$. In all computations, the initial guesses for $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{W}$ will be taken equal to the zero vector $\mathbf{O}_{n\times 1}$.

2.5.1 Complex Zakharov equation

The first test problem is the GZS [78]:

$$\begin{aligned} & i\,\partial_t\,\psi - \partial_{xx}\,\psi - \psi\,w = 0 \\ & \partial_{tt}\,w - \partial_{xx}\,w + \partial_{xx}\,|\psi|^2 = 0\,, \end{aligned} \tag{2.22}$$

with domain $\Lambda = [-1,\ 1] \times [0, 3.3]$. That GZS has the analytical solution

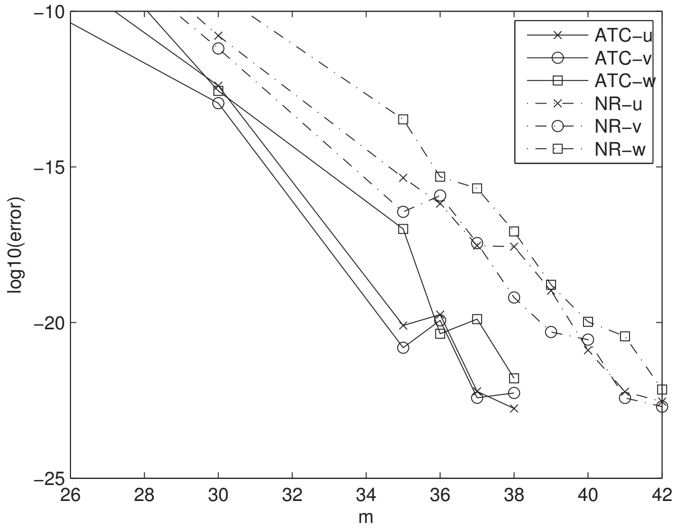


FIGURE 2.1: Errors under ATC and NR for the first test problem.

$$\begin{aligned}\psi(x,\ t) &= u(x,\ t) + i\,v(x,\ t) = \sqrt{3}\,e^{i(x+t)}\tanh\left(\frac{1}{\sqrt{2}}(x+2t)\right)\\ w(x,\ t) &= 1-\tanh^2\left(\frac{1}{\sqrt{2}}(x+2t)\right).\end{aligned} \tag{2.23}$$

We choose the parameter $\theta = 1.0 - 0.001\,\text{rand}(3n, 1)$, where $\text{rand}(3n, 1)$ is a uniform random vector of dimension $3n$ in the interval $[0, 1]$ for each component. The role of the parameter θ is important because θ may affect the actual speed of convergence and the convergence radius. We solved the complex Zakharov equation in the domain $[-1, 1] \times [0, 3.3]$ with $n_x = 23$ grid points for the space dimension and $n_t = 48$ grid points for the time dimension. Table 2.1 compares the errors obtained by ATC and the NR multi-step method as a function of the number of steps m. The table also gives the execution time of ATC for $m = 38$ and NR for $m = 42$, numbers of steps under which the errors in both methods are sufficiently small and similar. We can note that for the same number of steps the errors under ATC are significantly smaller than under NR, particularly when the number of steps m becomes large. With similar error targets, the ATC method is about 7 % faster than the NR method. Figure 2.1 shows the errors in u, v and w in logarithmic scale against the number of steps m for methods ATC and NT. Figures 2.2, 2.3 and 2.4 plot, respectively, $u(x, t)$, $v(x, t)$ and $w(x, t)$ at the grid points.

	ATC method			NR method		
	Execution time for $m = 38 = 49.285$ s			Execution time for $m = 42 = 53.096$ s		
m	E_u	E_v	E_w	E_u	E_v	E_w
1	0.47454	0.49695	0.85826	0.47463	0.49733	0.85754
2	1.1371	1.2102	3.72	1.1361	1.21	2.8038
3	4.9923	3.019	3.7464	5.0242	3.0273	3.2173
4	3.2748	5.5037	5.1003	3.2919	5.5702	5.1487
5	3.3246	2.1363	18.102	3.815	2.1929	18.948
10	1.0611	1.8962	3.0289	2.0823	4.5914	8.0358
15	0.047661	0.12338	0.53352	0.31678	0.5274	5.7334
20	0.081501	0.020402	0.0076958	1.0876	0.76647	0.33485
25	0.00029604	5.9708e-05	0.0027184	0.0040615	0.0019482	0.028504
30	4.143e-06	2.3705e-06	3.5209e-06	2.0652e-05	1.3753e-05	8.1167e-05
35	1.8722e-09	9.1849e-10	4.1729e-08	2.1499e-07	7.2315e-08	1.4055e-06
36	2.6555e-09	2.2107e-09	1.4384e-09	9.4002e-08	1.2213e-07	2.2354e-07
37	2.2531e-10	1.8415e-10	2.307e-09	2.4263e-08	2.6427e-08	1.5319e-07
38	1.3032e-10	2.1521e-10	3.4423e-10	2.3561e-08	4.6183e-09	3.8451e-08
39				5.74e-09	1.5185e-09	6.9447e-09
40				8.4909e-10	1.1898e-09	2.1135e-09
41				2.2375e-10	1.8279e-10	1.3227e-09
42				1.6263e-10	1.3847e-10	2.3933e-10

TABLE 2.1: Performance comparison of ATC and NR for the first test problem.

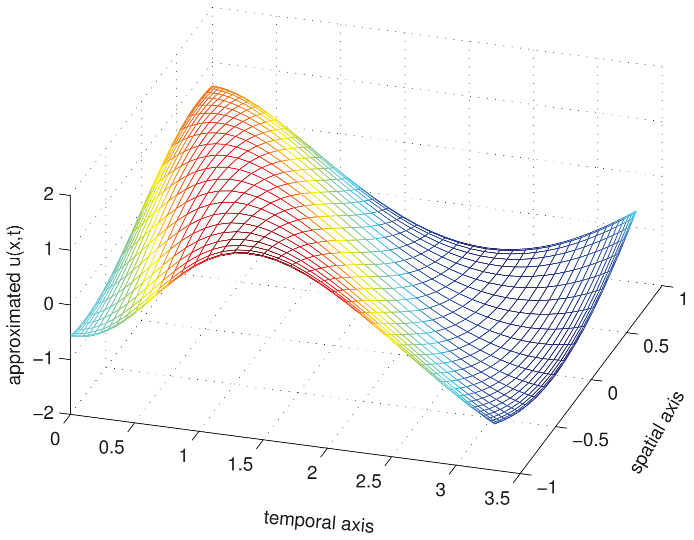


FIGURE 2.2: Computed $u(x,t)$ for the first test problem.

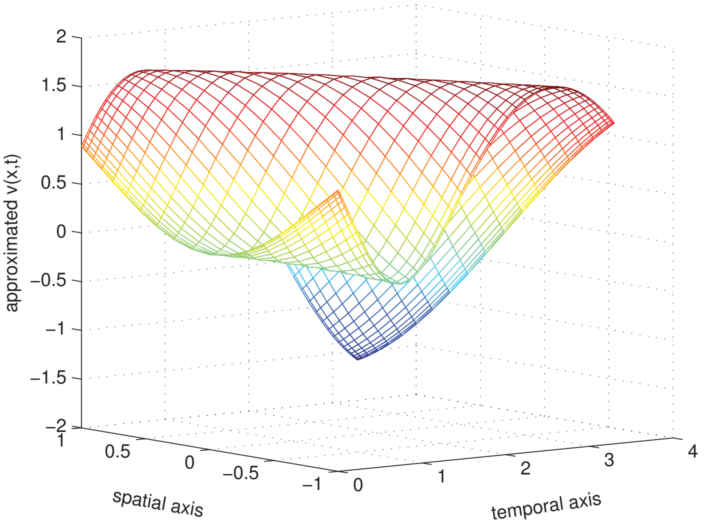


Figure 2.3: Computed $v(x,t)$ for the first test problem.

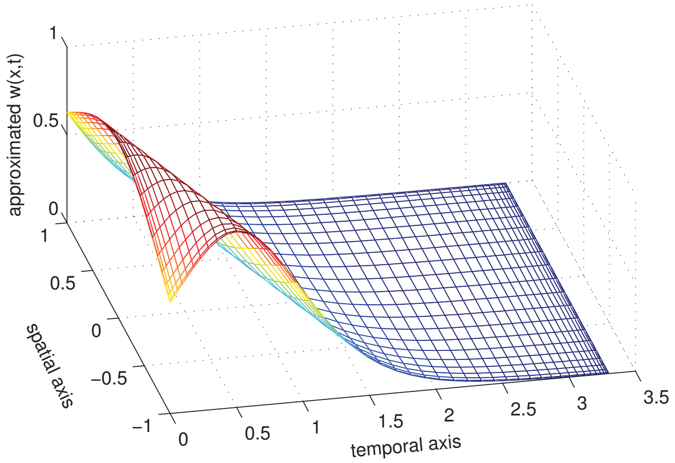


Figure 2.4: Computed $w(x,t)$ for the first test problem.

2.5.2 Complex generalized Zakharov equation

The second test problem we consider is the GZS in complex form, which is [78]:

$$\begin{aligned} &i\partial_t\psi + \partial_{xx}\psi + 2\psi w - 2|\psi|^2 = 0 \\ &\partial_{tt}w - \partial_{xx}w + \partial_{xx}|\psi|^2 = 0, \end{aligned} \tag{2.24}$$

with the domains $\Lambda = [-1,\ 1] \times [0, 1.2]$ and $\Lambda = [-1,\ 1] \times [0, 1.3]$. That problem has the analytical solution

$$\begin{aligned} \psi(x,\ t) &= u(x,\ t) + iv(x,\ t) = \frac{\sqrt{3}}{2} e^{-i(x+3t)} \tanh(x+2t) \\ w(x,\ t) &= -\frac{1}{4} \tanh^2(x+2t)\,. \end{aligned} \tag{2.25}$$

We will consider several numbers of grid points in the space dimension, n_x, and in the time dimension, n_t. We will also consider several values for θ. Table 2.2 compares the performance of ATC and NR for $\Lambda = [-1, 1] \times [0, 1.2]$, $n_x = 32$, $n_t = 28$, and $\theta = 1.3$, as a function of the number of steps. We can note that, for the same number of steps ATC yields smaller errors than NR. With similar errors, the execution time of ATC for $m = 21$ is smaller than the execution time of NR for $m = 14$. When we integrate the complex generalized Zakharov equation for $\Lambda = [-1, 1] \times [0, 1.3]$, the NR method shows divergence. This is illustrated in Table ?? which compares the behavior of ATC and NR for $\Lambda = [-1, 1] \times [0, 1.3]$, $n_x = 21$, $n_t = 34$, and $\theta = 2$. Therefore, an appropriate selection for the parameter θ increases the convergence radius of ATC in comparison with NR. Figure 2.5 plots the errors in $u(x, t)$, $v(x, t)$ and $w(x, t)$ as a function of the number of steps m under ATC and NR for $\Lambda = [-1, 1] \times [0, 1.2]$, $n_x = 32$, $n_t = 28$, and $\theta = 1.3$. We can note that, for the same number of steps the errors under ATC are smaller than under NR. Figure 2.6 gives the absolute errors in u, v, w at the different grid points for $m = 24$ under ATC for $\Lambda = [-1, 1] \times [0, 1.3]$, $n_x = 21$, $n_t = 34$, and $\theta = 2$. Figures 2.7, 2.8 and 2.9 plot, respectively, $u(x, t)$, $v(x, t)$, $w(x, t)$ and the corresponding absolute errors under ATC for $m = 24$, $\Lambda = [-1, 1] \times [0, 1.3]$, $n_x = 21$, $n_t = 34$, and $\theta = 2$.

The results obtained in [78] are presented in Table 2.4. We used more number of grid points in spatial dimension than that of [78] and got better numerical accuracy in numerical results.

2.6 Summary

Multi-step iterative methods for solving nonlinear systems tend to be computationally economical. The ATC method makes only one Jacobian evaluation.

	ATC method			NR method		
	Execution time for $m = 21 = 11.891$			Execution time for $m = 24$ =12.953		
m	E_u	E_v	E_w	E_u	E_v	E_w
1	0.45709	0.8603	0.21256	0.48046	1.1721	0.23063
2	1.1231	0.77984	0.64226	1.3562	1.2781	0.64226
3	0.96367	1.6933	0.40012	2.0621	2.5231	0.58512
4	1.5751	0.40543	0.32973	3.1627	1.6789	0.46649
5	0.20387	0.2434	0.095349	0.56035	0.64707	0.14493
6	0.062707	0.066969	0.039415	0.03373	0.14675	0.0401
7	0.045025	0.025539	0.011057	0.060376	0.020351	0.016544
8	0.0033644	0.0095145	0.0027489	0.0050918	0.016299	0.0047009
9	0.0036137	0.0015154	0.00086978	0.005709	0.0059042	0.0024091
10	0.00027201	0.00079506	0.00012986	0.0051656	0.0029367	0.0021463
11	0.00020245	7.6515e-05	5.1606e-05	0.0018987	0.0026428	0.00085315
12	3.5057e-05	4.5131e-05	1.0796e-05	0.0013403	0.00052468	0.0004666
13	7.0703e-06	8.5718e-06	2.8361e-06	0.00045672	0.00042808	0.00013906
14	2.6527e-06	1.962e-06	6.4814e-07	0.00011958	9.8066e-05	4.5534e-05
15	3.2703e-07	5.7061e-07	8.6105e-08	4.9319e-05	3.0739e-05	1.0606e-05
16	1.3652e-07	1.1004e-07	4.3605e-08	4.7169e-06	8.9355e-06	2.271e-06
17	1.4762e-08	2.3321e-08	4.677e-09	2.7754e-06	1.2644e-06	5.2229e-07
18	4.0657e-09	4.2216e-09	1.513e-09	1.3475e-07	5.0502e-07	7.0055e-08
19	5.6959e-10	6.0451e-10	3.4595e-10	1.0105e-07	4.7431e-08	2.7238e-08
20	2.8374e-10	2.833e-10	9.2313e-11	1.3424e-08	1.8865e-08	4.1215e-09
21	8.5017e-11	7.0587e-11	7.6073e-11	2e-09	2.5181e-09	9.5584e-10
22				6.8728e-10	3.6522e-10	2.2399e-10
23				6.0788e-11	1.1403e-10	6.8574e-11
24				3.1735e-11	1.5148e-11	7.1093e-11

Table 2.2: Performance comparison of ATC and NR for the second test problem with domain $\Lambda = [-1, 1] \times [0, 1.2]$, $n_x = 32$, $n_t = 28$ and $\theta = 1.3$.

ATC method				NR method			
Execution time for $m = 24 = 16.259$				Execution time for $m = 24$ =8.588			
m	E_u	E_v	E_w	m	E_u	E_v	E_w
1	2.3489	1.8274	1 0.49851	1	0.86782	1.1797	0.23024
2	1.1081	0.54759	0.96708	2	1.5424	1.9725	0.96708
7	0.10085	0.067689	0.029542	3	5.615	3.8987	0.95367
13	6.2449e-05	7.1607e-05	1.8598e-05	4	15.611	20.172	1.5002
17	2.4756e-07	3.291e-07	7.4927e-08	5	348.08	205.49	3.8896
19	1.9145e-08	2.1149e-08	5.7041e-09	6	2.355e+05	5.6693e+05	9506.1
22	5.151e-10	6.2691e-10	1.1519e-10	7	7.8999e+14	2.5429e+14	1.1504e+12
23	2.9278e-10	1.2219e-10	9.2533e-11	8	3.528e+42	5.2674e+42	1.3026e+31
24	5.3785e-11	6.9398e-11	7.4369e-11	9	1.8566e+127	1.1924e+127	2.1703e+86

Table 2.3: Performance comparison of ATC and NR for second test problem with domain $\Lambda = [-1,\ 1] \times [0, 1.3]$, $n_x = 21$, $n_t = 34$ and $\theta = 2$.

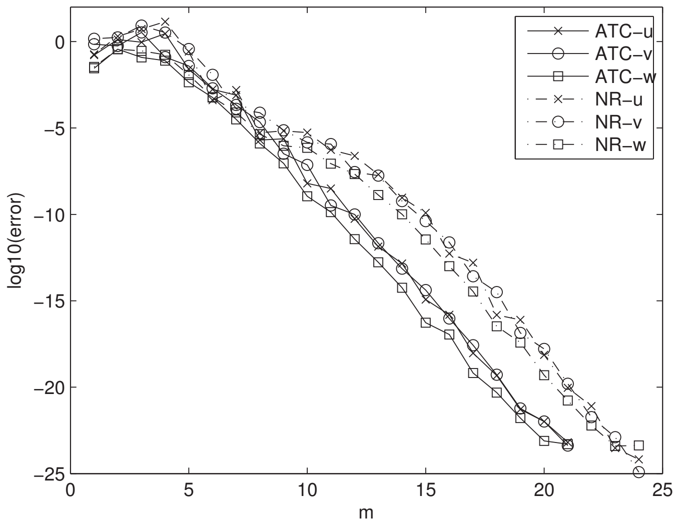


FIGURE 2.5: Errors in u, v, w for ATC and NR as a function on the number of steps for $\Lambda = [-1,\ 1] \times [0, 1.2]$, $n_t = 28$, $n_x = 32$, and $\theta = 1.3$

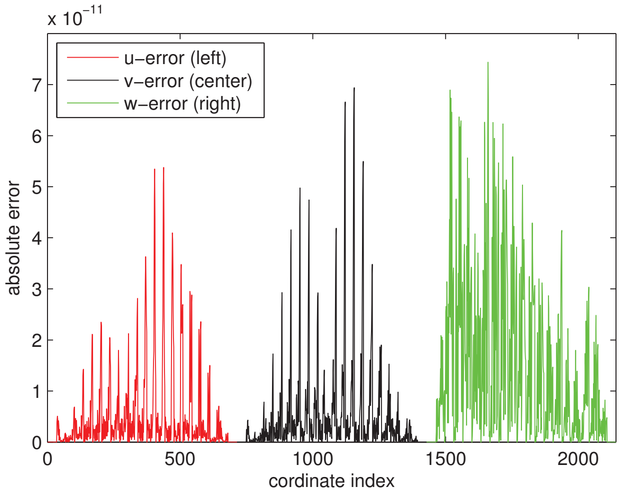


FIGURE 2.6: Absolute errors in u, v, w at the grid points for the second test problem under ATC for $m = 24$, $\Lambda = [-1,\ 1] \times [0, 1.3]$, $n_x = 21$, $n_t = 34$, and $\theta = 2$.

n_x	E_u	E_v	E_w
4	4.43e-2	7.12e-2	4.53e-2
8	2.13e-4	1.624e-4	1.20e-4
12	8.34e-7	6.02e-7	2.54e-7
16	3.83e-7	3.4e-7	1.51e-8

TABLE 2.4: Performance of method presented in [78] for the first test problem, $\Lambda = [-1,\ 1] \times [0, 1]$, $n_x = 4, 8, 12, 16$.

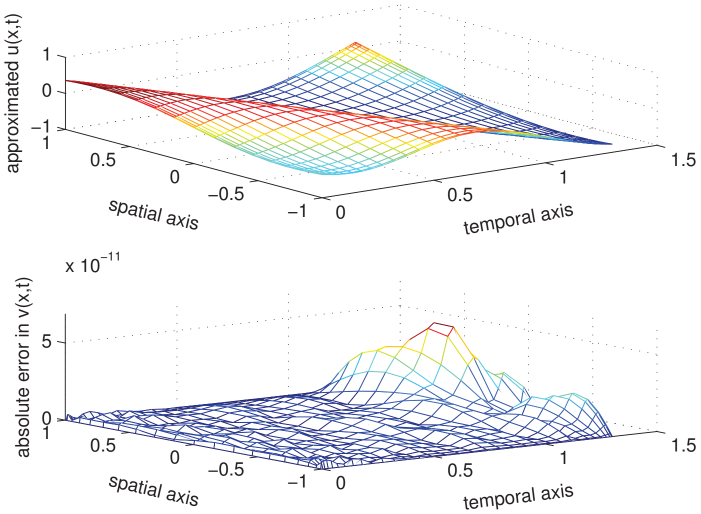


FIGURE 2.7: $u(x,t)$ and corresponding absolute errors under ATC for $m = 14$, $\Lambda = [-1,\ 1] \times [0, 1.3]$, $n_x = 21$, $n_t = 34$, and $\theta = 2$

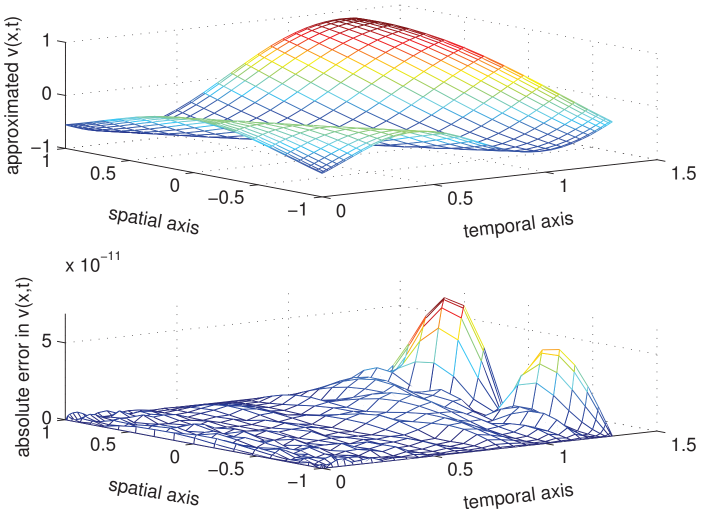


FIGURE 2.8: $v(x,t)$ and corresponding absolute errors under ATC for $m = 14$, $\Lambda = [-1,\ 1] \times [0, 1.3]$, $n_x = 21$, $n_t = 34$, and $\theta = 2$

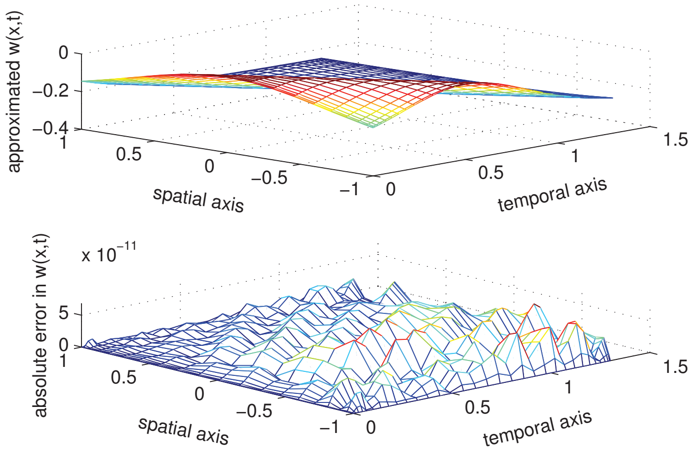


FIGURE 2.9: $w(x,t)$ and corresponding absolute errors under ATC for $m = 14$, $\Lambda = [-1,\ 1] \times [0, 1.3]$, $n_x = 21$, $n_t = 34$, and $\theta = 2$

Once the LU-factors of the Jacobian are evaluated, they are used in the multistep part to make the method computationally efficient. Our numerical results clearly show that ATC has better speed of convergence than NR and, with an appropriate selection of θ, wider radius of convergence. Applied to the complex generalized Zakharov equation with using the Chebyshev pseudo-spectral method for discretization, the ATC gives more accurate numerical solutions than they have been obtained in [78].

Chapter 3

Multi-step preconditioned Newton methods for solving systems of nonlinear equations

The study of different forms of preconditioners for solving a system of nonlinear equations, by using Newton's method, is presented. The preconditioners provide numerical stability and rapid convergence with reasonable computation cost, whenever chosen accurately. Various families of iterative methods can be constructed by using a different kind of preconditioners. The multi-step iterative method consists of a base method and multi-step part. The convergence order of the base method is quadratic and each multi-step provides an additional factor of one in the previously achieved convergence order. Hence the convergence of order of an m-step iterative method is $m + 1$. Numerical simulations confirm the claimed convergence order by calculating the computational order of convergence. Finally, the numerical results clearly show the benefit of preconditioning for solving systems of nonlinear equations.

3.1 Introduction

When computing simple roots of a system of nonlinear equations, Newton's method [1, 3, 7, 8] is a classical, well studied procedure that offers quadratic convergence, under suitably mild regularity assumptions. Many researchers have proposed higher order efficient iterative method [38, 57, 62, 102–106] for solving system of nonlinear equations. Recently, some authors have constructed multi-step iterative methods [70, 107, 108] for solving system of nonlinear equations. The main benefit of multi-step iterative methods is hidden in the multi-step part. Because, the Jacobian factorization information from the base method is utilized in the multi-step part repeatedly to enhance the convergence order at the cost of the solution of lower and upper triangular systems and a single evaluation of the system of nonlinear equations. X. Wu [109] wrote a note on the improvement of Newton's method for systems of nonlinear equations. In his note, the author introduced the idea of nonlinear preconditioners and showed that the improved Newton method enjoyed the quadratic convergence. Jose et al. [110] used the idea of nonlinear preconditioning to improve the Newton method, for solving the system of nonlinear equations with known multiplicities. Aslam et al. [111] proposed iterative methods for solving nonlinear equations with unknown multiplicity with the help of nonlinear preconditioners. In the another article Aslam and his co-researcher [112] proposed a preconditioned double Newton method with quartic convergence order for the solving system of nonlinear equations. What they have proposed is the following. Let

$$\mathbf{F}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_n(\mathbf{x})]^T = \mathbf{0} \tag{3.1}$$

be the system of nonlinear equations and let us suppose that only simple roots are present. Here $\mathbf{x} = [x_1, x_2, \cdots, x_n]^T$. Assume $\mathbf{G}(\mathbf{x}) = [g_1(\mathbf{x}), g_2(\mathbf{x}), \cdots, g_n(\mathbf{x})]^T$ is a function which is non-zero everywhere in its definition domain. We define a new function

$$\mathbf{Q}(\mathbf{x}) = \mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}) = [\![\mathbf{G}(\mathbf{x})]\!]\,\mathbf{F}(\mathbf{x}) = [\![\mathbf{F}(\mathbf{x})]\!]\,\mathbf{G}(\mathbf{x}), \tag{3.2}$$

where $\odot$ is the element-wise multiplication and $[\![\cdot]\!]$ represent the diagonal matrix, having as main diagonal its argument. The first order Fréchet derivative of (3.2) can be computed as

$$\begin{aligned}\mathbf{Q}'(\mathbf{x}) &= [\![\mathbf{F}(\mathbf{x})]\!]\,\mathbf{G}'(\mathbf{x}) + [\![\mathbf{G}(\mathbf{x})]\!]\,\mathbf{F}'(\mathbf{x}) \\ &= [\![\mathbf{G}(\mathbf{x})]\!]\left(\mathbf{F}'(\mathbf{x}) + [\![\mathbf{F}(\mathbf{x})]\!]\,[\![\mathbf{G}(\mathbf{x})]\!]^{-1}\,\mathbf{G}'(\mathbf{x})\right).\end{aligned} \tag{3.3}$$

The application of Newton method to (3.2) gives

$$\begin{aligned}\mathbf{x}_{k+1} &= \mathbf{x}_k - \mathbf{Q}'(\mathbf{x}_k)^{-1}\,\mathbf{Q}(\mathbf{x}_k)\\ &= \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}) + [\![\mathbf{F}(\mathbf{x}_k)]\!]\,[\![\mathbf{G}(\mathbf{x}_k)]\!]^{-1}\,\mathbf{G}'(\mathbf{x}_k)\right)^{-1}\mathbf{F}(\mathbf{x}_k).\end{aligned} \tag{3.4}$$

The convergence order of (3.4) is quadratic, because the considered scheme is the Newton method for solving the preconditioned system of nonlinear equations $\mathbf{Q}(\mathbf{x}) = \mathbf{0}$. If we take $\mathbf{G}(\mathbf{x}) = exp(\boldsymbol{\beta} \odot \mathbf{x})$ then (3.4) can be written as

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}) + [\![\boldsymbol{\beta} \odot \mathbf{F}(\mathbf{x})]\!]\right)^{-1}\mathbf{F}(\mathbf{x}), \tag{3.5}$$

where $\boldsymbol{\beta} = [\beta_1, \beta_2, \cdots, \beta_n]^T$.

3.2 Our proposal

In this section, we develop some preconditioned iterative methods for solving systems of nonlinear equations. We generalize the idea of preconditioning in such a way that the quadratic convergence will be guaranteed, under the usual regularity requirements. If we replace $\mathbf{G}(\mathbf{x})$ by $\exp((\mathbf{G}(\mathbf{x}))$ in (3.4), then we obtain

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \Big(\mathbf{F}'(\mathbf{x}_k) + [\![\mathbf{F}(\mathbf{x}_k)]\!]\,\mathbf{G}'(\mathbf{x}_k)\Big)^{-1}\mathbf{F}(\mathbf{x}_k). \tag{3.6}$$

Notice that $\mathbf{G}'(\mathbf{x})$ is a matrix that could be a diagonal matrix, but also a generic dense matrix. We proposed the following generalization of (3.6)

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \Big(\mathbf{F}'(\mathbf{x}_k) + M_1(\mathbf{x}_k)\,[\![\mathbf{F}(\mathbf{x}_k)]\!]\,M_2(\mathbf{x}_k)\Big)^{-1}\mathbf{F}(\mathbf{x}_k), \tag{3.7}$$

where $M_1(\mathbf{x})$ and $M_2(\mathbf{x})$ are matrices of size n. In the next development, we see that $[\![\mathbf{F}(\mathbf{x})]\!]$ is not the only option. Let $p(\mathbf{x})$ be a scalar function and let us define the following preconditioned system of nonlinear equations

$$\mathbf{Q}(\mathbf{x}) = p(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}) = [p(\mathbf{x})\,f_1(\mathbf{x}), p(\mathbf{x})\,f_2(\mathbf{x}), \cdots, p(\mathbf{x})\,f_n(\mathbf{x})]^T. \tag{3.8}$$

The first order Fréchet derivative of (3.8) can be computed as

$$\begin{aligned}\mathbf{Q}'(\mathbf{x}) &= p(\mathbf{x}) \odot \mathbf{F}'(\mathbf{x}) + \mathbf{F}(\mathbf{x})\,\nabla p(\mathbf{x})^T \\ \mathbf{Q}'(\mathbf{x}) &= p(\mathbf{x}) \odot \left(\mathbf{F}'(\mathbf{x}) + \mathbf{F}(\mathbf{x})\,\left(\frac{\nabla p(\mathbf{x})^T}{p(\mathbf{x})}\right)\right)\end{aligned} \tag{3.9}$$

where $p(\mathbf{x}) \odot \mathbf{F}'(\mathbf{x})$ means the element-wise product of $p(\mathbf{x})$ with each element of $\mathbf{F}'(\mathbf{x})$ and $/$ in $\left(\frac{\nabla p(\mathbf{x})^T}{p(\mathbf{x})}\right)$ is element-wise division. The application of the Newton method to (3.8) gives

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + \mathbf{F}(\mathbf{x}_k)\,\left(\frac{\nabla p(\mathbf{x}_k)^T}{p(\mathbf{x}_k)}\right)\right)^{-1}\,\mathbf{F}(\mathbf{x}_k). \tag{3.10}$$

Again, if we replace $p(\mathbf{x}_k)$ by $\exp\left(p(\mathbf{x}_k)\right)$ in (3.10), then we have

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + \mathbf{F}(\mathbf{x}_k)\,\nabla p(\mathbf{x}_k)^T\right)^{-1}\,\mathbf{F}(\mathbf{x}_k). \tag{3.11}$$

The following generalization of (3.11) can be obtained

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + \mathbf{F}(\mathbf{x}_k)\,\mathbf{V}(\mathbf{x}_k)^T\right)^{-1}\,\mathbf{F}(\mathbf{x}_k) \quad \text{and} \tag{3.12}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + \mathbf{V}(\mathbf{x}_k)\,\mathbf{F}(\mathbf{x}_k)^T\right)^{-1}\,\mathbf{F}(\mathbf{x}_k), \tag{3.13}$$

where $\mathbf{V}(\mathbf{x}_k) = [v_1(\mathbf{x}_k), v_2(\mathbf{x}_k), \cdots, v_n(\mathbf{x}_k)]^T$. Other possibilities could be

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + M_1(\mathbf{x}_k)\,\mathbf{F}(\mathbf{x}_k)\,\mathbf{V}(\mathbf{x}_k)^T\,M_2(\mathbf{x}_k)\right)^{-1}\,\mathbf{F}(\mathbf{x}_k) \quad \text{and} \tag{3.14}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + M_1(\mathbf{x}_k)\,\mathbf{V}(\mathbf{x}_k)\,\mathbf{F}(\mathbf{x}_k)^T\,M_2(\mathbf{x}_k)\right)^{-1}\,\mathbf{F}(\mathbf{x}_k). \tag{3.15}$$

Further, we write the multi-step version of the proposed generalizations

$$\begin{aligned}&\text{Base method} \longrightarrow \begin{cases}\mathbf{x}_0 = \text{initial guess} \\ \mathbf{A}\,\boldsymbol{\phi}_1 = \mathbf{F}\left(\mathbf{x}_0\right) \\ \mathbf{x}_1 = \mathbf{x}_0 - \boldsymbol{\phi}_1\end{cases} \\ &\text{Multi-step part} \rightarrow \begin{cases}\text{for} \ \ j = 2, m \\ \qquad \mathbf{A}\,\boldsymbol{\phi}_j = \mathbf{F}\left(\mathbf{x}_{j-1}\right) \\ \qquad \mathbf{x}_j = \mathbf{x}_{j-1} - \boldsymbol{\phi}_j \\ \text{end} \\ \mathbf{x}_0 = \mathbf{x}_m\,,\end{cases}\end{aligned} \tag{3.16}$$

where

$$\mathbf{A} = \begin{cases} & \text{Iterative method} \\ \mathbf{F}'(\mathbf{x}) + M_1(\mathbf{x})\, [\![\mathbf{F}(\mathbf{x})]\!]\, M_2(\mathbf{x}) & (3.7) \\ \mathbf{F}'(\mathbf{x}) + \mathbf{F}(\mathbf{x})\, \mathbf{V}(\mathbf{x})^T & (3.12) \\ \mathbf{F}'(\mathbf{x}) + \mathbf{V}(\mathbf{x})\, \mathbf{F}(\mathbf{x})^T & (3.13) \\ \mathbf{F}'(\mathbf{x}) + M_1(\mathbf{x})\, \mathbf{F}(\mathbf{x})\, \mathbf{V}(\mathbf{x})^T\, M_2(\mathbf{x}) & (3.14) \\ \mathbf{F}'(\mathbf{x}) + M_1(\mathbf{x})\, \mathbf{V}(\mathbf{x})\, \mathbf{F}(\mathbf{x})^T\, M_2(\mathbf{x}) & (3.15) \end{cases} .$$

The famous Newton multi-step iterative method [67] can be written as

$$\begin{aligned} \text{Base method} \longrightarrow & \begin{cases} \mathbf{x}_0 = \text{initial guess} \\ \mathbf{F}'(\mathbf{x}_0)\, \boldsymbol{\phi}_1 = \mathbf{F}\,(\mathbf{x}_0) \\ \mathbf{x}_1 = \mathbf{x}_0 - \boldsymbol{\phi}_1 \end{cases} \\ \text{Multi-step part} \rightarrow & \begin{cases} \text{for} \quad j = 2, m \\ \qquad \mathbf{F}'(\mathbf{x}_0)\, \boldsymbol{\phi}_j = \mathbf{F}\,(\mathbf{x}_{j-1}) \\ \qquad \mathbf{x}_j = \mathbf{x}_{j-1} - \boldsymbol{\phi}_j \\ \text{end} \\ \mathbf{x}_0 = \mathbf{x}_m\, . \end{cases} \end{aligned} \tag{3.17}$$

3.3 Convergence Analysis

In this section we give in detail the proofs of convergence order of (3.16) only for $m = 2$, while for the case $m \geq 3$ we use mathematical induction.

Theorem 3.1. *Let $\mathbf{F} : \Gamma \subseteq \mathbb{R}^n \to \mathbb{R}^n$ be sufficiently Frechet differentiable on an open convex neighborhood Γ of $\mathbf{x}^* \in \mathbb{R}^n$ with $\mathbf{F}\,(\mathbf{x}^*) = 0$, $det\,(\mathbf{F}'\,(\mathbf{x}^*)) \neq 0$, and with well defined quantities $||M_1(\mathbf{x}_k)||$, $||M_2(\mathbf{x}_k)||$, $||\mathbf{V}(\mathbf{x}_k)||$. Then the sequence $\{\mathbf{x}_k\}$ generated by (3.16) converges to $\mathbf{x}^*$ with local order of convergence at least three for $m = 2$. Furthermore, the following error inequality*

$$||\mathbf{e}_{k+1}|| \leq ||\mathbf{L}||\, ||\mathbf{e}_k||^3 \tag{3.18}$$

is satisfied, where $\mathbf{e}_k = \boldsymbol{x}_k - \mathbf{x}^*$, $\mathbf{e}_k{}^p = \overbrace{(\mathbf{e}_k, \mathbf{e}_k, \cdots, \mathbf{e}_k)}^{p\ times}$, $\mathbf{e}_k = [(\mathbf{e}_k)_1, (\mathbf{e}_k)_2, \cdots, (\mathbf{e}_k)_n]^T$ *and*

$$||\mathbf{L}|| = \begin{cases} & \textit{Iterative method} \\ (2||\mathbf{C}_2||^2 + 3||\mathbf{C}_2||\,||M_1(\mathbf{x}_0)||\,||M_2(\mathbf{x}_0)|| + (||M_1(\mathbf{x}_0)||\,||M_2(\mathbf{x}_0)||)^2)||\mathbf{e}_0||^3 & (3.7) \\ (2||\mathbf{C}_2||^2 + 3||\mathbf{C}_2||\,||\mathbf{V}(\mathbf{x}_0)|| + ||\mathbf{V}(\mathbf{x}_0)||^2)||\mathbf{e}_0||^3 & (3.12) \\ (2||\mathbf{C}_2||^2 + 3||\mathbf{C}_2||\,||\mathbf{V}(\mathbf{x}_0)|| + ||\mathbf{V}(\mathbf{x}_0)||^2)||\mathbf{e}_0||^3 & (3.13) \\ (2||\mathbf{C}_2||^2 + 3||\mathbf{C}_2||\,||M_1(\mathbf{x}_0)||\,||M_2(\mathbf{x}_0)||\,||\mathbf{V}(\mathbf{x}_0)|| + (||M_1(\mathbf{x}_0)||\,||M_2(\mathbf{x}_0)||\,||\mathbf{V}(\mathbf{x}_0)||)^2)||\mathbf{e}_0||^3 & (3.14) \\ (2||\mathbf{C}_2||^2 + 3||\mathbf{C}_2||\,||M_1(\mathbf{x}_0)||\,||M_2(\mathbf{x}_0)||\,||\mathbf{V}(\mathbf{x}_0)|| + (||M_1(\mathbf{x}_0)||\,||M_2(\mathbf{x}_0)||\,||\mathbf{V}(\mathbf{x}_0)||)^2)||\mathbf{e}_0||^3 & (3.15) \end{cases} . \tag{3.19}$$

Proof. The qth Frechet derivative of $\mathbf{F}$ at $v \in \mathbb{R}^n$, $q \geq 1$, is the $q-linear$ function $\mathbf{F}^{(q)}(v) : \overbrace{\mathbb{R}^n\mathbb{R}^n \cdots \mathbb{R}^n}^{\text{q times}}$ such that $\mathbf{F}^{(q)}(v)(u_1, u_2, \cdots, u_q) \in \mathbb{R}^n$. The Taylor's series expansion of $\mathbf{F}(\mathbf{x}_0)$ around $\mathbf{x}^*$ can be written as

$$\begin{aligned} \mathbf{F}(\mathbf{x}_0) &= \mathbf{F}(\mathbf{x}^* + \mathbf{x}_0 - \mathbf{x}^*) = \mathbf{F}(\mathbf{x}^* + \mathbf{e}_0), \\ &= \mathbf{F}(\mathbf{x}^*) + \mathbf{F}'(\mathbf{x}^*)\mathbf{e}_0 + \frac{1}{2!}\mathbf{F}''(\mathbf{x}^*)\mathbf{e}_0{}^2 + \frac{1}{3!}\mathbf{F}^{(3)}(\mathbf{x}^*)\mathbf{e}_0{}^3 + \cdots, \\ &= \mathbf{F}'(\mathbf{x}^*)\Bigg(\mathbf{e}_0 + \frac{1}{2!}\mathbf{F}'(\mathbf{x}^*)^{-1}\mathbf{F}''(\mathbf{x}^*)\mathbf{e}_0{}^2 \\ &+ \frac{1}{3!}\mathbf{F}'(\mathbf{x}^*)^{-1}\mathbf{F}^{(3)}(\mathbf{x}^*)\mathbf{e}_0{}^3 + \cdots\Bigg), \\ &= \mathbf{C}_1\left(\mathbf{e}_0 + \mathbf{C}_2\,\mathbf{e}_0{}^2 + \mathbf{C}_3\,\mathbf{e}_0{}^3 + O\left(\mathbf{e}_0{}^4\right)\right), \end{aligned} \tag{3.20}$$

where $\mathbf{C}_1 = \mathbf{F}'(\mathbf{x}^*)$ and $\mathbf{C}_s = \frac{1}{s!}\mathbf{F}'(\mathbf{x}^*)^{-1}\mathbf{F}^{(s)}(\mathbf{x}^*)$ for $s \geq 2$. From (3.20), we can calculate the Fréchet derivative of $\mathbf{F}$ as

$$\mathbf{F}'(\mathbf{x}_0) = \mathbf{C}_1\left(\mathbf{I} + 2\mathbf{C}_2\,\mathbf{e}_0 + 3\mathbf{C}_3\,\mathbf{e}_0{}^2 + 4\mathbf{C}_3\,\mathbf{e}_0{}^3 + O\left(\mathbf{e}_0{}^4\right)\right), \tag{3.21}$$

where $\mathbf{I}$ is the identity matrix. Before to proceed further, first we compute the norm bounds for

$$\begin{aligned} \mathbf{B}(\mathbf{x}) \in \Big\{ & M_1(\mathbf{x})\,[\![\mathbf{F}(\mathbf{x})]\!]\,M_2(\mathbf{x}), \mathbf{F}(\mathbf{x})\,\mathbf{V}(\mathbf{x})^T, \mathbf{V}(\mathbf{x})\,\mathbf{F}(\mathbf{x})^T, \\ & M_1(\mathbf{x})\,\mathbf{F}(\mathbf{x})\,\mathbf{V}(\mathbf{x})^T\,M_2(\mathbf{x}), M_1(\mathbf{x})\,\mathbf{V}(\mathbf{x})\,\mathbf{F}(\mathbf{x})^T\,M_2(\mathbf{x})\Big\}. \end{aligned}$$

$$\begin{aligned} \mathbf{B}(\mathbf{x}_0) &= M_1(\mathbf{x}_0)\,[\![\mathbf{F}(\mathbf{x}_0)]\!]\,M_2(\mathbf{x}_0) \\ &= M_1(\mathbf{x}_0)\,[\![\mathbf{C}_1\big(\mathbf{e}_0 + \mathbf{C}_2\mathbf{e}_0^2 + O\big(\mathbf{e}_0^3\big)\big)]\!]\,M_2(\mathbf{x}_0) \end{aligned} \tag{3.22}$$

$$||\mathbf{B}(\mathbf{x}_0)|| \leq ||\mathbf{C}_1||\,||M_1(\mathbf{x}_0)||\,||M_2(\mathbf{x}_0)||\,||\mathbf{e}_0||$$

$$\begin{aligned} \mathbf{B}(\mathbf{x}_0) &= \mathbf{F}(\mathbf{x}_0)\,\mathbf{V}(\mathbf{x}_0)^T \\ &= \mathbf{C}_1\big(\mathbf{e}_0 + \mathbf{C}_2\mathbf{e}_0^2 + O\big(\mathbf{e}_0^3\big)\big)\,\mathbf{V}(\mathbf{x}_0)^T \\ ||\mathbf{B}(\mathbf{x}_0)|| &\leq ||\mathbf{C}_1||\,||\mathbf{V}(\mathbf{x}_0)||\,||\mathbf{e}_0|| \end{aligned} \tag{3.23}$$

$$\begin{aligned} \mathbf{B}(\mathbf{x}_0) &= \mathbf{V}(\mathbf{x}_0)\,\mathbf{F}(\mathbf{x}_0)^T \\ &= \mathbf{V}(\mathbf{x}_0)\,\big(\mathbf{C}_1\big(\mathbf{e}_0 + \mathbf{C}_2\mathbf{e}_0^2 + O\big(\mathbf{e}_0^3\big)\big)\big)^T \\ ||\mathbf{B}(\mathbf{x}_0)|| &\leq ||\mathbf{C}_1||\,||\mathbf{V}(\mathbf{x}_0)||\,||\mathbf{e}_0|| \end{aligned} \tag{3.24}$$

$$\begin{aligned} \mathbf{B}(\mathbf{x}_0) &= M_1(\mathbf{x}_0)\,\mathbf{F}(\mathbf{x}_0)\,\mathbf{V}(\mathbf{x}_0)^T\,M_2(\mathbf{x}_0) \\ &= M_1(\mathbf{x}_0)\,\big(\mathbf{C}_1\big(\mathbf{e}_0 + \mathbf{C}_2\mathbf{e}_0^2 + O\big(\mathbf{e}_0^3\big)\big)\big)\,\mathbf{V}(\mathbf{x}_0)^T\;\;M_2(\mathbf{x}_0) \\ ||\mathbf{B}(\mathbf{x}_0)|| &\leq ||\mathbf{C}_1||\,||M_1(\mathbf{x}_0)||\,||M_2(\mathbf{x}_0)||\,||\mathbf{V}(\mathbf{x}_0)||\,||\mathbf{e}_0|| \end{aligned} \tag{3.25}$$

Notice that all the expressions for $\mathbf{B}(\mathbf{x})$ are of order $\mathbf{e}$ and this is essential in proving the quadratic convergence.

$$\begin{aligned} \mathbf{B}(\mathbf{x}_0) &= M_1(\mathbf{x}_0)\,\mathbf{V}(\mathbf{x}_0)\,\mathbf{F}(\mathbf{x}_0)^T\,M_2(\mathbf{x}_0) \\ &= M_1(\mathbf{x}_0)\,\mathbf{V}(\mathbf{x}_0)\,\big(\mathbf{C}_1\big(\mathbf{e}_0 + \mathbf{C}_2\mathbf{e}_0^2 + O\big(\mathbf{e}_0^3\big)\big)\big)^T\,M_2(\mathbf{x}_0) \\ ||\mathbf{B}(\mathbf{x}_0)|| &\leq ||\mathbf{C}_1||\,||M_1(\mathbf{x}_0)||\,||M_2(\mathbf{x}_0)||\,||\mathbf{V}(\mathbf{x}_0)||\,||\mathbf{e}_0|| \end{aligned} \tag{3.26}$$

$$\begin{aligned} \mathbf{A} &= \mathbf{F}'(\mathbf{x}_0) + \mathbf{B}(\mathbf{x}_0) \\ &= \mathbf{C}_1\,\big(\mathbf{I} + 2\mathbf{C}_2\,\mathbf{e}_0 + \mathbf{C}_1^{-1}\,\mathbf{B}(\mathbf{x}_0) + O(\mathbf{e}_0^2)\big) \\ \mathbf{A}^{-1} &= \big(\mathbf{I} - 2\mathbf{C}_2\,\mathbf{e}_0 - \mathbf{C}_1^{-1}\,\mathbf{B}(\mathbf{x}_0) + O((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2})\big)\mathbf{C}_1^{-1} \end{aligned} \tag{3.27}$$

The explanation to use the notation $O((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2})$ is that the quadratic terms are not of the form $\mathbf{e}_0^2$ because of $\mathbf{B}(\mathbf{x}_0)$. By using (3.27), we deduce

$$\begin{aligned} \mathbf{A}^{-1}\mathbf{F}(\mathbf{x}_0) = {} & \big(\mathbf{I} - 2\mathbf{C}_2\,\mathbf{e}_0 - \mathbf{C}_1^{-1}\,\mathbf{B}(\mathbf{x}_0) + O((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2})\big)\mathbf{C}_1^{-1}\mathbf{C}_1 \\ & \big(\mathbf{e}_0 + \mathbf{C}_2\mathbf{e}_0^2 + O\big(\mathbf{e}_0^3\big)\big) = \mathbf{e}_0 + \mathbf{C}_2\mathbf{e}_0^2 - 2\mathbf{C}_2\mathbf{e}_0^2 - \mathbf{C}_1^{-1}\mathbf{B}(\mathbf{x}_0)\mathbf{e}_0 \\ & + O\big((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2}\,(\mathbf{e}_0)_{i_3}\big)\mathbf{e}_1 \\ = {} & \mathbf{e}_0 - \mathbf{e}_0 + \mathbf{C}_2\mathbf{e}_0^2 + \mathbf{C}_1^{-1}\mathbf{B}(\mathbf{x}_0)\mathbf{e}_0 + O\big((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2}\,(\mathbf{e}_0)_{i_3}\big)\mathbf{e}_1 \\ = {} & \mathbf{C}_2\mathbf{e}_0^2 + \mathbf{C}_1^{-1}\mathbf{B}(\mathbf{x}_0)\mathbf{e}_0 + O\big((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2}\,(\mathbf{e}_0)_{i_3}\big) \end{aligned} \tag{3.28}$$

The error equation $\mathbf{e}_1$ tells that the order of convergence of base method of (3.16) is quadratic because $\mathbf{C}_1^{-1}\mathbf{B}(\mathbf{x}_0)\mathbf{e}_0$ is a second order term in $\mathbf{e}_0$. By using (3.28), (3.22), (3.23), (3.24), (3.25) and (3.26) , we get

$$\begin{aligned}
\mathbf{F}(\mathbf{x}_1) &= \mathbf{C}_1\big(\mathbf{e}_1 + O\big(\mathbf{e}_1^2\big)\big) \\
&= \mathbf{C}_1\big(\mathbf{C}_2\mathbf{e}_0^2 + \mathbf{C}_1^{-1}\mathbf{B}(\mathbf{x}_0)\mathbf{e}_0 + O\big((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2}\,(\mathbf{e}_0)_{i_3}\big)\big) \\
\mathbf{e}_2 &= \mathbf{e}_1 - \mathbf{A}^{-1}\mathbf{F}(\mathbf{x}_1) \\
&= \mathbf{C}_2\mathbf{e}_0^2 + \mathbf{C}_1^{-1}\mathbf{B}(\mathbf{x}_0)\mathbf{e}_0 - \big(\mathbf{I} - 2\mathbf{C}_2\,\mathbf{e}_0 - \mathbf{C}_1^{-1}\,\mathbf{B}(\mathbf{x}_0) \\
&\quad + O((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2})\big)\big(\mathbf{C}_2\mathbf{e}_0^2 + \mathbf{C}_1^{-1}\mathbf{B}(\mathbf{x}_0)\mathbf{e}_0\big) \\
&\quad + O\big((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2}\,(\mathbf{e}_0)_{i_3}\,(\mathbf{e}_0)_{i_4}\big) \\
&= 2\mathbf{C}_2^2\mathbf{e}_0^3 + 2\mathbf{C}_2\mathbf{e}_0\mathbf{C}_1^{-1}\mathbf{B}(\mathbf{x}_0)\mathbf{e}_0 + \mathbf{C}_1^{-1}\mathbf{B}(\mathbf{x}_0)\mathbf{C}_2\mathbf{e}_0^2 + \mathbf{C}_1^{-1}\mathbf{B}(\mathbf{x}_0)\mathbf{C}_1^{-1}\mathbf{B}(\mathbf{x}_0)\mathbf{e}_0 \\
&\quad + O\big((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2}\,(\mathbf{e}_0)_{i_3}\,(\mathbf{e}_0)_{i_4}\big), \\
||\mathbf{e}_2|| &\leq 2||\mathbf{C}_2||^2||\mathbf{e}_0||^3 + 3||\mathbf{C}_2||\,||\mathbf{C}_1||^{-1}||\mathbf{B}(\mathbf{x}_0)||\,||\mathbf{e}_0||^2 + ||\mathbf{C}_1||^{-2}||\mathbf{B}(\mathbf{x}_0)||^2||\mathbf{e}_0||, \\
||\mathbf{e}_2|| &\leq \begin{cases}
 & \text{Iterative method} \\
\big(2||\mathbf{C}_2||^2 + 3||\mathbf{C}_2||\,||M_1(\mathbf{x}_0)||\,||M_2(\mathbf{x}_0)|| + (||M_1(\mathbf{x}_0)||\,||M_2(\mathbf{x}_0)||)^2\big)||\mathbf{e}_0||^3 & (3.7) \\
\big(2||\mathbf{C}_2||^2 + 3||\mathbf{C}_2||\,||\mathbf{V}(\mathbf{x}_0)|| + ||\mathbf{V}(\mathbf{x}_0)||^2\big)||\mathbf{e}_0||^3 & (3.12) \\
\big(2||\mathbf{C}_2||^2 + 3||\mathbf{C}_2||\,||\mathbf{V}(\mathbf{x}_0)|| + ||\mathbf{V}(\mathbf{x}_0)||^2\big)||\mathbf{e}_0||^3 & (3.13) \\
\big(2||\mathbf{C}_2||^2 + 3||\mathbf{C}_2||\,||M_1(\mathbf{x}_0)||\,||M_2(\mathbf{x}_0)||\,||\mathbf{V}(\mathbf{x}_0)|| + (||M_1(\mathbf{x}_0)||\,||M_2(\mathbf{x}_0)||\,||\mathbf{V}(\mathbf{x}_0)||)^2\big)||\mathbf{e}_0||^3 & (3.14) \\
\big(2||\mathbf{C}_2||^2 + 3||\mathbf{C}_2||\,||M_1(\mathbf{x}_0)||\,||M_2(\mathbf{x}_0)||\,||\mathbf{V}(\mathbf{x}_0)|| + (||M_1(\mathbf{x}_0)||\,||M_2(\mathbf{x}_0)||\,||\mathbf{V}(\mathbf{x}_0)||)^2\big)||\mathbf{e}_0||^3 & (3.15)
\end{cases}
\end{aligned} \tag{3.29}$$

The error inequality (3.29) completes the proof. □

The proof of convergence when $m \geq 3$ can be carried via mathematical induction. Suppose the propose the iterative method (3.16) has convergence order s when $m = s - 1$. The error inequality for $(s-1)$-step iterative method (3.16) can be written as

$$||\mathbf{e}_{s-1}|| \leq ||\mathbf{N}_1||\,||\mathbf{e}_0||^s, \tag{3.30}$$

where $||\mathbf{N}_1||$ is finite. The error equation for $m = s$ is

$$\begin{aligned}
\mathbf{e}_s &= \mathbf{e}_{s-1} - \mathbf{A}^{-1}\,\mathbf{F}(\mathbf{x}_{s-1}) \\
&= \mathbf{e}_{s-1} - \big(\mathbf{I} - 2\mathbf{C}_2\mathbf{e}_0 - \mathbf{C}_1^{-1}\mathbf{B}(\mathbf{x}_0) + O\big((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2}\big)\big)\mathbf{e}_{s-1} \\
||\mathbf{e}_s|| &\leq 2||\mathbf{C}_2||\,||\mathbf{N}_1||\,||\mathbf{e}_0||^{s+1} + ||\mathbf{C}_1||^{-1}||\mathbf{B}(\mathbf{x}_0)||\,||\mathbf{N}_1||\,||\mathbf{e}_0||^s \\
||\mathbf{e}_s|| &\leq ||\mathbf{N}_2||\,||\mathbf{e}_0||^{s+1},
\end{aligned} \tag{3.31}$$

where $||\mathbf{N}_2||$ is finite and its value can be determine by using the fact that $||\mathbf{B}(\mathbf{x}_0)||$ is bounded above. The upper bounds for $||\mathbf{B}(\mathbf{x}_0)||$ are given in (3.22), (3.23), (3.24), (3.25) and (3.26).

3.4 Numerical simulations

Let $\boldsymbol{\alpha}$ be a simple root of system of nonlinear equations $\mathbf{F}(\mathbf{x}) = \mathbf{0}$. We adopt the following definition of computational order of convergence [68]

$$\text{CCO} = \frac{log\big(||\mathbf{F}(\mathbf{x}_{k+1})||_\infty/||\mathbf{F}(\mathbf{x}_k)||_\infty\big)}{log\big(||\mathbf{F}(\mathbf{x}_k)||_\infty/||\mathbf{F}(\mathbf{x}_{k-1})||_\infty\big)}. \tag{3.32}$$

To check the performance of our proposed iterative methods, we solve three problems.

$$\text{Problem 1} = \begin{cases} x_i^2\, x_{i+1} - 1 = 0, \quad i = 1, 2 \cdots, n-1 \\ x_n\, x_1 - 1 = 0, \qquad i = n \end{cases}$$

$$\text{Problem 2} = \begin{cases} F_1(\mathbf{x}) = (3 - 0.5\, x_1)\, x_1 - 2\, x_2 + 1 \\ F_2(\mathbf{x}) = (3 - 0.5\, x_n)\, x_n - 2\, x_{n-1} + 1 \\ F_i(\mathbf{x}) = (3 - 0.5\, x_i)\, x_i - x_{i-1} + 2\, x_{i+1} + 1, \quad i = 2, 3 \cdots, n-1 \end{cases}$$

$$\text{Problem 3} = \begin{cases} F_1(\mathbf{x}) = 10\, x_1 + \sin(x_1 + x_2) - 1 = 0 \\ F_2(\mathbf{x}) = 8\, x_2 - \cos^2(x_3 - x_2) - 1 = 0 \\ F_3(\mathbf{x}) = 12\, x_3 + \sin(x_3) - 1 = 0 \end{cases}$$

Tables 9.1, 9.2, and 9.3 clearly show that the claimed orders of convergence are in agreement with computational orders of convergence for a given number of steps. The simulation times in each Table for all the conducted tests are almost equal. In Table 9.3, we have used full matrices as preconditioners because the system of nonlinear equations is small. But for a large system of nonlinear equations, it is not recommendable to use full matrices as a preconditioners and reason is clear, since we get a penalty in terms of computational cost. One of the main targets of the present article is to explore different possibilities of preconditioning, by keeping the convergence order. In our analysis, we found that for solving system of nonlinear equations, the iterative method (3.7) is the most efficient when we use the preconditioning matrices $M_1(\mathbf{x})$ and $M_2(\mathbf{x})$ as diagonal matrices. It is also observed that the leading constant coefficients of preconditioners should be less one in magnitude to get better accuracy. The proposed iterative methods show better accuracy compared with multi-step Newton method for almost all tests, when considering the previous three model problems.

	Method (3.7)				Method (3.17)	
$M_1(\mathbf{x})$	$M_2(\mathbf{x})$	m	$\lvert\lvert\mathbf{F}(\mathbf{x})\rvert\rvert_\infty$	CCO	$\lvert\lvert\mathbf{F}(\mathbf{x})\rvert\rvert_\infty$	CCO
$\mathbf{I}$	$\llbracket -\sin(\mathbf{x})/(1.1+\cos(\mathbf{x}))\rrbracket$	1	$2.97e-23$	2.0	$3.20e-7$	1.99
$\mathbf{I}$	$-\mathbf{I}$	2	$1.95e-127$	4.0	$4.15e-28$	3.0
$\mathbf{I}$	$\llbracket -2\exp(-2\mathbf{x})\rrbracket$	3	$7.20e-89$	4.0	$5.56e-80$	4.0
$\mathbf{I}$	$\llbracket -\exp(-\mathbf{x})\rrbracket$	4	$3.21e-221$	5.0	$3.20e-185$	5.0

TABLE 3.1: Problem 1: Initial guess: $x_i = 15/10$, $n = 200$, Iter= 5

	Method (3.7)				Method (3.17)	
$M_1(\mathbf{x})$	$M_2(\mathbf{x})$	m	$\lvert\lvert\mathbf{F}(\mathbf{x})\rvert\rvert_\infty$	CCO	$\lvert\lvert\mathbf{F}(\mathbf{x})\rvert\rvert_\infty$	CCO
$\mathbf{I}$	$\llbracket 1/10\rrbracket$	3	$2.33e-219$	4.0	$5.92e-163$	4.0
$\mathbf{I}$	$\llbracket 1/10\rrbracket$	4	$4.09e-511$	5.0	$1.18e-388$	5.0
$\mathbf{I}$	$\llbracket 1/10\ \exp(\mathbf{x}/10)\rrbracket$	3	$4.92e-214$	4.0	$5.92e-163$	4.0
$\mathbf{I}$	$\llbracket 1/10\ \exp(\mathbf{x}/10)\rrbracket$	4	$3.79e-499$	5.0	$1.18e-388$	5.0
$\mathbf{I}$	$\llbracket\cosh(\mathbf{x})/(10+\sinh(\mathbf{x}))\rrbracket$	3	$3.52e-264$	4.0	$5.92e-163$	4.0
$\mathbf{I}$	$\llbracket\cosh(\mathbf{x})/(10+\sinh(\mathbf{x}))\rrbracket$	4	$2.77e-614$	5.0	$1.18e-388$	5.0
$\mathbf{I}$	$\llbracket\cosh(\mathbf{x})/10\rrbracket$	3	$1.97e-292$	4.0	$5.92e-163$	4.0
$\mathbf{I}$	$\llbracket\cosh(\mathbf{x})/10\rrbracket$	4	$7.68e-677$	5.0	$1.18e-388$	5.0
$\mathbf{I}$	$\llbracket\operatorname{sech}(\mathbf{x})/10\rrbracket$	3	$1.33e-200$	4.0	$5.92e-163$	4.0
$\mathbf{I}$	$\llbracket\operatorname{sech}(\mathbf{x})/10\rrbracket$	4	$4.26e-469$	5.0	$1.18e-388$	5.0
$\mathbf{I}$	$\llbracket\cos(\mathbf{x})/3\rrbracket$	3	$3.09e-267$	4.0	$5.92e-163$	4.0
$\mathbf{I}$	$\llbracket\cos(\mathbf{x})/3\rrbracket$	4	$2.55e-623$	5.0	$1.18e-388$	5.0
$\mathbf{I}$	$\llbracket\left(1+x^3\right)/3\rrbracket$	3	$5.71e-187$	4.0	$5.92e-163$	4.0
$\mathbf{I}$	$\llbracket\left(1+x^3\right)/3\rrbracket$	4	$4.09e-443$	5.0	$1.18e-388$	5.0
$\mathbf{I}$	$\llbracket\sinh(\left(\mathbf{x}^2\right)/10\rrbracket$	3	$7.21e-217$	4.0	$5.92e-163$	4.0
$\mathbf{I}$	$\llbracket\sinh(\left(\mathbf{x}^2\right)/10\rrbracket$	4	$4.16e-462$	5.0	$1.18e-388$	5.0
$\mathbf{I}$	$\llbracket\mathbf{x}^2/10\rrbracket$	3	$8.41e-204$	4.0	$5.92e-163$	4.0
$\mathbf{I}$	$\llbracket\mathbf{x}^2/10\rrbracket$	4	$1.13e-482$	5.0	$1.18e-388$	5.0

TABLE 3.2: Problem 2: Initial guess: $x_i = -1$, $n = 100$, Iter= 4

		Methods (3.16)					Method (3.17)	
Methods	$M_1(\mathbf{x})$	$M_2(\mathbf{x})$	$\mathbf{V}(\mathbf{x})$	m	$\|\|\mathbf{F}(\mathbf{x})\|\|_\infty$	CCO	$\|\|\mathbf{F}(\mathbf{x})\|\|_\infty$	CCO
(3.7)	$\mathbf{R}_1$	$\mathbf{I}$	-	1	$1.63e-241$	2.0	$6.69e-268$	2.0
(3.7)	$\mathbf{I}$	$\mathbf{R}_1$	-	1	$1.34e-281$	2.0	$6.69e-268$	2.0
(3.7)	$\mathbf{R}_1$	$\mathbf{I}$	-	2	$1.19e-6427$	3.0	$1.74e-6229$	3.0
(3.7)	$\mathbf{I}$	$\mathbf{R}_1$	-	2	$4.27e-8197$	3.0	$1.74e-6229$	3.0
(3.13)	$\mathbf{I}$	$\mathbf{I}$	$\mathbf{R}_2$	1	$1.49e-272$	2.0	$6.69e-268$	2.0
(3.14)	$[\![\mathbf{F}(\mathbf{x})]\!]$	$\mathbf{I}$	$0.001\,\mathbf{R}_2$	1	$4.43e-290$	2.0	$6.69e-268$	2.0
(3.14)	$[\![\mathbf{F}(\mathbf{x})]\!]$	$\mathbf{I}$	$0.001\,\mathbf{R}_3$	1	$2.22e-283$	2.0	$6.69e-268$	2.0
(3.14)	$[\![\mathbf{F}(\mathbf{x})]\!]$	$\mathbf{I}$	$0.001\,\mathbf{R}_4$	1	$1.25e-286$	2.0	$6.69e-268$	2.0
(3.14)	$\mathbf{I}$	$[\![\mathbf{F}(\mathbf{x})]\!]$	$0.001\,\mathbf{R}_2$	1	$2.84e-270$	2.0	$6.69e-268$	2.0
(3.14)	$\mathbf{I}$	$[\![\mathbf{F}(\mathbf{x})]\!]$	$0.001\,\mathbf{R}_3$	1	$3.11e-270$	2.0	$6.69e-268$	2.0
(3.14)	$\mathbf{I}$	$[\![\mathbf{F}(\mathbf{x})]\!]$	$0.001\,\mathbf{R}_4$	1	$5.34e-273$	2.0	$6.69e-268$	2.0
(3.15)	$[\![-\mathbf{F}(\mathbf{x})]\!]$	$\mathbf{I}$	$\mathbf{R}_5$	1	$1.36e-278$	2.0	$6.69e-268$	2.0
(3.15)	$\mathbf{I}$	$\mathbf{I}$	$\mathbf{R}_6$	1	$3.74e-296$	2.0	$6.69e-268$	2.0
(3.15)	$\mathbf{I}$	$\mathbf{I}$	$\mathbf{R}_7$	1	$4.75e-300$	2.0	$6.69e-268$	2.0
(3.15)	$\mathbf{I}$	$\mathbf{I}$	$\mathbf{R}_8$	1	$3.73e-334$	2.0	$6.69e-268$	2.0
(3.15)	$\mathbf{I}$	$\mathbf{I}$	$\mathbf{R}_9$	1	$7.06e-325$	2.0	$6.69e-268$	2.0
(3.15)	$\mathbf{I}$	$\mathbf{I}$	$\mathbf{R}_{10}$	1	$5.79e-303$	2.0	$6.69e-268$	2.0

$\mathbf{R}_1 = (0.1\,\exp(0.1\,\mathbf{x}))\,(0.1\,\exp(0.1\,\mathbf{x}))^T$, $\mathbf{R}_2 = [1,0,0]^T$, $\mathbf{R}_3 = [0,1,0]^T$, $\mathbf{R}_4 = [0,0,1]^T$

$\mathbf{R}_5 = 0.001\,[-1,-1,-1]^T$, $\mathbf{R}_6 = 0.001\,[1,-1,1]^T$, $\mathbf{R}_7 = 0.001\,[-1,-1,1]^T$, $\mathbf{R}_8 = 0.001\,[-1,-1,2]^T$

$\mathbf{R}_9 = 0.001\,[-1,-2,1]^T$, $\mathbf{R}_{10} = 0.001\,[-2,-1,1]^T$,

TABLE 3.3: Problem 3: Initial guess: $x_i = 15/10$, $n = 3$, Iter= 8

3.5 Summary

The computational cost of the classical multi-step Newton method and that of the proposed iterative methods are essentially the same, if we do not use dense preconditioners. Indeed, the iterative method (3.7) is more effective, when coupled with diagonal preconditioners. The proposed family of iterative methods has the same convergence order as that of Newton multi-step iterative method, with almost the same computational cost. The only assumption on the preconditioners is that they should have finite norms, in their definition domain.

Chapter 4

Multi-step derivative-free preconditioned Newton method for solving systems of nonlinear equations

Preconditioning of systems of nonlinear equations modifies the associated Jacobian and provides faster convergence. The preconditioners are introduced in a way that they do not affect the convergence order of underlying iterative method. The multi-step derivative-free iterative method consists of a base method and multi-step part. In the base method, the Jacobian of the system of nonlinear equation is approximated by a finite difference operator and preconditioners add an extra term to modify it. The inversion of the modified finite difference operator is avoided by computing LU factors. Once we have the LU factors, we repeatedly use them to solve lower and upper triangular systems in the multi-step part to enhance the convergence order. The convergence order of m-step Newton iterative method is $m + 1$. The claimed convergence orders are verified by computing numerically the convergence order. Furthermore, numerical simulations clearly show that a good selection of a preconditioning strategy provides numerical stability, accuracy, and rapid convergence.

4.1 Introduction

Let $\mathbf{F} : D \subseteq \mathbb{R}^n \longrightarrow \mathbb{R}^n$ be a nonlinear function and the system of nonlinear equations can be written as

$$\mathbf{F}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_n(\mathbf{x})]^T = \mathbf{0}, \tag{4.1}$$

where $\mathbf{x} = [x_1, x_2, \cdots, x_n]^T$. If $\mathbf{x}^*$ is a simple root of (4.1) then $\mathbf{F}(\mathbf{x}^*) = \mathbf{0}$ and $\det(\mathbf{F}'(\mathbf{x}^*)) \neq 0$ i.e. Jacobian should not be singular at the root. Newton method [1, 3, 7, 8] is the classical iterative method for computing the simple root of system of nonlinear equation. The multi-step Newton method[113] can be written as

$$\begin{aligned}
&\text{Base method} \longrightarrow \begin{cases} \mathbf{x}_0 = \text{initial guess} \\ \mathbf{F}'(\mathbf{x}_0)\,\boldsymbol{\phi}_1 = \mathbf{F}\left(\mathbf{x}_0\right) \\ \mathbf{x}_1 = \mathbf{x}_0 - \boldsymbol{\phi}_1 \end{cases} \\
&\text{Multi-step part} \rightarrow \begin{cases} \text{for} \quad j = 2, m \\ \qquad \mathbf{F}'(\mathbf{x}_0)\,\boldsymbol{\phi}_j = \mathbf{F}\left(\mathbf{x}_{j-1}\right) \\ \qquad \mathbf{x}_j = \mathbf{x}_{j-1} - \boldsymbol{\phi}_j \\ \text{end} \\ \mathbf{x}_0 = \mathbf{x}_m \end{cases}
\end{aligned} \tag{4.2}$$

and its order of convergence is $m + 1$. Many researchers [57, 70, 107, 108] have proposed higher order multi-step an iterative method for solving system of nonlinear equations. In most of real world problems, the closed form expression for the system of nonlinear equations is not always possible. when we get the information about the system of nonlinear equations from a black box then the computation of Jacobian analytically is no way possible. So it means, we need to compute it numerically. Recently people have proposed derivative-free iterative method [114–118] for the solution of the system of nonlinear equations. Grau et al. [116] have constructed the following multi-step derivative-free iterative method for solving the system of nonlinear equations.

$$
\begin{aligned}
&\text{Base method} \longrightarrow \begin{cases} \mathbf{x}_0 = \text{initial guess} \\ \boldsymbol{u} = \mathbf{x}_0 + \beta\, \mathbf{F}(\mathbf{x}_0) \\ [\boldsymbol{u}, \mathbf{x}_0; \mathbf{F}]\, \boldsymbol{\phi}_1 = \mathbf{F}\left(\mathbf{x}_0\right) \\ \mathbf{x}_1 = \mathbf{x}_0 - \boldsymbol{\phi}_1 \end{cases} \\
&\text{Multi-step part} \rightarrow \begin{cases} \text{for } \ j = 2, m \\ \qquad [\boldsymbol{u}, \mathbf{x}_0; \mathbf{F}]\, \boldsymbol{\phi}_j = \mathbf{F}\left(\mathbf{x}_{j-1}\right) \\ \qquad \mathbf{x}_j = \mathbf{x}_{j-1} - \boldsymbol{\phi}_j \\ \text{end} \\ \mathbf{x}_0 = \mathbf{x}_m \end{cases} \qquad ,
\end{aligned}
\tag{4.3}
$$

where β is a scalar parameter and the $[.,.;\mathbf{F}] : D \times D \subset \mathbb{R}^n \times \mathbb{R}^n \longrightarrow L(\mathbb{R}^n)$ is divided difference operator of $\mathbf{F}$. The divided difference operator is defined as

$$
\begin{aligned}
[\mathbf{x} + \mathbf{h}, \mathbf{x}; \mathbf{F}] &= \int_0^1 \mathbf{F}'(\mathbf{x} + t\,\mathbf{h})\, dt, \quad \forall \mathbf{x}, \mathbf{h} \in \mathbb{R}^n \\
&= \mathbf{F}'(\mathbf{x}) + \frac{1}{2}\mathbf{F}''(\mathbf{x})\, \mathbf{h} + \frac{1}{6}\, \mathbf{F}'''(\mathbf{x})\, \mathbf{h}^2 + O\big(\mathbf{h}^3\big),
\end{aligned}
\tag{4.4}
$$

where $\mathbf{h}^i = \overbrace{(h, h, \cdots, h)}^{i\, times}$. The idea of preconditioning of system of nonlinear equations are reported by many authors [109–112]. Let $\mathbf{G}(\mathbf{x}) = [g_1(\mathbf{x}), g_2(\mathbf{x}), \cdots, g_n(\mathbf{x})]^T$ be a non-zero sufficiently differentiable function. We define a new function

$$
\mathbf{Q}(\mathbf{x}) = \mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}) = [\![\mathbf{G}(\mathbf{x})]\!]\, \mathbf{F}(\mathbf{x}) = [\![\mathbf{F}(\mathbf{x})]\!]\, \mathbf{G}(\mathbf{x}),
\tag{4.5}
$$

where $\odot$ is the element-wise multiplication and $[\![\cdot]\!]$ represent the diagonal matrix. The first order Fréchet derivative of (4.5) can be computed as

$$
\begin{aligned}
\mathbf{Q}'(\mathbf{x}) &= [\![\mathbf{F}(\mathbf{x})]\!]\, \mathbf{G}'(\mathbf{x}) + [\![\mathbf{G}(\mathbf{x})]\!]\, \mathbf{F}'(\mathbf{x}) \\
&= [\![\mathbf{G}(\mathbf{x})]\!]\, \big(\mathbf{F}'(\mathbf{x}) + [\![\mathbf{F}(\mathbf{x})]\!]\, [\![\mathbf{G}(\mathbf{x})]\!]^{-1}\, \mathbf{G}'(\mathbf{x})\big).
\end{aligned}
\tag{4.6}
$$

The application of Newton method to (4.5) gives

$$\begin{aligned}\mathbf{x}_{k+1} &= \mathbf{x}_k - \mathbf{Q}'(\mathbf{x}_k)^{-1}\,\mathbf{Q}(\mathbf{x}_k)\\ &= \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + [\![\mathbf{F}(\mathbf{x}_k)]\!]\,[\![\mathbf{G}(\mathbf{x}_k)]\!]^{-1}\,\mathbf{G}'(\mathbf{x}_k)\right)^{-1}\mathbf{F}(\mathbf{x}_k).\end{aligned} \tag{4.7}$$

The convergence order of (4.7) is quadratic because, it is the Newton method for solving preconditioned system of nonlinear equations $\mathbf{Q}(\mathbf{x}) = \mathbf{0}$. If we replace $\mathbf{G}(\mathbf{x})$ by $exp(\mathbf{G}(\mathbf{x}))$ then (4.7) can be written as

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + [\![\mathbf{F}(\mathbf{x}_k)]\!]\mathbf{G}'(\mathbf{x}_k)\right)^{-1}\mathbf{F}(\mathbf{x}_k). \tag{4.8}$$

4.2 Proposed iterative methods

We are interested to proposed derivative-free version of (4.8) with some generalization of preconditioner. Our proposal of multi-step derivative-free preconditioned iterative method is the following

$$\begin{aligned}&\text{Base method} \longrightarrow \begin{cases}\mathbf{x}_0 = \text{initial guess}\\ \boldsymbol{u} = \mathbf{x}_0 + \beta\,\mathbf{F}(\mathbf{x}_0)\\ \mathbf{A} = [\boldsymbol{u}, \mathbf{x}_0; \mathbf{F}] + [\![\mathbf{q}_1(\mathbf{x}) \odot \mathbf{q}_2(\mathbf{F}(\mathbf{x}))]\!]\\ \mathbf{A}\,\boldsymbol{\phi}_1 = \mathbf{F}\left(\mathbf{x}_0\right)\\ \mathbf{x}_1 = \mathbf{x}_0 - \boldsymbol{\phi}_1\end{cases}\\ &\text{Multi-step part} \rightarrow \begin{cases}\text{for} \quad j = 2, m\\ \qquad \mathbf{A}\,\boldsymbol{\phi}_j = \mathbf{F}\left(\mathbf{x}_{j-1}\right)\\ \qquad \mathbf{x}_j = \mathbf{x}_{j-1} - \boldsymbol{\phi}_j\\ \text{end}\\ \mathbf{x}_0 = \mathbf{x}_m\end{cases}\end{aligned}, \tag{4.9}$$

where $\mathbf{q}_1, \mathbf{q}_2 : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ are sufficiently differentiable, $\mathbf{q}_2(\mathbf{0}) = \mathbf{0}$ and $\mathbf{q}_i(\mathbf{x}) = [q_i(x_1), q_i(x_2)\cdots, q_i(x_n)]^T$ for $i = 1, 2$. We claim that the convergence order of proposed preconditioned m-step derivative-free iterative method is $m + 1$.

4.3 Convergence Analysis

We demonstrate the proof of convergence order of (4.9) only for $m = 2$ and the case $m \geq 3$, we use mathematical induction.

Theorem 4.1. *Let* $\mathbf{F} : \Gamma \subseteq \mathbb{R}^n \to \mathbb{R}^n$ *be sufficiently Frechet differentiable on an open convex neighborhood* Γ *of* $\mathbf{x}^* \in \mathbb{R}^n$ *with* $\mathbf{F}(\mathbf{x}^*) = 0$ *and* $det\left(\mathbf{F}'(\mathbf{x}^*)\right) \neq 0$*. By taking* $\mathbf{x}_0$ *in the vicinity of* $\mathbf{x}^*$*, the sequence* $\{\mathbf{x}_k\}$ *generated by (4.9) converges to* $\mathbf{x}^*$ *with local order of convergence at least three for* $m = 2$ *and the following error equation*

$$\begin{aligned}\mathbf{e}_2 &= \mathbf{B}_1\mathbf{e}_0\big(-\mathbf{C}_2\mathbf{e}_0^2 + \mathbf{B}_1\mathbf{e}_0^2 + \mathbf{C}_1^{-1}\llbracket \mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)\rrbracket\mathbf{e}_0\big)\\ &\quad + \mathbf{C}_1^{-1}\llbracket \mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)\rrbracket\big(- \mathbf{C}_2\mathbf{e}_0^2 + \mathbf{B}_1\mathbf{e}_0^2 + \\ &\quad \mathbf{C}_1^{-1}\llbracket \mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)\rrbracket\mathbf{e}_0\big) + O\big((\mathbf{e}_0)_{i_1}(\mathbf{e}_0)_{i_2}(\mathbf{e}_0)_{i_3}(\mathbf{e}_0)_{i_4}\big)\end{aligned} \tag{4.10}$$

where $\mathbf{e}_k = \boldsymbol{x}_k - \mathbf{x}^*$, $\mathbf{e}_k{}^p = \overbrace{(\mathbf{e}_2, \mathbf{e}_k, \cdots, \mathbf{e}_k)}^{p\ times}$, $\mathbf{e}_k = [(\mathbf{e}_k)_1, (\mathbf{e}_k)_2, \cdots, (\mathbf{e}_k)_n]^T$, $\boldsymbol{\gamma}_1 = \mathbf{q}_2(\mathbf{0})$*,* $\mathbf{B}_1 = \beta\,\mathbf{C}_2\,\mathbf{C}_1 + 2\,\mathbf{C}_2$ *and* $\mathbf{B}_2 = \beta^2\,\mathbf{C}_3\,\mathbf{C}_1^2 + \beta\,\mathbf{C}_2\,\mathbf{C}_1\,\mathbf{C}_2 + 3\,\beta\,\mathbf{C}_3\,\mathbf{C}_1 + 3\,\mathbf{C}_3$*.*

Proof. The rth Frechet derivative of $\mathbf{F}$ at $v \in \mathbb{R}^n$, $r \geq 1$, is the $q-linear$ function $\mathbf{F}^{(r)}(v) : \overbrace{\mathbb{R}^n\mathbb{R}^n\cdots\mathbb{R}^n}^{\text{r times}}$ such that $\mathbf{F}^{(r)}(v)(u_1, u_2, \cdots, u_r) \in \mathbb{R}^n$. The Taylor's series expansion of $\mathbf{F}(\mathbf{x}_0)$ around $\mathbf{x}^*$ can be written as

$$\begin{aligned}\mathbf{F}(\mathbf{x}_0) &= \mathbf{F}(\mathbf{x}^* + \mathbf{x}_0 - \mathbf{x}^*) = \mathbf{F}(\mathbf{x}^* + \mathbf{e}_0),\\ &= \mathbf{F}(\mathbf{x}^*) + \mathbf{F}'(\mathbf{x}^*)\,\mathbf{e}_0 + \frac{1}{2!}\mathbf{F}''(\mathbf{x}^*)\,\mathbf{e}_0{}^2 + \frac{1}{3!}\mathbf{F}^{(3)}(\mathbf{x}^*)\,\mathbf{e}_0{}^3 + \cdots,\\ &= \mathbf{F}'(\mathbf{x}^*)\Bigg(\mathbf{e}_0 + \frac{1}{2!}\mathbf{F}'(\mathbf{x}^*)^{-1}\mathbf{F}''(\mathbf{x}^*)\,\mathbf{e}_0{}^2\\ &\quad + \frac{1}{3!}\mathbf{F}'(\mathbf{x}^*)^{-1}\mathbf{F}^{(3)}(\mathbf{x}^*)\,\mathbf{e}_0{}^3 + \cdots\Bigg),\\ &= \mathbf{C}_1\left(\mathbf{e}_0 + \mathbf{C}_2\,\mathbf{e}_0{}^2 + \mathbf{C}_3\,\mathbf{e}_0{}^3 + O\left(\mathbf{e}_0{}^4\right)\right),\end{aligned} \tag{4.11}$$

where $\mathbf{C}_1 = \mathbf{F}'(\mathbf{x}^*)$ and $\mathbf{C}_s = \frac{1}{s!}\mathbf{F}'(\mathbf{x}^*)^{-1}\mathbf{F}^{(s)}(\mathbf{x}^*)$ for $s \geq 2$. From (4.11), we can calculate the Fréchet derivatives of $\mathbf{F}$ as

$$\begin{aligned}
\mathbf{F}'\left(\mathbf{x}_0\right) &= \mathbf{C}_1\left(\mathbf{I}+2\mathbf{C}_2\,\mathbf{e}_0+3\mathbf{C}_3\,{\mathbf{e}_0}^2+4\mathbf{C}_3\,{\mathbf{e}_0}^3+O\left({\mathbf{e}_0}^4\right)\right)\\
\mathbf{F}''\left(\mathbf{x}_0\right) &= \mathbf{C}_1\left(2\mathbf{C}_2\;+6\mathbf{C}_3\,\mathbf{e}_0+O\left({\mathbf{e}_0}^2\right)\right)\\
\mathbf{F}'''\left(\mathbf{x}_0\right) &= \mathbf{C}_1\left(6\mathbf{C}_3\;+O\left(\mathbf{e}_0\right)\right),
\end{aligned}\tag{4.12}$$

where $\mathbf{I}$ is the identity matrix. By using (4.4), the following expansion

$$[\boldsymbol{u},\mathbf{x}_0;\mathbf{F}] = \mathbf{C}_1\left(\mathbf{I}+\mathbf{B}_1\mathbf{e}_0+\mathbf{B}_2\,\mathbf{e}_0^2+O\big(\mathbf{e}_0^3\big)\right),\tag{4.13}$$

where $\mathbf{B}_1=\beta\,\mathbf{C}_2\,\mathbf{C}_1+2\,\mathbf{C}_2$ and $\mathbf{B}_2=\beta^2\,\mathbf{C}_3\,\mathbf{C}_1^2+\beta\,\mathbf{C}_2\,\mathbf{C}_1\,\mathbf{C}_2+3\,\beta\,\mathbf{C}_3\,\mathbf{C}_1+3\,\mathbf{C}_3$. Next, we expand $\mathbf{q}_2(\mathbf{F}(\mathbf{x}))$ as

$$\begin{aligned}
\mathbf{q}_2(\mathbf{F}(\mathbf{x})) &= \mathbf{q}_2'(\mathbf{0})\,\mathbf{F}(\mathbf{x})+\mathbf{q}_2''(\mathbf{0})\,\mathbf{F}(\mathbf{x})\odot\mathbf{F}(\mathbf{x})+\cdots\\
\mathbf{q}_1(\mathbf{x})\odot\mathbf{q}_2(\mathbf{F}(\mathbf{x})) &= \mathbf{q}_1(\mathbf{x})\odot\mathbf{q}_2'(\mathbf{0})\,\mathbf{F}(\mathbf{x})\\
&\quad+\mathbf{q}_1(\mathbf{x})\odot\mathbf{q}_2''(\mathbf{0})\,\mathbf{F}(\mathbf{x})\odot\mathbf{F}(\mathbf{x})+\cdots\\
&= \mathbf{q}_1(\mathbf{x})\odot(\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)+\mathbf{q}_1(\mathbf{x})\odot\left(\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{C}_2\mathbf{e}_0^2\right)\\
&\quad+\mathbf{q}_1(\mathbf{x})\odot(\boldsymbol{\gamma}_2\,\llbracket\mathbf{C}_1\,\mathbf{e}_0\rrbracket\mathbf{C}_1\,\mathbf{e}_0)\\
&\quad+O\big((\mathbf{e}_0)_{i_1}(\mathbf{e}_0)_{i_2}(\mathbf{e}_0)_{i_3}\big),
\end{aligned}\tag{4.14}$$

where $\boldsymbol{\gamma}_1=\mathbf{q}_2'(\mathbf{0})$, $\boldsymbol{\gamma}_2=\mathbf{q}_2''(\mathbf{0})$ and $O\big((\mathbf{e}_0)_{i_1}(\mathbf{e}_0)_{i_2}(\mathbf{e}_0)_{i_3}\big)$ represents the third order term. The expansion of $\mathbf{A}$ by using (4.13) and (4.14) is

$$\begin{aligned}
\mathbf{A} &= \mathbf{C}_1\left(\mathbf{I}+\mathbf{B}_1\mathbf{e}_0+\mathbf{B}_2\,\mathbf{e}_0^2\right)+\llbracket\mathbf{q}_1(\mathbf{x})\odot(\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)\rrbracket\\
&\quad+\llbracket\mathbf{q}_1(\mathbf{x})\odot\left(\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{C}_2\mathbf{e}_0^2\right)\rrbracket+\llbracket\mathbf{q}_1(\mathbf{x})\odot(\boldsymbol{\gamma}_2\,\llbracket\mathbf{C}_1\,\mathbf{e}_0\rrbracket\mathbf{C}_1\,\mathbf{e}_0)\rrbracket+\\
&\qquad+O\big((\mathbf{e}_0)_{i_1}(\mathbf{e}_0)_{i_3}(\mathbf{e}_0)_{i_3}\big)\\
\mathbf{A} &= \mathbf{C}_1\Big(\mathbf{I}+\mathbf{B}_1\mathbf{e}_0+\mathbf{B}_2\,\mathbf{e}_0^2+\mathbf{C}_1^{-1}\llbracket\mathbf{q}_1(\mathbf{x})\odot(\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)\rrbracket\\
&\quad+\mathbf{C}_1^{-1}\llbracket\mathbf{q}_1(\mathbf{x})\odot\left(\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{C}_2\mathbf{e}_0^2\right)\rrbracket\\
&\qquad+\mathbf{C}_1^{-1}\llbracket\mathbf{q}_1(\mathbf{x})\odot(\boldsymbol{\gamma}_2\,\llbracket\mathbf{C}_1\,\mathbf{e}_0\rrbracket\mathbf{C}_1\,\mathbf{e}_0)\rrbracket+O\big((\mathbf{e}_0)_{i_1}(\mathbf{e}_0)_{i_2}(\mathbf{e}_0)_{i_3}\big)\Big)\\
\mathbf{A}^{-1} &= \Big(\mathbf{I}-\mathbf{B}_1\mathbf{e}_0-\mathbf{C}_1^{-1}\llbracket\mathbf{q}_1(\mathbf{x})\odot(\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)\rrbracket-\mathbf{B}_2\,\mathbf{e}_0^2\\
&\quad-\mathbf{C}_1^{-1}\llbracket\mathbf{q}_1(\mathbf{x})\odot\left(\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{C}_2\mathbf{e}_0^2\right)\rrbracket-\mathbf{C}_1^{-1}\llbracket\mathbf{q}_1(\mathbf{x})\odot(\boldsymbol{\gamma}_2\,\llbracket\mathbf{C}_1\,\mathbf{e}_0\rrbracket\mathbf{C}_1\,\mathbf{e}_0)\rrbracket\\
&\quad+O\big((\mathbf{e}_0)_{i_1}(\mathbf{e}_0)_{i_2}(\mathbf{e}_0)_{i_3}\big)\Big)\mathbf{C}_1^{-1}\\
\mathbf{A}^{-1} &= \Big(\mathbf{I}-\mathbf{B}_1\mathbf{e}_0-\mathbf{C}_1^{-1}\llbracket\mathbf{q}_1(\mathbf{x})\odot(\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)\rrbracket+O\big((\mathbf{e}_0)_{i_1}(\mathbf{e}_0)_{i_2}\big)\Big)\mathbf{C}_1^{-1}
\end{aligned}\tag{4.15}$$

By using (4.15), we get

$$\begin{aligned}
\mathbf{A}^{-1}\mathbf{F}(\mathbf{x}_0) = \Big(&\mathbf{I} - \mathbf{B}_1\mathbf{e}_0 - \mathbf{C}_1^{-1}[\![\mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)]\!] + O\big((\mathbf{e}_0)_{i_1}(\mathbf{e}_0)_{i_2}\big)\Big) \\
&\mathbf{C}_1^{-1}\mathbf{C}_1\big(\mathbf{e}_0 + \mathbf{C}_2\mathbf{e}_0^2 + O\big(\mathbf{e}_0^3\big)\big) \\
&= \mathbf{e}_0 + \mathbf{C}_2\mathbf{e}_0^2 - \mathbf{B}_1\mathbf{e}_0^2 - \mathbf{C}_1^{-1}[\![\mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)]\!]\mathbf{e}_0 \\
&+ O\big((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2}\,(\mathbf{e}_0)_{i_3}\big) \\
\mathbf{e}_1 &= \mathbf{e}_0 - \mathbf{e}_0 - \mathbf{C}_2\mathbf{e}_0^2 + \mathbf{B}_1\mathbf{e}_0^2 + \mathbf{C}_1^{-1}[\![\mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)]\!]\mathbf{e}_0 \\
&+ O\big((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2}\,(\mathbf{e}_0)_{i_3}\big) \\
\mathbf{e}_1 &= -\mathbf{C}_2\mathbf{e}_0^2 + \mathbf{B}_1\mathbf{e}_0^2 + \mathbf{C}_1^{-1}[\![\mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)]\!]\mathbf{e}_0 \\
&+ O\big((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2}\,(\mathbf{e}_0)_{i_3}\big).
\end{aligned} \tag{4.16}$$

The error equation $\mathbf{e}_1$ tells that the order of convergence of base method of (4.9) is quadratic.

$$\begin{aligned}
\mathbf{F}(\mathbf{x}_1) &= \mathbf{C}_1\big(\mathbf{e}_1 + O\big(\mathbf{e}_1^2\big)\big) \\
\mathbf{e}_2 &= \mathbf{e}_1 - \mathbf{A}^{-1}\mathbf{C}_1\big(\mathbf{e}_1 + O\big(\mathbf{e}_1^2\big)\big) \\
&= \mathbf{e}_1 - \Big(\mathbf{I} - \mathbf{B}_1\mathbf{e}_0 - \mathbf{C}_1^{-1}[\![\mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)]\!] + O\big((\mathbf{e}_0)_{i_1}(\mathbf{e}_0)_{i_2}\big)\Big) \\
&\mathbf{C}_1^{-1}\mathbf{C}_1\big(\mathbf{e}_1 + O\big(\mathbf{e}_1^2\big)\big) \\
&= \mathbf{B}_1\mathbf{e}_0\mathbf{e}_1 + \mathbf{C}_1^{-1}[\![\mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)]\!]\mathbf{e}_1 \\
&+ O\big((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2}\,(\mathbf{e}_0)_{i_3}\,(\mathbf{e}_0)_{i_4}\big) \\
&= \mathbf{B}_1\mathbf{e}_0\big(-\mathbf{C}_2\mathbf{e}_0^2 + \mathbf{B}_1\mathbf{e}_0^2 + \mathbf{C}_1^{-1}[\![\mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)]\!]\mathbf{e}_0\big) \\
&\ + \mathbf{C}_1^{-1}[\![\mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)]\!]\big(- \mathbf{C}_2\mathbf{e}_0^2 \\
&+ \mathbf{B}_1\mathbf{e}_0^2 + \mathbf{C}_1^{-1}[\![\mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)]\!]\mathbf{e}_0\big) + O\big((\mathbf{e}_0)_{i_1}\,(\mathbf{e}_0)_{i_2}\,(\mathbf{e}_0)_{i_3}\,(\mathbf{e}_0)_{i_4}\big)
\end{aligned} \tag{4.17}$$

It can be seen from (4.17) that it involves third order terms in $\mathbf{e}_0$. □

The proof of convergence when $m \geq 3$ can be carried via mathematical induction. Suppose the proposed iterative method (4.9) has convergence order s when $m =$

$s-1$. The error equation for s-step iterative method (4.9) can be written as

$$\begin{aligned}
\mathbf{e}_s &= \mathbf{e}_{s-1} - \mathbf{A}^{-1}\,\mathbf{F}(\mathbf{x}_{s-1}) \\
\mathbf{F}(\mathbf{x}_{s-1}) &= \mathbf{C}_1\big(\mathbf{e}_{s-1} + O(\mathbf{e}_{s-1}^2)\big) \\
\mathbf{e}_s &= \mathbf{e}_{s-1} - \big(\mathbf{I} - \mathbf{B}_1\mathbf{e}_0 - \mathbf{C}_1^{-1}[\![\mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)]\!] + O\big((\mathbf{e}_0)_{i_1}(\mathbf{e}_0)_{i_2}\big)\big) \\
&\quad \mathbf{C}_1^{-1}\mathbf{C}_1\big(\mathbf{e}_{s-1} + O(\mathbf{e}_{s-1}^2)\big) \\
&\quad = \mathbf{e}_{s-1} - \big(\mathbf{I} - \mathbf{B}_1\mathbf{e}_0 - \mathbf{C}_1^{-1}[\![\mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)]\!] \\
&\quad + O\big((\mathbf{e}_0)_{i_1}(\mathbf{e}_0)_{i_2}\big)\big)\big(\mathbf{e}_{s-1} + O(\mathbf{e}_{s-1}^2)\big) \\
&\quad = \big(\mathbf{B}_1\mathbf{e}_0 + \mathbf{C}_1^{-1}[\![\mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)]\!] + O\big((\mathbf{e}_0)_{i_1}(\mathbf{e}_0)_{i_2}\big)\big)\mathbf{e}_{s-1} \\
&\qquad - \big(\mathbf{I} - \mathbf{B}_1\mathbf{e}_0 - \mathbf{C}_1^{-1}[\![\mathbf{q}_1(\mathbf{x}) \odot (\boldsymbol{\gamma}_1\mathbf{C}_1\,\mathbf{e}_0)]\!] + O\big((\mathbf{e}_0)_{i_1}(\mathbf{e}_0)_{i_2}\big)\big) \\
&\qquad O\big(\mathbf{e}_{s-1}^2\big).
\end{aligned} \tag{4.18}$$

According to our assumption the order of convergence of $(s-1)-$step method is s. It means $\mathbf{e}_{s-1} = O\big((\mathbf{e}_0)_{i_1}(\mathbf{e}_0)_{i_2}\cdots(\mathbf{e}_0)_{i_s}\big)$. It is clearly evident from (4.18) that $\mathbf{e}_s = O\Big((\mathbf{e}_0)_{i_1}(\mathbf{e}_0)_{i_2}\cdots(\mathbf{e}_0)_{i_{s+1}}\Big)$. So $m-$step iterative method (4.9) has convergence order $m+1$.

4.4 Numerical simulations

It is important to verify the claimed order of convergence of the proposed iterative method. We adopt the following definition of computational order of convergence

$$\mathrm{CCO} = \frac{log\big(||\mathbf{F}(\mathbf{x}_{k+1})||_\infty/||\mathbf{F}(\mathbf{x}_k)||_\infty\big)}{log\big(||\mathbf{F}(\mathbf{x}_k)||_\infty/||\mathbf{F}(\mathbf{x}_{k-1})||_\infty\big)}. \tag{4.19}$$

To check the performance of our proposed iterative method, we solve three problems.

$$\text{Problem 1} = \begin{cases} x_i^2\,x_{i+1} - 1 = 0, & i = 1, 2\cdots, n-1 \\ x_n\,x_1 - 1 = 0, & i = n \end{cases}$$

$$\text{Problem } 2 = \begin{cases} F_1(\mathbf{x}) = (3 - 0.5\,x_1)\,x_1 - 2\,x_2 + 1 \\ F_2(\mathbf{x}) = (3 - 0.5\,x_n)\,x_n - 2\,x_{n-1} + 1 \\ F_i(\mathbf{x}) = (3 - 0.5\,x_i)\,x_i - x_{i-1} + 2\,x_{i+1} + 1, \;\; i = 2, 3, \cdots, n-1 \end{cases}$$

$$\text{Problem } 3 = \begin{cases} F_i(\mathbf{x}) = 2\left(n + i\,(1 - \cos(x_i)) - \sin(x_i) - \sum_{j=1}^{n} \cos(x_j)\right) \\ (2\,\sin(x_i) - \cos(x_i)), \;\; i = 1, 2, \cdots, n \end{cases}$$

The performance comparison is demonstrated between iterative methods (4.3) and (4.9). The first order divided difference operator (4.4) can be computed as

$$\begin{aligned}[\mathbf{x}, \mathbf{y}; \mathbf{F}]_{ij} = \Big(& F_i(y_1, y_2, \cdots, y_{j-1}, y_j, y_{j+1}, \cdots, x_n) \\ & - F_i(y_1, y_2, \cdots, y_{j-1}, x_j, x_{j+1}, \cdots, x_n)\Big)/(y_j - x_j),\end{aligned} \tag{4.20}$$

$\mathbf{x} = [x_1, x_2, \cdots, x_n]^T$ and $\mathbf{y} = [y_1, y_2, \cdots, y_n]^T$ and $i, j = 1, 2, \cdots, n$. If $\mathbf{F}(\mathbf{x})$ and $\mathbf{F}(\mathbf{y})$ are provided separately then number of scalar function evaluation in (4.20) are $n(n-1)$. The following Mathematica code can be used to compute the first order divided difference operator (4.20).

```
dF[x_, y_, F1_, F2_, n_] := (
M = Table[0, {i, 1, n}, {j, 1, n}];
Do[
z1 = y;
z1[[1]] = x[[1]];
dum = Fi[z1, i, n];
M[[i, 1]] = (dum - F2[[i]])/(x[[1]] - y[[1]]);
Do[
z1[[j]] = x[[j]];
dum1 = Fi[z1, i, n];
M[[i, j]] = (dum1 - dum)/(x[[j]] - y[[j]]);
dum = dum1;
, {j, 2, n - 1}];
M[[i, n]] = (F1[[i]] - dum)/(x[[n]] - y[[n]]);
, {i, 1, n}];
Return[M];
);
```

Tables 4.1, 4.2 and 4.3 clearly shows that the computational convergence orders confirm our claim that $m-$step iterative methods (4.9) and (4.3) have convergence order $m + 1$. In all test problems, the selection of preconditioners show that the performance of proposed iterative method (4.9) is better than competitor iterative

Method (4.9)					Method (4.3)	
$\mathbf{q}_1(\mathbf{x})$	$\mathbf{q}_2(\mathbf{F}(\mathbf{x}))$	m	$\Vert\mathbf{F}(\mathbf{x})\Vert_\infty$	CCO	$\Vert\mathbf{F}(\mathbf{x})\Vert_\infty$	CCO
$\mathbf{1}$	$-\mathbf{F}(\mathbf{x})$	1	$1.41e-46$	2.0	$9.12e-14$	2.0
$\mathbf{1}$	$-\mathbf{F}(\mathbf{x})+\mathbf{F}(\mathbf{x})^3/100$	1	$4.77e-52$	2.0	$9.12e-14$	2.0
$\mathbf{1}$	$-\mathbf{F}(\mathbf{x})$	2	$9.23e-220$	3.0	$4.24e-81$	3.0
$\mathbf{1}$	$-\mathbf{F}(\mathbf{x})+\mathbf{F}(\mathbf{x})^3/100$	2	$5.61e-245$	3.0	$4.24e-81$	3.0
$\mathbf{1}$	$-\mathbf{F}(\mathbf{x})$	3	$4.99e-754$	4.0	$3.63e-310$	4.0
$\mathbf{1}$	$-\mathbf{F}(\mathbf{x})+\mathbf{F}(\mathbf{x})^3/100$	3	$3.63e-827$	4.0	$3.63e-310$	4.0
$\mathbf{1}$	$-\mathbf{F}(\mathbf{x})$	4	$7.95e-2062$	5.0	$1.19e-900$	5.0
$\mathbf{1}$	$-\mathbf{F}(\mathbf{x})+\mathbf{F}(\mathbf{x})^3/100$	4	$6.44e-2225$	5.0	$1.19e-900$	5.0
$\mathbf{1}$	$-\mathbf{F}(\mathbf{x})$	5	$2.21e-4799$	6.0	$6.53e-2175$	6.0
$\mathbf{1}$	$-\mathbf{F}(\mathbf{x})+\mathbf{F}(\mathbf{x})^3/100$	5	$6.48e-5105$	6.0	$6.53e-2175$	6.0
$\sin(\mathbf{x})$	$-\mathbf{F}(\mathbf{x})$	5	$2.21e-4536$	6.0	$6.53e-2175$	6.0
$\cos(\mathbf{x})$	$-\mathbf{F}(\mathbf{x})$	5	$3.66e-2464$	6.0	$6.53e-2175$	6.0
$\exp(-\mathbf{x}/10)$	$-\mathbf{F}(\mathbf{x})$	5	$6.90e-5217$	6.0	$6.53e-2175$	6.0
$\mathbf{1}$	$-\sin(\mathbf{F}(\mathbf{x}))$	6	$4.56e-6550$	7.0	$4.79e-4608$	7.0
$\mathbf{1}$	$-\tan(\mathbf{F}(\mathbf{x}))$	6	$7.98e-5136$	7.0	$4.79e-4608$	7.0
$\mathbf{1}$	$-\mathbf{F}(\mathbf{x})/(\mathbf{1}+\mathbf{F}(\mathbf{x}))$	6	$4.34e-6513$	7.0	$4.79e-4608$	7.0
$\mathbf{1}$	$-\mathbf{F}(\mathbf{x})/(\mathbf{1}+\lvert\mathbf{F}(\mathbf{x})/100\rvert)$	6	$2.20e-10069$	7.0	$4.79e-4608$	7.0

TABLE 4.1: Problem 1: Initial guess: $x_i = 15/10$, $n = 10$, Iter= 5, $\beta = 1/100$

method (4.3). The computational cost of preconditioners is reasonable because we use diagonal preconditioners. We observed that the leading coefficients in all preconditioners are of order one or less than one in magnitude. By choosing properly preconditioners we obtain high accuracy in numerical results.

Method (4.9)					Method (4.3)	
$\mathbf{q}_1(\mathbf{x})$	$\mathbf{q}_2(\mathbf{F}(\mathbf{x}))$	m	$\Vert\mathbf{F}(\mathbf{x})\Vert_\infty$	CCO	$\Vert\mathbf{F}(\mathbf{x})\Vert_\infty$	CCO
$\mathbf{1}$	$\mathbf{F}(\mathbf{x})/10$	1	$6.29e-36$	2.0	$2.25e-24$	2.0
$\mathbf{1}$	$\mathbf{F}(\mathbf{x})/10-(\mathbf{F}(\mathbf{x})/10)^2+(\mathbf{F}(\mathbf{x})/10)^3$	1	$2.47e-41$	2.0	$2.25e-24$	2.0
$\mathbf{1}$	$\mathbf{F}(\mathbf{x})/10$	2	$5.04e-224$	3.0	$3.60e-161$	3.0
$\exp(-\mathbf{x}/10)$	$\mathbf{F}(\mathbf{x})/10$	2	$5.31e-231$	3.0	$3.60e-161$	3.0
$\mathbf{1}$	$\mathbf{F}(\mathbf{x})/10-(\mathbf{F}(\mathbf{x})/10)^2+(\mathbf{F}(\mathbf{x})/10)^3$	2	$4.72e-233$	3.0	$3.60e-161$	3.0
$\exp(-\mathbf{x}/10)$	$\mathbf{F}(\mathbf{x})/10-(\mathbf{F}(\mathbf{x})/10)^2+(\mathbf{F}(\mathbf{x})/10)^3$	2	$2.52e-243$	3.0	$3.60e-161$	3.0

TABLE 4.2: Problem 2: Initial guess: $x_i = -1$, $n = 100$, Iter= 5, $\beta = 1/100$

	Method (4.9)				Method (4.3)	
$\mathbf{q}_1(\mathbf{x})$	$\mathbf{q}_2(\mathbf{F}(\mathbf{x}))$	m	$\lVert\mathbf{F}(\mathbf{x})\rVert_\infty$	CCO	$\lVert\mathbf{F}(\mathbf{x})\rVert_\infty$	CCO
$\mathbf{1}$	$-\mathbf{F}(\mathbf{x})$	1	$1.49e-130$	2.0	$5.06e-73$	2.0
$\exp(\mathbf{x}/10)$	$-\mathbf{F}(\mathbf{x})$	1	$1.33e-137$	2.0	$5.06e-73$	2.0
$\mathbf{1}$	$-\mathbf{F}(\mathbf{x})-\mathbf{F}(\mathbf{x})^2/100$	1	$1.76e-146$	2.0	$5.06e-73$	2.0
$\mathbf{1}$	$-\mathbf{F}(\mathbf{x})-\mathbf{F}(\mathbf{x})^2/100-\mathbf{F}(\mathbf{x})^3/10000$	1	$1.87e-152$	2.0	$5.06e-73$	2.0
$\cos(\mathbf{x})$	$-\mathbf{F}(\mathbf{x})-\mathbf{F}(\mathbf{x})^2/100-\mathbf{F}(\mathbf{x})^3/10000$	1	$2.01e-116$	2.0	$5.06e-73$	2.0
$\sin(\mathbf{x})$	$-\mathbf{F}(\mathbf{x})-\mathbf{F}(\mathbf{x})^2/100-\mathbf{F}(\mathbf{x})^3/10000$	1	$7.53e-122$	2.0	$5.06e-73$	2.0
$\sinh(\mathbf{x})$	$-\mathbf{F}(\mathbf{x})-\mathbf{F}(\mathbf{x})^2/100-\mathbf{F}(\mathbf{x})^3/10000$	1	$3.72e-153$	2.0	$5.06e-73$	2.0
$\exp(\mathbf{x}/10)$	$-\mathbf{F}(\mathbf{x})-\mathbf{F}(\mathbf{x})^2/100-\mathbf{F}(\mathbf{x})^3/10000$	1	$1.62e-165$	2.0	$5.06e-73$	2.0

TABLE 4.3: Problem 3: Initial guess: $x_i = 1$, $n = 20$, Iter= 10, $\beta = 1/100$

4.5 Summary

The derivative-free iterative methods become important when the system of nonlinear equations is a black box and we compute Jacobian numerically. The multi-step iterative methods are efficient and provide a high order of convergence. The high efficiency of multi-step iterative methods is hidden in the fact that we repeatedly use LU factors of frozen Jacobian from the base method in the multi-step part. The computational cost that we pay is the per step single evaluation of the system of nonlinear equation and solution of lower and upper triangular systems. The proposed preconditioners offer high numerical accuracy in the computed solutions with very low computational cost. It can be seen that the embedding of our proposed preconditioners modifies the Jacobian without altering the convergence order with very low computational cost.

Chapter 5

A preconditioned iterative method for solving systems of nonlinear equations having unknown multiplicity

We present a modification to an existing iterative method for computing zeros with unknown multiplicities of nonlinear equations or systems of nonlinear equations. More precisely, we introduce preconditioners to nonlinear equations or systems of nonlinear equations and their corresponding Jacobians. The inclusion of preconditioners provides numerical stability and accuracy. The different selection of preconditioner produces a family of iterative methods. We modified an existing method in a way that we do not alter its inherited quadratic convergence. Numerical simulations confirm that the quadratic convergence order of the preconditioned iterative method is maintained. The influence of the considered preconditioners is clearly reflected numerically in the achieved accuracy of the computed solutions.

5.1 Introduction

The designing of an iterative method for solving nonlinear equations and system of nonlinear equations is an active area of research. Many researchers have proposed iterative methods for solving nonlinear and system of nonlinear equations for finding simple zeros or zeros with multiplicity greater than one [57, 62, 70, 72, 102, 107, 108, 119–126]. The classical iterative method for solving nonlinear and system of nonlinear equations to find simple zeros is the Newton method that offers quadratic convergence [1, 3] under certain conditions. When we are talking about the iterative method for solving nonlinear equations or system of nonlinear equations to find zeros with multiplicities greater than one, the classical Newton method requires a modification. The modified Newton method for finding zeros with multiplicity greater than one for nonlinear equations can be written as

$$\begin{cases} x_0 = \text{initial guess} \\ x_{k+1} = x_k - m\,\frac{\phi(x_k)}{\phi'(x_k)}, \quad k = 0, 1, \cdots . \end{cases} \tag{5.1}$$

Jose et al. [110] proposed the multidimensional version of of (5.1) as

$$\begin{cases} \mathbf{x}_0 = \text{initial guess} \\ \mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{F}'(\mathbf{x}_k)^{-1} \llbracket \mathbf{m} \rrbracket \, \mathbf{F}(\mathbf{x}_k), \quad k = 0, 1, \cdots , \end{cases} \tag{5.2}$$

where $\mathbf{m} = [m_1, m_2, \cdots, m_n]^T$ is a vector of multiplicities for system of nonlinear equations $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ and $\llbracket \cdot \rrbracket$ represents a diagonal matrix that keep the vector at its main diagonal. The proof of quadratic convergence of (5.2) is provided in [110]. W. Wu [109] proposed a variant of Newton method with the help of an auxiliary or a preconditioner exponential function. Suppose, we have a system of nonlinear equations $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ and we define a new system of nonlinear equation with a nonlinear preconditioner function that have the same root

$$\mathbf{U}(\mathbf{x}) = e^{\mathbf{v} \odot \mathbf{x}} \odot \mathbf{F}(\mathbf{x}) = \mathbf{0}, \tag{5.3}$$

where $\odot$ is the element-wise multiplication of two vectors. The application of Newton method for (5.3) is

$$\begin{aligned}
\mathbf{x}_{k+1} &= \mathbf{x}_k - \mathbf{U}'(\mathbf{x}_k)^{-1}\,\mathbf{U}(\mathbf{x}_k)\\
\mathbf{x}_{k+1} &= \mathbf{x}_k - \left(\llbracket e^{\mathbf{v}\odot\mathbf{x}_k}\rrbracket\,(\mathbf{F}'(\mathbf{x}_k) + \llbracket\mathbf{v}\odot\mathbf{F}(\mathbf{x}_k)\rrbracket)\right)^{-1}\, e^{\mathbf{v}\odot\mathbf{x}_k}\odot\mathbf{F}(\mathbf{x}_k)\\
\mathbf{x}_{k+1} &= \mathbf{x}_k - (\mathbf{F}'(\mathbf{x}_k) + \llbracket\mathbf{v}\odot\mathbf{F}(\mathbf{x}_k)\rrbracket)^{-1}\,\llbracket e^{\mathbf{v}\odot\mathbf{x}_k}\rrbracket^{-1} e^{\mathbf{v}\odot\mathbf{x}_k}\odot\mathbf{F}(\mathbf{x}_k)\\
\mathbf{x}_{k+1} &= \mathbf{x}_k - (\mathbf{F}'(\mathbf{x}_k) + \llbracket\mathbf{v}\odot\mathbf{F}(\mathbf{x}_k)\rrbracket)^{-1}\,\mathbf{F}(\mathbf{x}_k).
\end{aligned} \tag{5.4}$$

The rate of convergence of (5.4) is quadratic. A modification [110] in (5.1) is proposed by using a exponential preconditioner

$$\mathbf{U}(\mathbf{x}) = e^{\mathbf{v}\odot\mathbf{x}}\odot\mathbf{F}(\mathbf{x})^{1/\mathbf{m}} = \mathbf{0}, \tag{5.5}$$

where $1/\mathbf{m} = [1/m_1, 1/m_2, \cdots, 1/m_n]^T$ and power of $\mathbf{F}(\mathbf{x})$ is component-wise. The application of Newton method to (5.5) gives

$$\mathbf{x}_{k+1} = \mathbf{x}_k - (\mathbf{F}'(\mathbf{x}_k) + \llbracket\mathbf{v}\odot\mathbf{F}(\mathbf{x}_k)\rrbracket)^{-1}\,\llbracket\mathbf{m}\rrbracket\,\mathbf{F}(\mathbf{x}_k). \tag{5.6}$$

The original idea of nonlinear preconditioner function was proposed in [109]. Noor et al. [112] have proposed Newton method with general preconditioner. They defined a preconditioned system of nonlinear equations as follows

$$\mathbf{U}(\mathbf{x}) = \mathbf{G}(\mathbf{x})\odot\mathbf{F}(\mathbf{x}) = \mathbf{0}, \tag{5.7}$$

where $\mathbf{G}(\mathbf{x}) \neq \mathbf{0}$. Notice that the roots of $\mathbf{U}(\mathbf{x}) = \mathbf{0}$ and $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ are same because $\mathbf{G}(\mathbf{x}) \neq \mathbf{0}$ for all $\mathbf{x}$. The first order Fréchet derivative of (5.7) can be computed as

$$\begin{aligned}
&\Psi_i(\mathbf{x}) = \Phi_i(\mathbf{x})\ \Lambda_i(\mathbf{x})\\
&\nabla\Psi_i(\mathbf{x})^T = \Phi_i(\mathbf{x})\,\nabla\Lambda_i(\mathbf{x})^T + \Lambda_i(\mathbf{x})\,\nabla\Phi_i(\mathbf{x})^T,\ \ i = 1, 2, \cdots, n
\end{aligned} \tag{5.8}$$

$$\begin{bmatrix} \nabla\Psi_1(\mathbf{x})^T \\ \nabla\Psi_2(\mathbf{x})^T \\ \nabla\Psi_3(\mathbf{x})^T \\ \vdots \\ \nabla\Psi_n(\mathbf{x})^T \end{bmatrix} = \begin{bmatrix} \Phi_1(\mathbf{x}) & 0 & \cdots & 0 \\ 0 & \Phi_2(\mathbf{x}) & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \Phi_n(\mathbf{x}) \end{bmatrix} \begin{bmatrix} \nabla\Lambda_1(\mathbf{x})^T \\ \nabla\Lambda_2(\mathbf{x})^T \\ \nabla\Lambda_3(\mathbf{x})^T \\ \vdots \\ \nabla\Lambda_n(\mathbf{x})^T \end{bmatrix}$$
$$+ \begin{bmatrix} \Lambda_1(\mathbf{x}) & 0 & \cdots & 0 \\ 0 & \Lambda_2(\mathbf{x}) & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \Lambda_n(\mathbf{x}) \end{bmatrix} \begin{bmatrix} \nabla\Phi_1(\mathbf{x})^T \\ \nabla\Phi_2(\mathbf{x})^T \\ \nabla\Phi_3(\mathbf{x})^T \\ \vdots \\ \nabla\Phi_n(\mathbf{x})^T \end{bmatrix}$$

From (5.8), the Fréchet derivative of $\mathbf{F}(\mathbf{x}) \odot \mathbf{G}(\mathbf{x})$ is

$$\begin{aligned} (\mathbf{F}(\mathbf{x}) \odot \mathbf{G}(\mathbf{x}))' &= [\![\mathbf{F}(\mathbf{x})]\!]\mathbf{G}'(\mathbf{x}) + [\![\mathbf{G}(\mathbf{x})]\!]\mathbf{F}'(\mathbf{x}_k) \\ \mathbf{U}'(\mathbf{x}) &= [\![\mathbf{G}(\mathbf{x})]\!]\, \mathbf{F}'(\mathbf{x}_k) + [\![\mathbf{F}(\mathbf{x})]\!]\, \mathbf{G}'(\mathbf{x}) \\ \mathbf{U}'(\mathbf{x}) &= [\![\mathbf{G}(\mathbf{x})]\!] \left(\mathbf{F}'(\mathbf{x}_k) + [\![\mathbf{F}(\mathbf{x})]\!]\, [\![\mathbf{G}(\mathbf{x})]\!]^{-1}\, \mathbf{G}'(\mathbf{x})\right). \end{aligned} \tag{5.9}$$

If we apply Newton method to (5.7), we obtain

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + [\![\mathbf{F}(\mathbf{x}_k)]\!]\, [\![\mathbf{G}(\mathbf{x})]\!]^{-1}\, \mathbf{G}'(\mathbf{x})\right)^{-1} [\![\mathbf{G}(\mathbf{x})]\!]^{-1}\, \mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}_k) \\ \mathbf{x}_{k+1} &= \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + [\![\mathbf{F}(\mathbf{x})]\!]\, [\![\mathbf{G}(\mathbf{x})]\!]^{-1}\mathbf{G}'(\mathbf{x})\right)^{-1} \mathbf{F}(\mathbf{x}_k). \end{aligned} \tag{5.10}$$

The convergence order of (5.10) is two. The iterative method (5.6) with general preconditioner can be written as

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\mathbf{F}'(\mathbf{x}_k) + [\![\mathbf{F}(\mathbf{x}_k)]\!]\, [\![\mathbf{G}(\mathbf{x})]\!]^{-1}\mathbf{G}'(\mathbf{x})\right)^{-1} [\![\mathbf{m}]\!]\, \mathbf{F}(\mathbf{x}_k). \tag{5.11}$$

The convergence order of (5.11) is also two. The modified Newton method [1, 7, 8] for solving nonlinear equations with unknown multiplicity can be developed in this way. We define a new function

$$s(x) = \frac{\phi(x)}{\phi'(x)}. \tag{5.12}$$

The application of Newton method to (5.12) gives

$$\begin{aligned} x_{k+1} &= x_k - \frac{s(x_k)}{s'(x_k)} \\ x_{k+1} &= x_k - \frac{\phi'(x_k)\phi(x_k)}{\phi'(x_k)^2 - \phi''(x_k)\,\phi(x_k)}. \end{aligned} \tag{5.13}$$

The order of convergence of (5.13) is two. Noor and his co-researchers [111] have constructed a family of the method for solving nonlinear equations with unknown multiplicity by introducing a preconditioner. They defined a new function

$$q(x) = \frac{\phi(x)\,\lambda(x)}{\phi'(x)} \tag{5.14}$$

and application of Newton method to (5.14) gives

$$\begin{aligned} x_{k+1} &= x_k - \frac{q(x_k)}{q'(x_k)} \\ x_{k+1} &= x_k - \frac{\phi'(x_k)\,\phi(x_k)\,\lambda(x_k)}{\phi'(x_k)(\phi(x_k)\,\lambda(x_k))' - \phi''(x_k)\,\phi(x_k)\,\lambda(x_k)}, \end{aligned} \tag{5.15}$$

where $\lambda(x)$ is a non-zero function. The order of convergence of (5.15) is two.

5.2 Proposed method

When we observe (5.14), we can notice that the preconditioner is only introduced for $\phi(x)$ but not for $\phi'(x)$. We will also introduce a preconditioner for $\phi'(x)$ and will show that the convergence order (5.14) is still quadratic. We define a new function

$$q(x) = \frac{\lambda(x)\,\phi(x)}{(\omega(x)\,\phi(x))'}, \tag{5.16}$$

and after applying Newton method, we obtain

$$\begin{aligned} x_{k+1} &= x_k - \frac{q(x_k)}{q'(x_k)} \\ x_{k+1} &= x_k - \frac{(\omega(x_k)\,\phi(x_k))'\,\lambda(x_k)\,\phi(x_k)}{(\omega(x_k)\,\phi(x_k))'\,(\lambda(x_k)\,\phi(x_k))' - \underline{(\omega(x_k)\,\phi(x_k))''\,\lambda(x_k)\,\phi(x_k)}}, \end{aligned} \tag{5.17}$$

where $\omega(x)$ is a non-zero function. For the purpose of generalization of iterative method (5.17) to system of nonlinear equations, we define a new function $\mathbf{Q}(\mathbf{x})$

$$\mathbf{Q}(\mathbf{x}) = \big((\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\big)^{-1}\,(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x})) = \mathbf{0}. \tag{5.18}$$

The first order Fréchet derivative of (5.18) can be written as

$$\begin{aligned} \mathbf{Q}'(\mathbf{x}) =& \Big(\big((\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\big)^{-1}\Big)^2 \Big((\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\,\Big(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x})\Big)' \\ &- (\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\,(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))''\,\big((\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\big)^{-1} \\ &(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))\Big). \end{aligned} \tag{5.19}$$

Further simplification of $\mathbf{Q}'(\mathbf{x})^{-1}\,\mathbf{Q}(\mathbf{x})$ gives

$$\begin{aligned} \mathbf{Q}'(\mathbf{x})^{-1}\,\mathbf{Q}(\mathbf{x}) =& \Big(\,(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\,(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))' \\ &- \underline{(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\,(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))''\,\big((\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\big)^{-1}} \\ &(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))\,\Big)(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\,(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))\,. \end{aligned} \tag{5.20}$$

If compare underline expressions in (5.17) and (5.20), They are different. Generally, It is not possible to commute $(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'$ with $(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))''$. But, we artificially eliminate terms $(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'$ and $\big((\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\big)^{-1}$ from expression $(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\,(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))''\,\big((\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\big)^{-1}$ and get the following iterative method.

$$\begin{aligned} \mathbf{x}_{k+1} =& \mathbf{x}_k - \Big(\,(\mathbf{H}(\mathbf{x}_k) \odot \mathbf{F}(\mathbf{x}_k))'\,(\mathbf{G}(\mathbf{x}_k) \odot \mathbf{F}(\mathbf{x}_k))' \\ &- (\mathbf{H}(\mathbf{x}_k) \odot \mathbf{F}(\mathbf{x}_k))''\,(\mathbf{G}(\mathbf{x}_k) \odot \mathbf{F}(\mathbf{x}_k))\Big)^{-1} \\ &(\mathbf{H}(\mathbf{x}_k) \odot \mathbf{F}(\mathbf{x}_k))'\,(\mathbf{G}(\mathbf{x}_k) \odot \mathbf{F}(\mathbf{x}_k))\,. \end{aligned} \tag{5.21}$$

It can be seen that the iterative method (5.21) is not the application of Newton method to (5.18). The iterative method (5.17) for solving scalar nonlinear equations with unknown multiplicity and vector version (5.21) are exactly same. We will only provide the proof of quadratic convergence for (5.21) and it is applicable automatically to scalar version (5.17). An iterative method was proposed in [127] to compute the zeros with multiplicity of system of nonlinear equations that used preconditioners for system of nonlinear equations but not for Jacobian of the system of nonlinear equations. Notice that we are introducing preconditioners for the system of nonlinear equations as well as Jacobian of the system of nonlinear equations.

5.3 Convergence

In the following theorem, we established the proof of quadratic convergence of (5.21).

Theorem 5.1. *Let* $\mathbf{F} : D \subseteq \mathbb{R}^n \longrightarrow \mathbb{R}^n$ *and* $\boldsymbol{\kappa} = [\kappa_1, \kappa_2, \kappa_3, \cdots, \kappa_n]^T \in D$ *is a root of* $\mathbf{F}(\mathbf{x}) = (\mathbf{x} - \boldsymbol{\kappa})^{\mathbf{m}} \odot \mathbf{P}(\mathbf{x}) = \mathbf{0}$ *with corresponding multiplicities vector* $\mathbf{m} = [m_1, m_2, \cdots, m_n]^T$ *and non-zero function* $\mathbf{P} = [p_1(\mathbf{x}), p_2(\mathbf{x}), \cdots, p_n(\mathbf{x})]^T$ *with* $p_i(\mathbf{x})(\neq \mathbf{0}) \in C^2(D)$*. Then there exists a subset* $E \subseteq D$ *such that, if we choose* $\mathbf{x}_0 \in E$ *, the iterative method (5.21) has quadratic convergence in* E*.*

Proof. Let $\mathbf{e} = \mathbf{x} - \boldsymbol{\kappa}$ then $\mathbf{F}(\mathbf{x}) = \mathbf{e}^{\mathbf{m}} \odot \mathbf{P}(\mathbf{x})$. Whenever we take vector power of a vector, it is always component-wise. So $\mathbf{e}^{\mathbf{m}} = [\epsilon_1^{m_1}, \epsilon_2^{m_2}, \cdots, \epsilon_n^{m_n}]^T$. The first order Fréchet derivative of $\mathbf{F}(\mathbf{x})$ is

$$\mathbf{F}'(\mathbf{x}) = [\![\mathbf{m} \odot \mathbf{e}^{\mathbf{m}-\mathbf{1}} \odot \mathbf{P}(\mathbf{x})]\!] + [\![\mathbf{e}^{\mathbf{m}}]\!]\mathbf{P}'(\mathbf{x}). \tag{5.22}$$

The expressions for terms in (5.21) are computed as follows.

$$\begin{aligned}(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))' &= [\![\mathbf{m} \odot \mathbf{e}^{\mathbf{m}-\mathbf{1}} \odot \mathbf{P}(\mathbf{x}) \odot \mathbf{H}(\mathbf{x})]\!] + [\![\mathbf{e}^{\mathbf{m}} \odot \mathbf{H}(\mathbf{x})]\!]\mathbf{P}'(\mathbf{x}) \\ &+ [\![\mathbf{e}^{\mathbf{m}} \odot \mathbf{P}(\mathbf{x})]\!]\mathbf{H}'(\mathbf{x}) \qquad (5.23) \\ (\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))' &= [\![\mathbf{m} \odot \mathbf{e}^{\mathbf{m}-\mathbf{1}} \odot \mathbf{P}(\mathbf{x}) \odot \mathbf{G}(\mathbf{x})]\!] + [\![\mathbf{e}^{\mathbf{m}} \odot \mathbf{G}(\mathbf{x})]\!]\mathbf{P}'(\mathbf{x}) \\ &+ [\![\mathbf{e}^{\mathbf{m}} \odot \mathbf{P}(\mathbf{x})]\!]\mathbf{G}'(\mathbf{x}) \qquad (5.24)\end{aligned}$$

By using (5.23) and (5.24), we can write the product of $(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'$ and $(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'$ as

$$\begin{aligned}(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\,(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))' =& \big(\mathbf{m}^2 \odot \mathbf{e}^{\mathbf{2m-2}} \odot \mathbf{P}(\mathbf{x})^2 \odot \mathbf{H}(\mathbf{x}) \odot \mathbf{G}(\mathbf{x})\big) \\ &+ O\big([\![\mathbf{e}^{\mathbf{2m-1}}]\!]\big).\end{aligned} \tag{5.25}$$

Next, we compute second order Fréceht derivative of $(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'$. Let $\boldsymbol{\phi}$ be a scalar vector of length n

$$(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\boldsymbol{\phi} = \mathbf{m} \odot \mathbf{e}^{\mathbf{m-1}} \odot \mathbf{P}(\mathbf{x}) \odot \mathbf{H}(\mathbf{x}) \odot \boldsymbol{\phi} + \mathbf{A}(\boldsymbol{\phi}), \tag{5.26}$$

where $\mathbf{A}(\boldsymbol{\phi}) = [\![\mathbf{e}^{\mathbf{m}} \odot \mathbf{H}(\mathbf{x})]\!]\mathbf{P}'(\mathbf{x})\boldsymbol{\phi} + [\![\mathbf{e}^{\mathbf{m}} \odot \mathbf{P}(\mathbf{x})]\!]\mathbf{H}'(\mathbf{x})\boldsymbol{\phi}$. We again compute the first order Fréchet derivative of (5.26)

$$\begin{aligned}&(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))''\boldsymbol{\phi} = [\![\mathbf{m} \odot (\mathbf{m}-\mathbf{1}) \odot \mathbf{e}^{\mathbf{m-1}} \odot \boldsymbol{\phi} \odot \mathbf{P}(\mathbf{x}) \odot \mathbf{H}(\mathbf{x})]\!] \\ &+ [\![\mathbf{m} \odot \mathbf{e}^{\mathbf{m-1}} \odot \boldsymbol{\phi}]\!](\mathbf{P}(\mathbf{x}) \odot \mathbf{H}(\mathbf{x}))' + \mathbf{A}'(\boldsymbol{\phi}) \\ &(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))''(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x})) \\ &= [\![\mathbf{m} \odot (\mathbf{m}-\mathbf{1}) \odot \mathbf{e}^{\mathbf{2m-2}} \odot \mathbf{P}(\mathbf{x})^2 \odot \mathbf{H}(\mathbf{x})]\!] \\ &+ [\![\mathbf{m} \odot \mathbf{e}^{\mathbf{2m-1}} \odot \mathbf{P}(\mathbf{x})]\!](\mathbf{P}(\mathbf{x}) \odot \mathbf{H}(\mathbf{x}))'\mathbf{A}'(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x})) \\ &(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))''(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x})) = \\ &[\![\mathbf{m} \odot (\mathbf{m}-\mathbf{1}) \odot \mathbf{e}^{\mathbf{2m-2}} \odot \mathbf{P}(\mathbf{x})^2 \odot \mathbf{H}(\mathbf{x})]\!] + O\big([\![\mathbf{e}^{\mathbf{2m-1}}]\!]\big).\end{aligned} \tag{5.27}$$

By subtracting (5.27) from (5.25), we obtain

$$\begin{aligned}&(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\,(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))' - (\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))''(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x})) \\ &= [\![\mathbf{m} \odot \mathbf{e}^{\mathbf{2m-2}} \odot \mathbf{P}(\mathbf{x})^2 \odot \mathbf{H}(\mathbf{x})]\!](\mathbf{I} + O([\![\mathbf{e}]\!])). \\ &\big((\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'\,(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))' - (\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))''(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))\big)^{-1} \\ &= (\mathbf{I} - O([\![\mathbf{e}]\!]))\left([\![\mathbf{m} \odot \mathbf{e}^{\mathbf{2m-2}} \odot \mathbf{P}(\mathbf{x})^2 \odot \mathbf{H}(\mathbf{x})]\!]\right)^{-1}.\end{aligned} \tag{5.28}$$

By using (5.26), the expression for $(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))$ is

$$\begin{aligned}&(\mathbf{H}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x}))'(\mathbf{G}(\mathbf{x}) \odot \mathbf{F}(\mathbf{x})) \\ &= \mathbf{m} \odot \mathbf{e}^{\mathbf{2m-1}} \odot \mathbf{P}(\mathbf{x})^2 \odot \mathbf{H}(\mathbf{x}) \odot (\mathbf{1} + O(\mathbf{e})).\end{aligned} \tag{5.29}$$

From (5.28) and (5.29), we get

$$\begin{aligned}
&\big((\mathbf{H}(\mathbf{x})\odot\mathbf{F}(\mathbf{x}))'\,(\mathbf{G}(\mathbf{x})\odot\mathbf{F}(\mathbf{x}))'-(\mathbf{H}(\mathbf{x})\odot\mathbf{F}(\mathbf{x}))''(\mathbf{G}(\mathbf{x})\odot\mathbf{F}(\mathbf{x}))\big)^{-1}\\
&(\mathbf{H}(\mathbf{x})\odot\mathbf{F}(\mathbf{x}))'(\mathbf{G}(\mathbf{x})\odot\mathbf{F}(\mathbf{x}))\\
&=(\mathbf{I}-O(\llbracket\mathbf{e}\rrbracket))\Big(\llbracket\mathbf{m}\odot\mathbf{e}^{\mathbf{2m-2}}\odot\mathbf{P}(\mathbf{x})^2\odot\mathbf{H}(\mathbf{x})\rrbracket\Big)^{-1} \qquad (5.30)\\
&\mathbf{m}\odot\mathbf{e}^{\mathbf{2m-1}}\odot\mathbf{P}(\mathbf{x})^2\odot\mathbf{H}(\mathbf{x})\odot(\mathbf{1}+O(\mathbf{e}))\\
&=(\mathbf{I}-O(\llbracket\mathbf{e}\rrbracket))\llbracket\mathbf{e}\rrbracket(\mathbf{1}+O(\mathbf{e}))=\mathbf{e}+O(\mathbf{e}^2).
\end{aligned}$$

The error equation for (5.21) can be written as

$$\mathbf{e}_{k+1}=\mathbf{e}_k-\big(\mathbf{e}_k+O\big(\mathbf{e}_k^2\big)\big)=O\big(\mathbf{e}_k^2\big) \qquad (5.31)$$

The error equation(5.31) for (5.21) tells that the order of convergence for the proposed iterative method is quadratic. □

5.4 Numerical testing

The two preconditioners $\omega(x)$ and $\lambda(x)$ produce families of iterative methods. If we define $\omega(x)=exp(\varpi\, z)$ and $\lambda(x)=exp(\vartheta\, z)$, we get the following two parameter family of iterative method for solving nonlinear equations have zeros with unknown multiplicity.

$$\text{S1: } x_{k+1}=x_k-\frac{(\varpi\,\phi(x)+\phi'(x))\;\phi(x)}{(\vartheta-\varpi)\,\phi'(x)\,(\varpi+\phi(x))\,\phi'(x)^2-\phi''(x)\,\phi(x)}$$

Now we choose $\omega(x)=exp(\varpi\,\phi(x))$ and $\lambda(x)=(\vartheta\,\phi(x))$ and obtain the following method

$$\text{S2: } x_{k+1}=x_k-\frac{\phi'(x)\,(1+\varpi\,\phi(x))\,\phi(x)}{\phi'(x)^2\,(1+(\vartheta-\varpi)\,(1+\varpi\,\phi(x))\,\phi(x))-\phi''(x)\,\phi(x)\,(1+\varpi\,\phi(x))}.$$

We only conducted numerical testing for the system of nonlinear equation and the cases for the nonlinear equations are similar. It is important to test the computational convergence order (CCO) of proposed iterative methods. In all our simulations, we adopt the following definition of CCO [68, 69]

$$\text{CCO} = \frac{log\big(||\mathbf{F}(\mathbf{x}_{k+1})||_\infty/||\mathbf{F}(\mathbf{x}_k)||_\infty\big)}{log\big(||\mathbf{F}(\mathbf{x}_k)||_\infty/||\mathbf{F}(\mathbf{x}_{k-1})||_\infty\big)} \text{ or } \frac{log\big(||\mathbf{x}_{k+1}-\boldsymbol{\kappa}||_\infty/||\mathbf{x}_k-\boldsymbol{\kappa}||_\infty\big)}{log\big(||\mathbf{x}_k-\boldsymbol{\kappa}||_\infty/||\mathbf{x}_{k-1}-\boldsymbol{\kappa}||_\infty\big)}. \quad (5.32)$$

For numerical simulations, three problems are selected with different multiplicities. The performance of iterative method (5.11) is not better comparatively. The various choices for the preconditioners are made in Tables 5.1, 5.2, 5.3 for all three problems. In Table 5.1, we have shown that the selection of preconditioners has an influence on the numerical accuracy of computed zeros with multiplicities. Moreover, the computational cost of performing the different operation is reasonable because, in all cases, we selected preconditioners in a way that their first and second order Fréchet derivatives are diagonal matrices. When we select $\mathbf{G}(\mathbf{x}) = 6 + \cos(\mathbf{x})/10$ and $\mathbf{H}(\mathbf{x}) = 6 + \cos(\mathbf{x})/10$ for Problem 1, we achieved the best accuracy in computed zeros with different multiplicities. For the second problem, Table 5.2 shows that the selection of $\mathbf{G}(\mathbf{x})$ produces good accuracy. In Table 5.3, again the selection of both preconditioners provides the best accuracy comparatively.

$$\text{Problem 1} = \begin{cases} \Phi_1(\mathbf{x}) = (x_1-1)^4 \exp(x_2) = 0 \\ \Phi_2(\mathbf{x}) = (x_2-2)^5\,(x_1\,x_2-1) = 0 \\ \Phi_3(\mathbf{x}) = (x_3+4)^6 = 0. \end{cases} \quad (5.33)$$

$$\text{Problem 2} = \begin{cases} \Phi_1(\mathbf{x}) = x_1\,x_2 = 0 \\ \Phi_2(\mathbf{x}) = x_2\,x_3 = 0 \\ \Phi_3(\mathbf{x}) = x_3\,x_4 = 0 \\ \Phi_4(\mathbf{x}) = x_4\,x_1 = 0 \end{cases} \quad (5.34)$$

$$\text{Problem 3} = \begin{cases} \Phi_1(\mathbf{x}) = \sqrt{x_1-1}\,x_2\,x_3 = 0 \\ \Phi_2(\mathbf{x}) = \sqrt{x_2-1}\,x_1\,x_3 = 0 \\ \Phi_3(\mathbf{x}) = \sqrt{x_3-1}\,x_1\,x_2 = 0 \end{cases} \quad (5.35)$$

	$\mathbf{G}(\mathbf{x})$	$\mathbf{H}(\mathbf{x})$	Iter	$\|\|\mathbf{x}-\boldsymbol{\kappa}\|\|_\infty$	CCO
Iterative method (5.21)	$\mathbf{1}$	$\mathbf{1}$	6	$O\left(10^{-43}\right)$	2.0
	$6+\cos(\mathbf{x})/10$	$\mathbf{1}$	6	$O\left(10^{-51}\right)$	2.05
	$1+\mathbf{x}^3/1000$	$\mathbf{1}$	6	$O\left(10^{-42}\right)$	2.0
	$\exp(-\mathbf{x}/100)$	$\mathbf{1}$	6	$O\left(10^{-46}\right)$	2.0
	$\mathbf{1}$	$6+\cos(\mathbf{x})/10$	6	$O\left(10^{-38}\right)$	2.0
	$\mathbf{1}$	$1+\mathbf{x}^3/1000$	6	$O\left(10^{-46}\right)$	2.0
	$\mathbf{1}$	$\exp(-\mathbf{x}/100)$	6	$O\left(10^{-39}\right)$	2.0
	$6+\cos(\mathbf{x})/10$	$6+\cos(\mathbf{x})/10$	6	$O\left(10^{-41}\right)$	2.0
	$6+\cos(\mathbf{x})/10$	$1+\mathbf{x}^3/1000$	6	$O\left(10^{-65}\right)$	2.0
	$6+\cos(\mathbf{x})/10$	$\exp(-\mathbf{x}/100)$	6	$O\left(10^{-43}\right)$	2.0
	$1+\mathbf{x}^3/1000$	$1+\mathbf{x}^3/1000$	6	$O\left(10^{-45}\right)$	2.0
	$1+\mathbf{x}^3/1000$	$6+\cos(\mathbf{x})/10$	6	$O\left(10^{-37}\right)$	2.0
	$1+\mathbf{x}^3/1000$	$\exp(-\mathbf{x}/100)$	6	$O\left(10^{-38}\right)$	2.0
	$\exp(-\mathbf{x}/100)$	$\exp(-\mathbf{x}/100)$	6	$O\left(10^{-41}\right)$	2.0
	$\exp(-\mathbf{x}/100)$	$\exp(\mathbf{x}/100)$	6	$O\left(10^{-53}\right)$	2.0
	$\exp(-\mathbf{x}/100)$	$6+\cos(\mathbf{x})/10$	6	$O\left(10^{-40}\right)$	2.0
	$\exp(-\mathbf{x}/100)$	$1+\mathbf{x}^3/1000$	6	$O\left(10^{-53}\right)$	2.0
Iterative method (5.11)	$\mathbf{1}$	-	6	$O\left(10^{-30}\right)$	2.0
	$6+\cos(\mathbf{x})/10$	-	6	$O\left(10^{-30}\right)$	2.0
	$1+\mathbf{x}^3/1000$	-	6	$O\left(10^{-30}\right)$	2.0
	$\exp(\mathbf{x}/100)$	-	6	$O\left(10^{-30}\right)$	2.0

Table 5.1: Problem 1: initial guess $= [2, 1, -2]$, $m = [4, 5, 6]$

	$\mathbf{G}(\mathbf{x})$	$\mathbf{H}(\mathbf{x})$	Iter	$\|\|\mathbf{F}(\mathbf{x})\|\|_\infty$	CCO
Iterative method (5.21)	$\mathbf{1}$	$\mathbf{1}$	1	-	-
	$6+\cos(\mathbf{x})/10$	$\mathbf{1}$	7	$O\left(10^{-2042}\right)$	3.0
	$1+\mathbf{x}^3/1000$	$\mathbf{1}$	7	$O\left(10^{-8482}\right)$	3.98
	$\exp(\mathbf{x}/100)$	$\mathbf{1}$	7	$O\left(10^{-376}\right)$	2.00
Iterative method (5.11)	$\mathbf{1}$	-	1	-	-
	$6+\cos(\mathbf{x})/10$	-	20	$O\left(10^{-23}\right)$	1.0
	$1+\mathbf{x}^3/1000$	-	20	Not converging	-
	$\exp(\mathbf{x}/100)$	-	7	$O\left(10^{-443}\right)$	2.0

Table 5.2: Problem 2: initial guess $= [1, 2, 4, 3]$, $m = [2, 2, 2, 2]$

5.5 Summary

The inclusion of preconditioners in the existing iterative methods for finding zeros with multiplicities for solving system of nonlinear equations gives benefit in numerical stability and numerical accuracy. The proposed methodology is equally effective for nonlinear and system of nonlinear equations. It is assumed in all

	$\mathbf{G}(\mathbf{x})$	$\mathbf{H}(\mathbf{x})$	Iter	$\|\|\mathbf{F}(\mathbf{x})\|\|_\infty$	CCO
Iterative method (5.21)	$\mathbf{1}$	$\mathbf{1}$	12	$O\left(10^{-2011}\right)$	2.00
	$6+\cos(\mathbf{x})/10$	$\mathbf{1}$	12	$O\left(10^{-1914}\right)$	2.00
	$1+\mathbf{x}^3/1000$	$\mathbf{1}$	12	$O\left(10^{-1248}\right)$	2.00
	$\exp(-\mathbf{x}/10)$	$\mathbf{1}$	12	$O\left(10^{-2767}\right)$	2.00
	$\exp(-\mathbf{x}/10)$	$\exp(\mathbf{x}/10000)$	12	$O\left(10^{-2110}\right)$	2.00
	$\exp(-\mathbf{x}/10)$	$\exp(-\mathbf{x}/10000)$	12	$O\left(10^{-2771}\right)$	2.00
Iterative method (5.11)	$\mathbf{1}$	-	1	-	-
	$6+\cos(\mathbf{x})/10$	-	12	$O\left(10^{-56}\right)$	2.00
	$1+\mathbf{x}^3/1000$	-	20	Not converging	-
	$\exp(-\mathbf{x}/10)$	-	7	$O\left(10^{-35}\right)$	2.00

TABLE 5.3: Problem 3: initial guess = $[2,4,3]$, $m=[1/2,1/2,1/2]$

cases that the preconditioners should be non-zero because in this way, it does not affect the zeros of nonlinear or system of nonlinear equations. The different selections of preconditioners provide different families of iterative methods. The claimed order of convergence is also verified by computing the computational order of convergence in all numerical simulations. To study the dynamics of nonlinear preconditioners for finding zeros with multiplicities of nonlinear equations and system of nonlinear equations could be an interesting topic for research.

Chapter 6

Higher order multi-step Jarratt-like method for solving systems of nonlinear equations: Application to PDEs and ODEs

A multi-step iterative method for solving systems of nonlinear equations with a local convergence order of $3m$–4, where m ($\geq$ 2) is the number of steps, is proposed. The multi-step iterative method includes two components: the base method and the multi-step part. The base method involves two function evaluations, two Jacobian evaluations, one LU decomposition of a Jacobian, and two matrix-vector multiplications. Every stage of the multi-step part involves the solution of two triangular linear systems and one matrix vector multiplication. The computational efficiency of the new method is better than that of previously proposed methods. The method is applied to several nonlinear problems stemming from the numerical approximation of nonlinear ordinary differential equations and of nonlinear partial differential equations.

6.1 Introduction

A multi-step iterative method includes the base method and the multi-step part. The base method is designed to be computationally efficient. Since matrix inversion is an expensive operation, most base methods involve only one inversion of the Jacobian matrix. In the multi-step part, systems of linear equations should be solved by using the inverse of the Jacobian matrix computed in the base method. For instance, one can use the LU-factors of the Jacobian matrix. Other expensive operations which should be minimized in multi-step iterative methods include functions and Jacobian evaluations, matrix vector multiplications, vector vector multiplications, and solutions of systems of linear equations. A multi-step method that offers an increment of two in the convergence order of base method can be found in [67]. Recently, Malik et al. [56] have constructed a general class of multi-step iterative methods with two matrix inversions in the base method, making those methods expensive. The method proposed in [65] also involves two matrix inversions in the base method. Independent recent efforts by two different research groups have resulted [63, 102] in the same multi-step iterative method. We will improve the higher order multi-step Jarratt-like (HJ) method method described in [63, 102]. That method can be described, taking note of the convergence order and the computational cost, as

$$
\mathrm{HJ} = \left\{
\begin{array}{ll}
\text{Number of steps} & = m \geq 2 \\
\text{Convergence order} & = 2m \\
\text{Function evaluations} & = m-1 \\
\text{Jacobian evaluations} & = 2 \\
\text{LU decomposition} & = 1 \\
\text{Matrix vector multiplications} & = m \\
\text{Vector vector multiplications} & = 2m \\
\text{Number of solutions of systems} & \\
\text{of lower and upper triangular} & \\
\text{systems of equations} & = 2m-1
\end{array}
\right.
\left\{
\begin{array}{ll}
\text{Base method} \longrightarrow & \left\{
\begin{array}{l}
\mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_1 = \mathbf{F}(\mathbf{x}_k) \\
\mathbf{y}_1 = \mathbf{x}_k - \frac{2}{3}\,\boldsymbol{\phi}_1 \\
\mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_2 = \mathbf{F}'(\mathbf{y}_1)\,\boldsymbol{\phi}_1 \\
\mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_3 = \mathbf{F}'(\mathbf{y}_1)\,\boldsymbol{\phi}_2 \\
\mathbf{y}_2 = \mathbf{x}_k - \frac{23}{8}\,\boldsymbol{\phi}_1 + 3\,\boldsymbol{\phi}_2 - \frac{9}{8}\,\boldsymbol{\phi}_3
\end{array}
\right. \\
\text{Multi-step part} \rightarrow & \left\{
\begin{array}{l}
\text{for } s = 1, m-2 \\
\quad \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_{2s+2} = \mathbf{F}(\mathbf{y}_{s+1}) \\
\quad \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_{2s+3} = \mathbf{F}'(\mathbf{y}_1)\,\boldsymbol{\phi}_{2s+2} \\
\quad \mathbf{y}_{s+2} = \mathbf{y}_{s+1} - \frac{5}{2}\,\boldsymbol{\phi}_{2s+2} + \frac{3}{2}\,\boldsymbol{\phi}_{2s+3} \\
\text{end}
\end{array}
\right.
\end{array}
\right.
$$

where $\mathbf{F}'(\cdot)$ denotes the Frechet derivative [62] or the Jacobian of $\mathbf{F}(\cdot)$. The base method in HJ has a convergence order of four and involves one LU decomposition, one function evaluation, and two Jacobian evaluations. Each step of the multi-step part increases the convergence order by two. In 2015, Malik et al. [70] developed another efficient multi-step iterative method (MSF) for solving nonlinear systems arising from particular ODEs which can be described as

$$\text{MSF} = \begin{cases} \text{Number of steps} & = m \\ \text{Convergence-order} & = 3m \\ \text{Function evaluations} & = m \\ \text{Jacobian evaluations} & = 2 \\ \text{Second-order Fréchet derivative} & = 1 \\ \text{LU decomposition} & = 1 \\ \text{Matrix vector multiplications} & = 2m-2 \\ \text{Vector vector multiplications} & = m+2 \\ \text{Number of solutions of systems} & \\ \text{of lower and upper triangular} & \\ \text{ systems of equations} & = 3m-1 \end{cases} \begin{cases} \text{Base method} \longrightarrow \begin{cases} \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_1 = \mathbf{F}(\mathbf{x}_k) \\ \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_2 = \mathbf{F}''(\mathbf{x}_k)\,\boldsymbol{\phi}_1^2 \\ \mathbf{y}_1 = \mathbf{x}_k - \boldsymbol{\phi}_1 - \frac{1}{2}\,\boldsymbol{\phi}_2 \end{cases} \\ \text{Multi-step part} \rightarrow \begin{cases} \text{for } s = 1, m-1 \\ \quad \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_{3s} = \mathbf{F}(\mathbf{y}_s) \\ \quad \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_{3s+1} = \mathbf{F}'(\mathbf{y}_1)\,\boldsymbol{\phi}_{3s} \\ \quad \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_{3s+2} = \mathbf{F}'(\mathbf{y}_1)\,\boldsymbol{\phi}_{3s+1} \\ \quad \mathbf{y}_{s+1} = \mathbf{y}_s - 3\,\boldsymbol{\phi}_{3s} + 3\,\boldsymbol{\phi}_{3s+1} \\ \qquad\quad -\boldsymbol{\phi}_{3s+2} \\ \text{end} \end{cases} \end{cases}$$

The limitation of MSF is that it was only constructed for a particular class of ordinary differential equations (ODEs) of the form $L(x(t)) + f(x(t)) = g(t)$. The MSF method uses a second-order Fréchet derivative that is a diagonal matrix. The computational cost of that second-order Fréchet derivative is prohibitive for general systems of nonlinear equations. Interested readers can find information about other multi-step iterative methods for scalar as well as systems of nonlinear equations in [1, 2, 6, 39, 64, 72, 92–97].

6.2 New multi-step iterative method

The new multi-step iterative method (FTUC) can be described as

$$\text{FTUC} = \begin{cases} \text{Number of steps} & = m \geq 3 \\ \text{Convergence-order} & = 3m-4 \\ \text{Function evaluations} & = m-1 \\ \text{Jacobian evaluations} & = 2 \\ \text{LU decomposition} & = 1 \\ \text{Matrix-vector multiplications} & = m-1 \\ \text{Vector-vector multiplications} & = m+1 \\ \text{Number of solutions of systems} & \\ \text{of lower and upper triangular} & \\ \text{systems of equations} & = 2m-2 \end{cases} \begin{cases} \text{Base method} \longrightarrow \begin{cases} \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_1 = \mathbf{F}(\mathbf{x}_k) \\ \mathbf{y}_1 = \mathbf{x}_k - \boldsymbol{\phi}_1 \\ \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_2 = \mathbf{F}(\mathbf{y}_1) \\ \mathbf{y}_2 = \mathbf{y}_1 - 3\,\boldsymbol{\phi}_2 \\ \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_3 = \mathbf{F}'(\mathbf{y}_2)\,\boldsymbol{\phi}_2 \\ \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_4 = \mathbf{F}'(\mathbf{y}_2)\,\boldsymbol{\phi}_3 \\ \mathbf{y}_3 = \mathbf{y}_1 - \frac{7}{4}\,\boldsymbol{\phi}_2 + \frac{1}{2}\,\boldsymbol{\phi}_3 + \frac{1}{4}\,\boldsymbol{\phi}_4 \end{cases} \\ \text{Multi-step part} \rightarrow \begin{cases} \text{for } s = 1, m-3 \\ \quad \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_{2s+3} = \mathbf{F}(\mathbf{y}_{s+2}) \\ \quad \mathbf{F}'(\mathbf{x}_k)\,\boldsymbol{\phi}_{2s+4} = \mathbf{F}'(\mathbf{y}_2)\,\boldsymbol{\phi}_{2s+3} \\ \quad \mathbf{y}_{s+3} = \mathbf{y}_{s+2} - 2\,\boldsymbol{\phi}_{2s+3} + \boldsymbol{\phi}_{2s+4} \\ \text{end} \end{cases} \end{cases} ,$$

The convergence-order of the base method of FTUC is five, and each step of the multi-step part increases the convergence order by three. We will prove the convergence order for $m = 4$ and, then, will use mathematical induction to obtain the convergence order for any m.

6.3 Convergence analysis

In this section, we will prove that the local convergence-order of FTUC is eight for $m = 4$ and later we will establish the proof for the convergence order for arbitrary m via mathematical induction.

Theorem 6.1. *Let $\mathbf{F} : \Gamma \subseteq \mathbb{R}^n \to \mathbb{R}^n$ be sufficiently Frechet differentiable on an open convex neighborhood Γ of $\mathbf{x}^* \in \mathbb{R}^n$ with $\mathbf{F}(\mathbf{x}^*) = 0$ and $det\left(\mathbf{F}'(\mathbf{x}^*)\right) \neq 0$. Then the sequence $\{\mathbf{x}_k\}$ generated by FTUC converges to $\mathbf{x}^*$ with local order of convergence at least eight and the following error equation*

$$\mathbf{e}_{k+1} = \mathbf{L}\mathbf{e}_k{}^8 + O\left(\mathbf{e}_k{}^9\right), \tag{6.1}$$

where $\mathbf{e}_k = \boldsymbol{x}_k - \mathbf{x}^*$, $\mathbf{e}_k{}^p = \overbrace{(\mathbf{e}_k, \mathbf{e}_k, \cdots, \mathbf{e}_k)}^{p\ times}$ *and* $\mathbf{L} = -15\,\mathbf{A}_3\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2 + 45\,\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^2 + 120\,\mathbf{A}_3\mathbf{A}_2^5 + 5\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 - 15\,\mathbf{A}_2\mathbf{A}_3^2\mathbf{A}_2^2 + 150\,\mathbf{A}_2^3\mathbf{A}_3\mathbf{A}_2^2 - 40\,\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^4 - 50\,\mathbf{A}_2^4\mathbf{A}_3\mathbf{A}_2 + 400\,\mathbf{A}_2^7$ *is a p-linear function i.e.* $\mathbf{L} \in \mathbb{L}\overbrace{(\mathbb{R}^n, \mathbb{R}^n, \cdots, \mathbb{R}^n)}^{p\ times}$ *and* $\mathbf{L}\mathbf{e}_k{}^p \in \mathbb{R}^n$.

Proof. Let $\mathbf{F} : \Gamma \subseteq \mathbb{R}^n \to \mathbb{R}^n$ be sufficiently Frechet differentiable function in Γ. The qth Frechet derivative of $\mathbf{F}$ at $v \in \mathbb{R}^n$, $q \geq 1$, is the $q - linear$ function $\mathbf{F}^{(q)}(v) : \overbrace{\mathbb{R}^n\mathbb{R}^n \cdots \mathbb{R}^n}^{\text{q times}}$ such that $\mathbf{F}^{(q)}(v)(u_1, u_2, \cdots, u_q) \in \mathbb{R}^n$. The Taylor's series expansion of $\mathbf{F}(\mathbf{x}_k)$ around $\mathbf{x}^*$ can be written as

$$\begin{aligned}
\mathbf{F}(\mathbf{x}_k) &= \mathbf{F}(\mathbf{x}^* + \mathbf{x}_k - \mathbf{x}^*) = \mathbf{F}(\mathbf{x}^* + \mathbf{e}_k), \\
&= \mathbf{F}(\mathbf{x}^*) + \mathbf{F}'(\mathbf{x}^*)\,\mathbf{e}_k + \frac{1}{2!}\mathbf{F}''(\mathbf{x}^*)\,\mathbf{e}_k{}^2 + \frac{1}{3!}\mathbf{F}^{(3)}(\mathbf{x}^*)\,\mathbf{e}_k{}^3 + \cdots, \\
&= \mathbf{F}'(\mathbf{x}^*)\left(\mathbf{e}_k + \frac{1}{2!}\mathbf{F}'(\mathbf{x}^*)^{-1}\mathbf{F}''(\mathbf{x}^*)\,\mathbf{e}_k{}^2 + \frac{1}{3!}\mathbf{F}'(\mathbf{x}^*)^{-1}\mathbf{F}^{(3)}(\mathbf{x}^*)\,\mathbf{e}_k{}^3 + \cdots\right), \\
&= \mathbf{A}_1\left(\mathbf{e}_k + \mathbf{A}_2\,\mathbf{e}_k{}^2 + \mathbf{A}_3\,\mathbf{e}_k{}^3 + O\left(\mathbf{e}_k{}^4\right)\right), \qquad (6.2)
\end{aligned}$$

where $\mathbf{A}_1 = \mathbf{F}'(\mathbf{x}^*)$ and $\mathbf{A}_s = \frac{1}{s!}\mathbf{F}'(\mathbf{x}^*)^{-1}\mathbf{F}^{(s)}(\mathbf{x}^*)$ for $s \geq 2$. From (6.2), we can calculate the Fréchet derivative of $\mathbf{F}$ as

$$\mathbf{F}'\left(\mathbf{x}_k\right) = \mathbf{A}_1 \left(\mathbf{I} + 2\mathbf{A}_2\, \mathbf{e}_k + 3\mathbf{A}_3\, {\mathbf{e}_k}^2 + 4\mathbf{A}_3\, {\mathbf{e}_k}^3 + O\left({\mathbf{e}_k}^4\right)\right), \tag{6.3}$$

where $\mathbf{I}$ is the identity matrix. Furthermore, using the Maple software package, we obtained the following expression for the inverse of the Fréchet derivative:

$$\begin{aligned}
\mathbf{F}'\Big(\mathbf{x}_k\Big)^{-1} = \bigg(&\mathbf{I} - 2\mathbf{A}_2\, \mathbf{e}_k + \Big(4\mathbf{A}_2^2 - 3\mathbf{A}_3\Big)\mathbf{e}_k^2 + \Big(6\mathbf{A}_3\mathbf{A}_2 + 6\mathbf{A}_2\mathbf{A}_3 - \\
&8\mathbf{A}_2^3 - 4\mathbf{A}_4\Big)\mathbf{e}_k^3 + \Big(8\mathbf{A}_4\mathbf{A}_2 + 9\mathbf{A}_3^2 + 8\mathbf{A}_2\mathbf{A}_4 - 5\mathbf{A}_5 - 12\mathbf{A}_3\mathbf{A}_2^2 - \\
&12\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 - 12\mathbf{A}_2^2\mathbf{A}_3 + 16\mathbf{A}_2^4\Big)\, \mathbf{e}_k^4 + \Big(24\mathbf{A}_3\mathbf{A}_2^3 + 24\mathbf{A}_2^3\mathbf{A}_3 + \\
&24\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2 + 24\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^2 + 10\mathbf{A}_5\mathbf{A}_2 + 12\mathbf{A}_4\mathbf{A}_3 + 12\mathbf{A}_3\mathbf{A}_4 + \\
&10\mathbf{A}_2\mathbf{A}_5 - 6\mathbf{A}_6 - 16\mathbf{A}_4\mathbf{A}_2^2 - 18\mathbf{A}_3^2\mathbf{A}_2 - 18\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3 - \\
&16\mathbf{A}_2\mathbf{A}_4\mathbf{A}_2 - 18\mathbf{A}_2\mathbf{A}_3^2 - 16\mathbf{A}_2^2\mathbf{A}_4 - 32\mathbf{A}_2^5\Big)\, \mathbf{e}_k^5 + \Big(32\mathbf{A}_4\mathbf{A}_2^3 + \\
&64\mathbf{A}_2^6 - 48\mathbf{A}_3\mathbf{A}_2^4 + 12\mathbf{A}_2\mathbf{A}_6 + 16\mathbf{A}_4^2 + 15\mathbf{A}_3\mathbf{A}_5 + 15\mathbf{A}_5\mathbf{A}_3 + \\
&12\mathbf{A}_6\mathbf{A}_2 - 24\mathbf{A}_4\mathbf{A}_2\mathbf{A}_3 - 24\mathbf{A}_4\mathbf{A}_3\mathbf{A}_2 - 20\mathbf{A}_2^2\mathbf{A}_5 - 24\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4 - \\
&24\mathbf{A}_2\mathbf{A}_4\mathbf{A}_3 + 32\mathbf{A}_2^3\mathbf{A}_4 - 20\mathbf{A}_2\mathbf{A}_5\mathbf{A}_2 + 36\mathbf{A}_2^2\mathbf{A}_3^2 - 20\mathbf{A}_5\mathbf{A}_2^2 + \\
&32\mathbf{A}_2^2\mathbf{A}_4\mathbf{A}_2 + 32\mathbf{A}_2\mathbf{A}_4\mathbf{A}_2^2 + 36\mathbf{A}_2\mathbf{A}_3^2\mathbf{A}_2 + 36\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3 + \\
&36\mathbf{A}_3^2\mathbf{A}_2^2 - 7\mathbf{A}_7 - 24\mathbf{A}_3\mathbf{A}_2\mathbf{A}_4 - 27\mathbf{A}_3^3 - 24\mathbf{A}_3\mathbf{A}_4\mathbf{A}_2 + \\
&36\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 + 36\mathbf{A}_3\mathbf{A}_2^2\mathbf{A}_3 - 48\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2^2 - 48\mathbf{A}_2^3\mathbf{A}_3\mathbf{A}_2 - \\
&48\mathbf{A}_2^4\mathbf{A}_3 - 48\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^3\Big)\, \mathbf{e}_k^6 + O\Big(\mathbf{e}_k^7\Big)\bigg)\, \mathbf{A}_1^{-1}.
\end{aligned} \tag{6.4}$$

Substituting (6.2) and (6.4) in $\boldsymbol{\phi}_1 = \mathbf{F}'(\mathbf{x}_k)^{-1}\mathbf{F}(\mathbf{x}_k)$, we get

$$\begin{aligned}
\boldsymbol{\phi}_1 = \mathbf{e}_k - \mathbf{A}_2\, \mathbf{e}_k^2 + \Big(2\mathbf{A}_2^2 - 2\mathbf{A}_3\Big)\, \mathbf{e}_k^3 + \Big(&-3\mathbf{A}_4 - 4\mathbf{A}_2^3 + 3\mathbf{A}_3\mathbf{A}_2 + \\
4\mathbf{A}_2\mathbf{A}_3\Big)\, \mathbf{e}_k^4 + \Big(&-4\mathbf{A}_5 - 6\mathbf{A}_3\mathbf{A}_2^2 - 6\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 - 8\mathbf{A}_2^2\mathbf{A}_3 + 8\mathbf{A}_2^4 + 4\mathbf{A}_4\mathbf{A}_2 \\
&+ 6\mathbf{A}_3^2 + 6\mathbf{A}_2\mathbf{A}_4\Big)\, \mathbf{e}_k^5 + \Big(-5\mathbf{A}_6 + 12\mathbf{A}_3\mathbf{A}_2^3 + 16\mathbf{A}_2^3\mathbf{A}_3 + 12\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2 + \\
&12\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^2 - 8\mathbf{A}_4\mathbf{A}_2^2 - 9\mathbf{A}_3^2\mathbf{A}_2 - 12\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3 - 8\mathbf{A}_2\mathbf{A}_4\mathbf{A}_2 - 12\mathbf{A}_2\mathbf{A}_3^2 - \\
&12\mathbf{A}_2^2\mathbf{A}_4 - 16\mathbf{A}_2^5 + 5\mathbf{A}_5\mathbf{A}_2 + 8\mathbf{A}_4\mathbf{A}_3 + 9\mathbf{A}_3\mathbf{A}_4 + 8\mathbf{A}_2\mathbf{A}_5\Big)\, \mathbf{e}_k^6 + O\Big(\mathbf{e}_k^7\Big).
\end{aligned} \tag{6.5}$$

Using $\mathbf{y}_1 = \mathbf{x} - \boldsymbol{\phi}_1$, we obtain

$$\mathbf{y}_1 - \mathbf{x}^* = \mathbf{x} - \mathbf{x}^* - \boldsymbol{\phi}_1 = \mathbf{e}_k - \boldsymbol{\phi}_1\,,$$

and substituting (6.5), we get

$$\begin{aligned}\mathbf{y}_1-\mathbf{x}^* = {}& \mathbf{A}_2\,\mathbf{e}_k^2+\Big(-2\mathbf{A}_2^2+2\mathbf{A}_3\Big)\,\mathbf{e}_k^3+\Big(-3\mathbf{A}_3\mathbf{A}_2-4\mathbf{A}_2\mathbf{A}_3+\\ & 4\mathbf{A}_2^3+3\mathbf{A}_4\Big)\,\mathbf{e}_k^4+\Big(-4\mathbf{A}_4\mathbf{A}_2-6\mathbf{A}_3^2-6\mathbf{A}_2\mathbf{A}_4+6\mathbf{A}_3\mathbf{A}_2^2+\\ & 8\mathbf{A}_2^2\mathbf{A}_3+6\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2-8\mathbf{A}_2^4+4\mathbf{A}_5\Big)\,\mathbf{e}_k^5+\Big(16\mathbf{A}_2^5-5\mathbf{A}_5\mathbf{A}_2-\\ & 8\mathbf{A}_4\mathbf{A}_3-9\mathbf{A}_3\mathbf{A}_4-8\mathbf{A}_2\mathbf{A}_5+8\mathbf{A}_4\mathbf{A}_2^2+12\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3+9\mathbf{A}_3^2\mathbf{A}_2+\\ & 12\mathbf{A}_2\mathbf{A}_3^2+12\mathbf{A}_2^2\mathbf{A}_4+8\mathbf{A}_2\mathbf{A}_4\mathbf{A}_2-12\mathbf{A}_3\mathbf{A}_2^3-12\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2-\\ & 16\mathbf{A}_2^3\mathbf{A}_3-12\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^2+5\mathbf{A}_6\Big)\,\mathbf{e}_k^6+O\left(\mathbf{e}_k^7\right).\end{aligned}\tag{6.6}$$

Substituting (6.4) and

$$\mathbf{F}(\mathbf{y}_1)=\mathbf{A}_1\left(\mathbf{y}_1+\mathbf{A}_2\,\mathbf{y}_1{}^2+\mathbf{A}_3\,\mathbf{y}_1{}^3+\cdots\right)$$

in $\boldsymbol{\phi}_2=\mathbf{F}'(\mathbf{x}_k)^{-1}\mathbf{F}(\mathbf{y}_1)$, we get

$$\begin{aligned}\boldsymbol{\phi}_2 = {}& \mathbf{A}_2\mathbf{e}_k^2+\Bigg(-4\mathbf{A}_2^2+2\mathbf{A}_3\Bigg)\mathbf{e}_k^3+\Bigg(3\mathbf{A}_4-8\mathbf{A}_2\mathbf{A}_3-6\mathbf{A}_3\mathbf{A}_2+13\mathbf{A}_2^3\Bigg)\mathbf{e}_k^4\\ & +\Bigg(4\mathbf{A}_5-38\mathbf{A}_2^4+20\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2+26\mathbf{A}_2^2\mathbf{A}_3+18\mathbf{A}_3\mathbf{A}_2^2-12\mathbf{A}_2\mathbf{A}_4-12\mathbf{A}_3^2\end{aligned}\tag{6.7}$$

$$\begin{aligned} & -8\mathbf{A}_4\mathbf{A}_2\Bigg)\,\mathbf{e}_k^5+\Bigg(5\mathbf{A}_6+104\mathbf{A}_2^5+27\mathbf{A}_2\mathbf{A}_4\mathbf{A}_2+40\mathbf{A}_2\mathbf{A}_3^2+39\mathbf{A}_2^2\mathbf{A}_4\\ & +27\mathbf{A}_3^2\mathbf{A}_2+36\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3-50\mathbf{A}_3\mathbf{A}_2^3-16\mathbf{A}_2\mathbf{A}_5-18\mathbf{A}_3\mathbf{A}_4-16\mathbf{A}_4\mathbf{A}_3\\ & -10\mathbf{A}_5\mathbf{A}_2+24\mathbf{A}_4\mathbf{A}_2^2-55\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^2-76\mathbf{A}_2^3\mathbf{A}_3-59\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2\Bigg)\,\mathbf{e}_k^6\\ & +O\Bigg(\mathbf{e}_k^7\Bigg).\end{aligned}$$

Using $\mathbf{y}_2=\mathbf{y}_1-3\boldsymbol{\phi}_2$ and substituting (6.6) and (6.7), we get

$$\begin{aligned}\mathbf{y}_2-\mathbf{x}^* = {}& -2\mathbf{A}_2\,\mathbf{e}_k^2+\Big(10\mathbf{A}_2^2-4\mathbf{A}_3\Big)\,\mathbf{e}_k^3+\Big(-6\mathbf{A}_4+15\mathbf{A}_3\mathbf{A}_2+20\mathbf{A}_2\mathbf{A}_3\\ & -35\mathbf{A}_2^3\Big)\,\mathbf{e}_k^4+\Big(-8\mathbf{A}_5+106\mathbf{A}_2^4+20\mathbf{A}_4\mathbf{A}_2+30\mathbf{A}_3^2+30\mathbf{A}_2\mathbf{A}_4\\ & -48\mathbf{A}_3\mathbf{A}_2^2-70\mathbf{A}_2^2\mathbf{A}_3-54\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2\Big)\,\mathbf{e}_k^5+\Big(-296\mathbf{A}_2^5\\ & -73\mathbf{A}_2\mathbf{A}_4\mathbf{A}_2-108\mathbf{A}_2\mathbf{A}_3^2-105\mathbf{A}_2^2\mathbf{A}_4-72\mathbf{A}_3^2\mathbf{A}_2-96\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3\\ & +138\mathbf{A}_3\mathbf{A}_2^3+40\mathbf{A}_2\mathbf{A}_5+45\mathbf{A}_3\mathbf{A}_4+40\mathbf{A}_4\mathbf{A}_3+25\mathbf{A}_5\mathbf{A}_2-64\mathbf{A}_4\mathbf{A}_2^2\\ & +153\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^2+212\mathbf{A}_2^3\mathbf{A}_3+165\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2-10\mathbf{A}_6\Big)\,\mathbf{e}_k^6+O\Big(\mathbf{e}_k^7\Big).\end{aligned}\tag{6.8}$$

Substituting (6.4), (6.7) and (6.8) in $\boldsymbol{\phi}_3 = \mathbf{F}^{'}(\mathbf{x}_k)^{-1}\mathbf{F}(\mathbf{y}_2)\,\boldsymbol{\phi}_2$, we get

$$
\begin{aligned}
\boldsymbol{\phi}_3 = {} & \mathbf{A}_2\,\mathbf{e}_k^2 + \Big(-6\mathbf{A}_2^2 + 2\mathbf{A}_3\Big)\,\mathbf{e}_k^3 + \Big(3\mathbf{A}_4 - 9\mathbf{A}_3\mathbf{A}_2 - 12\mathbf{A}_2\mathbf{A}_3 \\
& + 21\mathbf{A}_2^3\Big)\,\mathbf{e}_k^4 + \Big(4\mathbf{A}_5 - 44\mathbf{A}_2^4 + 36\mathbf{A}_3\mathbf{A}_2^2 + 42\mathbf{A}_2^2\mathbf{A}_3 + 30\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 \\
& - 12\mathbf{A}_4\mathbf{A}_2 - 18\mathbf{A}_3^2 - 18\mathbf{A}_2\mathbf{A}_4\Big)\,\mathbf{e}_k^5 + \Big(5\mathbf{A}_6 - 10\mathbf{A}_2^5 - 101\mathbf{A}_3\mathbf{A}_2^3 \\
& - 55\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2 - 88\mathbf{A}_2^3\mathbf{A}_3 - 65\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^2 + 48\mathbf{A}_4\mathbf{A}_2^2 + 72\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3 \\
& + 54\mathbf{A}_3^2\mathbf{A}_2 + 63\mathbf{A}_2^2\mathbf{A}_4 + 60\mathbf{A}_2\mathbf{A}_3^2 + 39\mathbf{A}_2\mathbf{A}_4\mathbf{A}_2 - 15\mathbf{A}_5\mathbf{A}_2 - 24\mathbf{A}_4\mathbf{A}_3 \\
& - 27\mathbf{A}_3\mathbf{A}_4 - 24\mathbf{A}_2\mathbf{A}_5\Big)\,\mathbf{e}_k^6 + O\Big(\mathbf{e}_k^7\Big).
\end{aligned}
\tag{6.9}
$$

Using $\boldsymbol{\phi}_4 = \mathbf{F}^{'}(\mathbf{x}_k)^{-1}\mathbf{F}(\mathbf{y}_2)\,\boldsymbol{\phi}_3$ and substituting (6.4), (6.8) and (6.9) , we get

$$
\begin{aligned}
\boldsymbol{\phi}_4 = {} & \mathbf{A}_2\mathbf{e}_k^2 + \Big(-8\mathbf{A}_2^2 + 2\mathbf{A}_3\Big)\mathbf{e}_k^3 + \Big(-12\mathbf{A}_3\mathbf{A}_2 - 16\mathbf{A}_2\mathbf{A}_3 + 33\mathbf{A}_2^3 \\
& + 3\mathbf{A}_4\Big)\,\mathbf{e}_k^4 + \Big(60\mathbf{A}_3\mathbf{A}_2^2 + 66\mathbf{A}_2^2\mathbf{A}_3 + 46\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 - 16\mathbf{A}_4\mathbf{A}_2 - 24\mathbf{A}_3^2 \\
& - 24\mathbf{A}_2\mathbf{A}_4 - 66\mathbf{A}_2^4 + 4\mathbf{A}_5\Big)\,\mathbf{e}_k^5 + \Big(80\mathbf{A}_4\mathbf{A}_2^2 + 120\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3 + 90\mathbf{A}_3^2\mathbf{A}_2 \\
& + 99\mathbf{A}_2^2\mathbf{A}_4 + 92\mathbf{A}_2\mathbf{A}_3^2 + 59\mathbf{A}_2\mathbf{A}_4\mathbf{A}_2 - 20\mathbf{A}_5\mathbf{A}_2 - 32\mathbf{A}_4\mathbf{A}_3 - 36\mathbf{A}_3\mathbf{A}_4 \\
& - 32\mathbf{A}_2\mathbf{A}_5 - 152\mathbf{A}_2^5 - 188\mathbf{A}_3\mathbf{A}_2^3 - 71\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2 - 132\mathbf{A}_2^3\mathbf{A}_3 \\
& - 107\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^2 + 5\mathbf{A}_6\Big)\,\mathbf{e}_k^6 + O\Big(\mathbf{e}_k^7\Big).
\end{aligned}
\tag{6.10}
$$

Using $\mathbf{y}_3 = \mathbf{y}_1 - (7/4)\,\boldsymbol{\phi}_2 + (1/2)\,\boldsymbol{\phi}_3 + (1/4)\,\boldsymbol{\phi}_4$ and substituting (6.6), (6.7), (6.9), and (6.10), we get

$$
\begin{aligned}
\mathbf{y}_3 - \mathbf{x}^* = {} & \Big(20\mathbf{A}_2^4 - (5/2)\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 + (15/2)\mathbf{A}_3\mathbf{A}_2^2\Big)\,\mathbf{e}_k^5 + \Big(-209\mathbf{A}_2^5 \\
& - 5\mathbf{A}_2\mathbf{A}_4\mathbf{A}_2 - 5\mathbf{A}_2\mathbf{A}_3^2 + (45/4)\mathbf{A}_3^2\mathbf{A}_2 + 15\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3 - 22\mathbf{A}_3\mathbf{A}_2^3 \\
& + 10\mathbf{A}_4\mathbf{A}_2^2 + 25\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^2 + 40\mathbf{A}_2^3\mathbf{A}_3 + 46\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2\Big)\,\mathbf{e}_k^6 \\
& + \Big(-(267/2)\mathbf{A}_3\mathbf{A}_2^4 - (15/2)\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4 - 10\mathbf{A}_2\mathbf{A}_4\mathbf{A}_3 \\
& - (15/2)\mathbf{A}_2\mathbf{A}_5\mathbf{A}_2 + 15\mathbf{A}_3\mathbf{A}_4\mathbf{A}_2 + (45/2)\mathbf{A}_3^3 + (45/2)\mathbf{A}_3\mathbf{A}_2\mathbf{A}_4 \\
& + 15\mathbf{A}_4\mathbf{A}_3\mathbf{A}_2 + 20\mathbf{A}_4\mathbf{A}_2\mathbf{A}_3 - 44\mathbf{A}_4\mathbf{A}_2^3 + 72\mathbf{A}_2^2\mathbf{A}_4\mathbf{A}_2 + 92\mathbf{A}_2^2\mathbf{A}_3^2 \\
& + 60\mathbf{A}_2^3\mathbf{A}_4 + 54\mathbf{A}_2\mathbf{A}_3^2\mathbf{A}_2 + 50\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3 + 40\mathbf{A}_2\mathbf{A}_4\mathbf{A}_2^2 \\
& - 50\mathbf{A}_3^2\mathbf{A}_2^2 - 44\mathbf{A}_3\mathbf{A}_2^2\mathbf{A}_3 - 20\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 - (451/2)\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^3 \\
& - 357\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2^2 - 418\mathbf{A}_2^4\mathbf{A}_3 - 385\mathbf{A}_2^3\mathbf{A}_3\mathbf{A}_2 + 25/2\mathbf{A}_5\mathbf{A}_2^2 \\
& + 1316\mathbf{A}_2^6\Big)\,\mathbf{e}_k^7 + O\Big({\mathbf{e}_k}^8\Big).
\end{aligned}
\tag{6.11}
$$

Substituting (6.4) and

$$\mathbf{F}(\mathbf{y}_3) = \mathbf{A}_1 \left(\mathbf{y}_3 + \mathbf{A}_2\, {\mathbf{y}_3}^2 + \mathbf{A}_3\, {\mathbf{y}_3}^3 + \cdots\right),$$

in $\boldsymbol{\phi}_5 = \mathbf{F}'(\mathbf{x}_k)^{-1}\mathbf{F}(\mathbf{y}_3)$, we get

$$\begin{aligned}\boldsymbol{\phi}_5 = &\Big(20\mathbf{A}_2^4 - (5/2)\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 + (15/2)\mathbf{A}_3\mathbf{A}_2^2\Big)\,\mathbf{e}_k^5 + \Big(-22\mathbf{A}_3\mathbf{A}_2^3 \\ &+ 51\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2 + 40\mathbf{A}_2^3\mathbf{A}_3 + 10\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^2 - 249\mathbf{A}_2^5 + 10\mathbf{A}_4\mathbf{A}_2^2 \\ &+ 15\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3 + (45/4)\mathbf{A}_3^2\mathbf{A}_2 - 5\mathbf{A}_2\mathbf{A}_3^2 - 5\mathbf{A}_2\mathbf{A}_4\mathbf{A}_2\Big)\,\mathbf{e}_k^6 \\ &+ \Big(-(387/2)\mathbf{A}_3\mathbf{A}_2^4 - (15/2)\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4 - 10\mathbf{A}_2\mathbf{A}_4\mathbf{A}_3 \\ &- (15/2)\mathbf{A}_2\mathbf{A}_5\mathbf{A}_2 + 15\mathbf{A}_3\mathbf{A}_4\mathbf{A}_2 + (45/2)\mathbf{A}_3^3 + (45/2)\mathbf{A}_3\mathbf{A}_2\mathbf{A}_4 \\ &+ 15\mathbf{A}_4\mathbf{A}_3\mathbf{A}_2 + 20\mathbf{A}_4\mathbf{A}_2\mathbf{A}_3 - 44\mathbf{A}_4\mathbf{A}_2^3 + 82\mathbf{A}_2^2\mathbf{A}_4\mathbf{A}_2 + 102\mathbf{A}_2^2\mathbf{A}_3^2 \\ &+ 60\mathbf{A}_2^3\mathbf{A}_4 + (63/2)\mathbf{A}_2\mathbf{A}_3^2\mathbf{A}_2 + 20\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3 + 20\mathbf{A}_2\mathbf{A}_4\mathbf{A}_2^2 \\ &- (145/2)\mathbf{A}_3^2\mathbf{A}_2^2 - 44\mathbf{A}_3\mathbf{A}_2^2\mathbf{A}_3 - (25/2)\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 - (363/2)\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^3 \\ &- 377\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2^2 - 498\mathbf{A}_2^4\mathbf{A}_3 - 487\mathbf{A}_2^3\mathbf{A}_3\mathbf{A}_2 + 25/2\mathbf{A}_5\mathbf{A}_2^2 \\ &+ 1814\mathbf{A}_2^6\Big)\,\mathbf{e}_k^7 + O\Big({\mathbf{e}_k}^8\Big).\end{aligned} \tag{6.12}$$

Substituting (6.4), (6.8) and (6.12) in $\boldsymbol{\phi}_6 = \mathbf{F}'(\mathbf{x}_k)^{-1}\mathbf{F}'(\mathbf{y}_2)\,\boldsymbol{\phi}_5$, we get

$$\begin{aligned}\boldsymbol{\phi}_6 = &\Big(20\mathbf{A}_2^4 - (5/2)\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 + (15/2)\mathbf{A}_3\mathbf{A}_2^2\Big)\,\mathbf{e}_k^5 + \Big(-289\mathbf{A}_2^5 \\ &+ 10\mathbf{A}_4\mathbf{A}_2^2 + 15\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3 + (45/4)\mathbf{A}_3^2\mathbf{A}_2 - 5\mathbf{A}_2\mathbf{A}_3^2 - 5\mathbf{A}_2\mathbf{A}_4\mathbf{A}_2 \\ &- 22\mathbf{A}_3\mathbf{A}_2^3 + 56\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2 + 40\mathbf{A}_2^3\mathbf{A}_3 - 5\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^2\Big)\,\mathbf{e}_k^6 \\ &+ \Big(2312\mathbf{A}_2^6 - (507/2)\mathbf{A}_3\mathbf{A}_2^4 - (275/2)\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^3 - 397\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2^2 \\ &- 578\mathbf{A}_2^4\mathbf{A}_3 - 589\mathbf{A}_2^3\mathbf{A}_3\mathbf{A}_2 + (25/2)\mathbf{A}_5\mathbf{A}_2^2 + 20\mathbf{A}_4\mathbf{A}_2\mathbf{A}_3 \\ &+ 15\mathbf{A}_4\mathbf{A}_3\mathbf{A}_2 + (45/2)\mathbf{A}_3\mathbf{A}_2\mathbf{A}_4 + (45/2)\mathbf{A}_3^3 + 15\mathbf{A}_3\mathbf{A}_4\mathbf{A}_2 \\ &- (15/2)\mathbf{A}_2\mathbf{A}_5\mathbf{A}_2 - 10\mathbf{A}_2\mathbf{A}_4\mathbf{A}_3 - (15/2)\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4 - 44\mathbf{A}_4\mathbf{A}_2^3 \\ &- 44\mathbf{A}_3\mathbf{A}_2^2\mathbf{A}_3 - 95\mathbf{A}_3^2\mathbf{A}_2^2 + 92\mathbf{A}_2^2\mathbf{A}_4\mathbf{A}_2 + 112\mathbf{A}_2^2\mathbf{A}_3^2 + 60\mathbf{A}_2^3\mathbf{A}_4 \\ &+ 9\mathbf{A}_2\mathbf{A}_3^2\mathbf{A}_2 - 5\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 - 10\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3\Big)\,\mathbf{e}_k^7 + O\Big({\mathbf{e}_k}^8\Big).\end{aligned} \tag{6.13}$$

Finally, using $\mathbf{y}_4 - \mathbf{x}^* = \mathbf{y}_3 - \mathbf{x}^* - 2\boldsymbol{\phi}_5 + 2\boldsymbol{\phi}_6$ and substituting (6.11), (6.12) and (6.13), we get

$$\begin{aligned}\mathbf{y}_4 - \mathbf{x}^* = \Big(&-15\mathbf{A}_3\mathbf{A}_2^2\mathbf{A}_3\mathbf{A}_2 + 45\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^2 + 120\mathbf{A}_3\mathbf{A}_2^5 \\ &+ 5\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2 - 15\mathbf{A}_2\mathbf{A}_3^2\mathbf{A}_2^2 + 150\mathbf{A}_2^3\mathbf{A}_3\mathbf{A}_2^2 - 40\mathbf{A}_2\mathbf{A}_3\mathbf{A}_2^4 \\ &- 50\mathbf{A}_2^4\mathbf{A}_3\mathbf{A}_2 + 400\mathbf{A}_2^7\Big)\,\mathbf{e}_k^8 + O\Big(\mathbf{e}_k{}^9\Big).\end{aligned} \tag{6.14}$$

□

Theorem 6.2. *The multi-step iterative method FTUC has local convergence order at least* $3m-4$ *for* $m \geq 3$.

Proof. The proof is established by mathematical induction. For $m = 3, 4$ the convergence orders are according to (6.11) and (6.14) five and eight, respectively. Consequently, our claim concerning the convergence-order $3m-4$ is true for $m = 3, 4$.

Assume our claim true for $m = q \geq 4$, i.e., that the convergence order of FTUC for $m = q$ is $3q-4$. The qth and $(q-1)\,th$ steps FTUC can be written as

$$\begin{aligned}&\mathbf{F}'(\mathbf{x}_k)\,\mathbf{T} = \mathbf{F}'(\mathbf{y}_2)\\ &\text{Frozen factor} = (2\mathbf{I} - \mathbf{T})\,\mathbf{F}'(\mathbf{x}_k)^{-1},\\ &\mathbf{y}_{q-1} = \mathbf{y}_{q-2} - (\text{Frozen factor})\,\mathbf{F}(\mathbf{y}_{q-2}),\\ &\mathbf{y}_q = \mathbf{y}_{q-1} - (\text{Frozen factor})\,\mathbf{F}(\mathbf{y}_{q-1}).\end{aligned}$$

The enhancement in the convergence order of FTUC from the $(q-1)\,th$ step to the qth step is $(3q-4) - (3(q-1)-4) = 3$. Now, we write the $(q+1)\,th$ step of FTUC as

$$\mathbf{y}_{q+1} = \mathbf{y}_q - (\text{Frozen factor})\,\mathbf{F}(\mathbf{y}_q).$$

The increment in the convergence-order of FTUC, due to $(q+1)\,th$-step, is precisely three because the use of the Frozen factor adds an additive constant to the convergence order [65]. Therefore, the convergence order after the addition of the $(q+1)\,th$ step is $3q - 4 + 3 = 3q - 1 = 3(q+1) - 4$, completing the induction step. □

6.4 Efficiency index

In the literature, the computational efficiency of an iterative method is evaluated by the efficiency index[99] defined as

$$E = p^{1/C}, \tag{6.15}$$

where p is the order of convergence of the method and C is the computational cost of the method defined as

$$C\left(\mu_0,\ \mu_1,\ n\right) = P_0\left(n\right)\mu_0 + P_1\left(n\right)\mu_1 + P\left(n\right), \tag{6.16}$$

where $P_0\left(n\right)$ is the number of scalar function f_i evaluations in $\mathbf{F}\left(\cdot\right)$, $P_1\left(n\right)$ is the number of scalar function $\frac{\partial f_i}{\partial x_j}$ evaluations in the Jacobian, $P\left(n\right)$ is the number of products/divisions, and μ_i are coefficients which are required to express the computational cost in terms of products/divisions. Table 6.1 gives the number of products, divisions and computational cost of some operations in terms on the number of variables n. In the table and in the sequel, l denotes the cost of a division relative to the cost of a multiplication.

	Multiplications	Divisions	Computational cost
LU decomposition	$\frac{n(n-1)(2n-1)}{6}$	$\frac{n(n-1)}{2}$	$\frac{n(n-1)(2n-1)}{6} + l\,\frac{n(n-1)}{2}$
Solution of two triangular systems	$n\left(n-1\right)$	n	$n\left(n-1\right) + ln$
Matrix vector multiplication	n^2		n^2
Vector vector multiplication	n		n

TABLE 6.1: Computational cost of different operations.

The computational costs of HJ and FTUC, C_{HJ} and C_{FTUC}, are

$$\begin{aligned} C_{HJ} &= \left(m-1\right)\mu_0 n + 2n^2\mu_1 + mn^2 + 2mn + \left(2m-1\right)\left(n\left(n-1\right)+ln\right) \\ &+ \frac{n\left(n-1\right)\left(2n-1\right)}{6} + l\,\frac{n\left(n-1\right)}{2}, \end{aligned} \tag{6.17}$$

$$\begin{aligned} C_{FTUC} &= \left(m-1\right)\mu_0 n + 2n^2\mu_1 + \left(m-1\right)n^2 + \left(m+1\right)n \\ &+ \left(2m-2\right)\left(n\left(n-1\right)+ln\right) + \frac{n\left(n-1\right)\left(2n-1\right)}{6} + l\frac{n\left(n-1\right)}{2}. \end{aligned} \tag{6.18}$$

To compare the efficiencies of HJ and FTUC we define the quotient

$$R = \frac{log\left((3m_1-4)^{1/C_{FTUC}}\right)}{log\left((2m_2)^{1/C_{HJ}}\right)} = \frac{C_{HJ}}{C_{FTUC}}\frac{log\,(3m_1-4)}{2m_2}\,, \tag{6.19}$$

where m_1 is the number of steps of FTUC and m_2 is the number of steps of HJ. Clearly, if $R > 1$ the efficiency of FTUC is higher than that of HJ.

6.4.1 Comparison of the efficiencies of FTUC and HJ for convergence orders five and four, respectively

When FTUC and HJ have convergence orders five and four, respectively, the value of R is

$$R = \frac{ln\,(5)\,(3ln + 12\mu_1 n + 2n^2 + 15l + 6\mu_0 + 27n + 7)}{2\,(3ln + 12\mu_1 n + 2n^2 + 21l + 12\mu_0 + 33n + 1)\,ln\,(2)}\,. \tag{6.20}$$

R is > 1 if

$$\begin{aligned}\mu_0 <&0.06394801344n^2 + (0.09592202004l + 0.3836880802\mu_1\\ &- 0.3285458591)n - 0.7122339393l + 1.415662087\,.\end{aligned} \tag{6.21}$$

In that case the base method of FTUC will be more efficient than the base method of HJ.

6.4.2 Comparison of the efficiencies of FTUC and HJ for the same convergence order

FTUC and HJ have convergence order $6s-4$ for numbers of steps $m_1 = 2s$ and $m_2 = 3s-2$, $s \geq 2$, respectively. In that case,

$$R = 1 + \frac{6(2ls + \mu_0 s + 3ns - 3l - 2\mu_0 - 4n + 2s - 2)}{3ln + 24ls + 12\mu_0 s + 12\mu_1 n + 2n^2 + 36ns - 15l - 6\mu_0 - 21n - 12s + 19}\,. \tag{6.22}$$

The value of l depends on the computer system where the methods are run. Reasonable values for l fall between 2.5 and 3. Replacing s and l by $s+2$ and

$l+1$ in (6.22) gives

$$R = 1 + \frac{6\,(2ls + \mu_0 s + 3ns + l + 2n + 4s + 3)}{3ln + 24ls + 12\mu_0 s + 12\mu_1 n + 2n^2 + 36ns + 33l + 18\mu_0 + 54n + 12s + 28}\,, \tag{6.23}$$

which is > 1 for $s \geq 0$. Since $l > 1$, this shows that $R > 1$ and FTUC is more efficient than HJ when $s \geq 2$, i.e. with a convergence order ≥ 8.

6.4.3 Comparison of FTUC and HJ for the same number of function evaluations

The values of R when both methods make the same number of function evaluations ($\geq$ 3) are given for several cases in Table 6.2. In the expressions for R, l is replaced by $l+1$. We can see that, for the same number of function evaluations, the convergence order of FTUC is higher than that of HJ for a number of steps larger than 4. Also, since $l > 1$, the values of $R-1$ reveal that FTUC has, for the same number of function evaluations, higher efficiency than HJ. Table 6.3 compares the charectristics of FTUC and HJ. For two function evaluations the convergence orders of FTUC and HJ are five and six, respectively. In that case, the performance of HJ is better than that of.However, for a number of function evaluations greater then two, the performance of FTUC is better than that of HJ.

Number of steps	Function Evaluations	Convergence order of (FTUC, HJ)	R
4	3	(8, 8)	$1+\frac{6(l+2\,n+3)}{12\,n\mu_1+3\,ln+2\,n^2+18\,\mu_0+33\,l+54\,n+28}$
5	4	(11, 10)	$1+\frac{0.16\,\mu_0+0.08\,n\mu_1+2.57\,n+4.40+1.35\,l+0.01\,n^2+0.02\,ln}{4.0\,\mu_0+2.0\,n\mu_1+12.0\,n+5.66+7.50\,l+0.33\,n^2+0.50\,ln}$
6	5	(14, 12)	$1+\frac{0.31\,\mu_0+0.12\,n\mu_1+3.05\,n+5.72+1.65\,l+0.02\,n^2+0.03\,ln}{5.0\,\mu_0+2.0\,n\mu_1+15.0\,n+6.66+9.50\,l+0.33\,n^2+0.50\,ln}$
100	99	(296, 200)	$1+\frac{7.32\,\mu_0+0.14\,n\mu_1+24.12\,n+113.77+15.68\,l+0.02\,n^2+0.03\,ln}{99.0\,\mu_0+2.0\,n\mu_1+297.0\,n+100.66+197.50\,l+0.33\,n^2+0.50\,ln}$

TABLE 6.2: Comparison between computational efficiencies when FTUC and HJ make the same number of function evaluations.

	FTUC $(m \geq 3)$	HJ $(m \geq 2)$
Number of steps	m	m
Convergence order	$3m-4$	$2m$
Function evaluations	$m-1$	$m-1$
Jacobian evaluations	2	2
LU decompositions	1	1
Matrix vector multiplications	$m-1$	m
Vector vector multiplications	$m+1$	$2m$
Number of solutions of systems of linear equations	$2m-2$	$2m-1$

Table 6.3: Comparison between FTUC and HJ when both methods make the same number of function evaluations.

6.4.4 Comparison of FTUC and MSF for the same number of function evaluations

The MSF method uses m_1 function evaluations and one second-order Fréchet derivative. For the purpose of comparison, we assume that the evaluation of a second-order Fréchet derivative has same computational cost as that of a function. The above assumption only makes sense if the second-order Fréchet derivative is a diagonal matrix. Under that assumption, MSF makes m_1+1 function evaluations to achieve a convergence order of $3m_1$ in m_1 steps and FTUC makes m_2-1 function evaluations to achieve a convergence order of $3m_2-4$ in m_2 steps. If we equate the number of function evaluations of FTUC and MSF we get

$$\begin{aligned} m_2 - 1 &= m_1 + 1, \\ m_2 &= m_1 + 2. \end{aligned}$$

Suppose $m_1 = m$ and $m_2 = m+2$. Table 6.4 compares the characteristics of FTUC and MSF for those numbers of steps. MSF achieves a convergence order of $3m$ in m steps by making $m+1$ function evaluation while FTUC achieves a convergence order of $3m+2$ in $m+2$ steps by making $m+1$ function evaluations. MSF makes $m-3$, $m+3$ and $m-3$ more matrix vector multiplications, vector vector multiplications and solutions of upper and lower triangular systems of linear equations than FTUC. Obviously, for the same number of function evaluations, FTUC is computationally more efficient than MSF and achieves a better convergence order. The high computational cost of second-order Fréchet derivatives makes that method impractical for general systems of nonlinear equations. Since, even if the second-order Fréchet derivative is a diagonal matrix, MSF has worse performance than FTUC, we will not test numerically MSF against FTUC.

	FTUC	MSF
Number of steps	$m+2$	m
Convergence order	$3m+2$	$3m$
Function evaluations	$m+1$	$m+1$
Jacobian evaluations	2	2
LU decompositions	1	1
Matrix vector multiplications	$m+1$	$2m-2$
Vector vector multiplications	$m+3$	$m+2$
Number of solutions of lowers and upper triangular systems	$2m+2$	$3m-1$

Table 6.4: Comparison FTUC and MSF for, respectively, $m+2$ and m steps.

6.5 Numerical Results

We will use the definition for the computational convergence order (CCO)

$$\rho_q = \frac{\log\left(||\mathbf{y}_{q+2} - \mathbf{y}^*||_1 / ||\mathbf{y}_{q+1} - \mathbf{y}^*||_1\right)}{\log\left(||\mathbf{y}_{q+1} - \mathbf{y}^*||_1 / ||\mathbf{y}_{q} - \mathbf{y}^*||_1\right)}, \tag{6.24}$$

where $\mathbf{y}^*$ is the zero of function $\mathbf{F}(\cdot)$ and $\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \ldots$ is the sequence of approximations given by the muti-step iterative method. In all tests, we will use the Chebyshev pseudospectral collocation method to approximate derivatives.

6.5.1 General systems of nonlinear equations

As a first example, we will first solve a small system of nonlinear equations. The results will confirm the claimed convergence order and accuracy of our method. The system of nonlinear equations is

$$\begin{aligned}
&x_2x_3 + x_4\left(x_2 + x_3\right) = 0\\
&x_1x_3 + x_4\left(x_1 + x_3\right) = 0\\
&x_1x_2 + x_4\left(x_1 + x_2\right) = 0\\
&x_1x_2 + x_1x_3 + x_2x_3 - 1 = 0\,.
\end{aligned} \tag{6.25}$$

Its solution up to an accuracy of 32 digits is

$$\begin{aligned}
x_1 &= 0.57735026918962576450914878050 2\\
x_2 &= 0.577350269189625764509148780502\\
x_3 &= 0.577350269189625764509148780502\\
x_4 &= -0.288675134594812882254574390251\,.
\end{aligned} \tag{6.26}$$

Table 6.5 gives the results obtained under FTUC and HJ. The accuracy obtained by HJ method is superior to that of FTUC, but the execution time of HJ is longer than that of FTUC. In Table 6.6, we give results obtained by approximately equating the execution times of both methods. Now, the accuracy obtained by FTUC is better, showing the superiority of FTUC over HJ.

Iterative methods		FTUC	HJ
Number of iterations		3	1
Size of problem		4	4
Number of steps		6	7
Theoretical convergence order		14	14
Computational convergence order		14.1	14.1
Number of function evaluations per iteration		5	6
Solutions of system of linear equations per iteration		10	13
Number of matrix vector multiplication per iteration		5	7
	Iteration		
$\|\|\mathbf{y}_q - \mathbf{y}^*\|\|_1$	1	$8.21e-10$	$1.66e-10$
	2	$4.76e-136$	$2.51e-146$
	3	$5.26e-1918$	$1.84e-2062$
Execution time (s)		3.37	3.82

Table 6.5: Results under FTUC and HJ for the first example.

Iterative methods		FTUC	HJ
Number of iterations		3	1
Number of steps		7	7
Theoretical convergence order		17	14
Computational convergence order		17.1	14.1
Number of function evaluations per iteration		6	6
Solutions of system of linear equations per iteration		12	13
Number of matrix vector multiplication per iteration		6	7
	Iteration		
$\|\|\mathbf{y}_q - \mathbf{y}^*\|\|_1$	1	$1.03e-11$	$1.66e-10$
	2	$6.58e-199$	$2.51e-146$
	3	$1.27e-3400$	$1.84e-2062$
Execution time (s)		3.71	3.82

Table 6.6: Comparison of performance between FTUC and HJ when both have approximately same execution time.

6.5.2 Classical Blasius flat-plate problem

The Classical Blasius flat-plate flow problem [100] is the boundary value problem

$$\begin{aligned}
&u'''\left(x\right)+\tfrac{1}{2}u''\left(x\right)u\left(x\right)=0\\
&u\left(0\right)=u'\left(0\right)=0,\ \left(u',\ u''\right)\longrightarrow\left(1,\ 0\right)\ \text{as}\ x\longrightarrow\infty\\
&\mathbf{F}\left(\boldsymbol{u}\right)=D_x^3\boldsymbol{u}+\tfrac{1}{2}\boldsymbol{u}D_x^2\boldsymbol{u}\\
&\mathbf{F}'\left(\boldsymbol{u}\right)=D_x^3+\tfrac{1}{2}\left(\text{diag}\left(D_x^2\boldsymbol{u}\right)+\text{diag}\left(\boldsymbol{u}\right)D_x^2\right).
\end{aligned}\tag{6.27}$$

We considered that problem for a domain for x $[0, 200]$ with a mesh giving a problem size of 250. From a practical point of view, many researchers have been interested in computing $u''\left(0\right)$, and Howarth [100] reported $u''\left(0\right)=0.332057$. We have computed $u''(0)$ with an accuracy of twenty five digits with the result 0.33205733628286351714556307. The FTUC iterative method produced numerical values of u' and u'' at $x=200$ with accuracies of $3.87e-26$ and $5.65e-25$, respectively. The initial guess was

$$u_0\left(x\right)=\begin{cases}x^2 & \text{if } x\leq 1\\ x & \text{othersise}\end{cases}.\tag{6.28}$$

Figure 6.1 shows $u\left(x\right)$, $u'\left(x\right)$ and $u''\left(x\right)$. Figure 6.2 plots of $\log\left(||\mathbf{F}(\mathbf{u})||_1\right)$ versus the number of steps for the first three iterations. FTUC showed clear dominance over HJ. Table 6.7 gives the results obtained under FTUC and HJ with 30 steps.

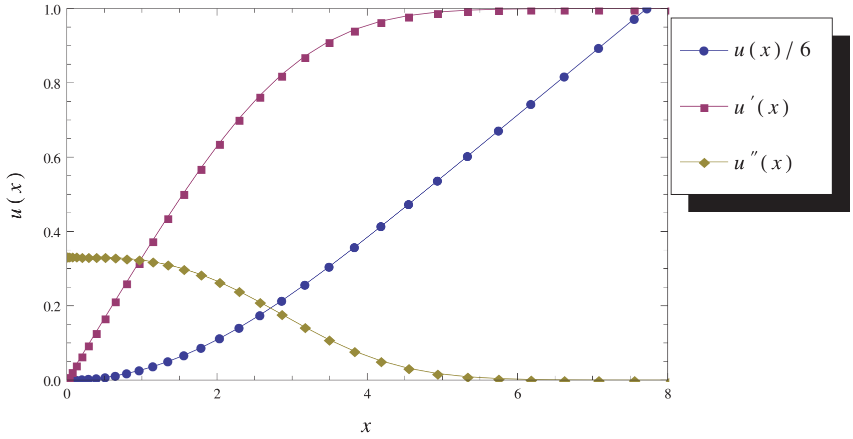


FIGURE 6.1: Numerically calculated profile of $u\left(x\right)$, $u'\left(x\right)$ and $u''\left(x\right)$.

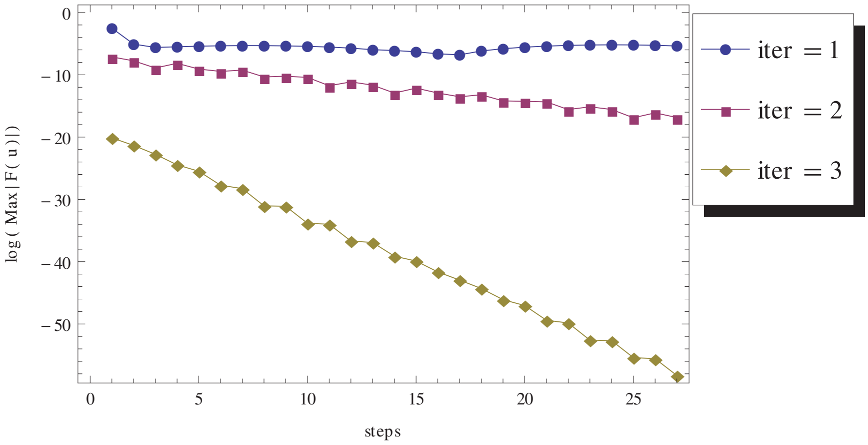


FIGURE 6.2: Plot of $\log\left(||\mathbf{F}(\mathbf{u})||_1\right)$ versus the number steps, for the first three iterations.

Iterative methods		FTUC	HJ
Number of iterations		3	3
Number of steps		30	30
Theoretical convergence order		86	60
Number of function evaluations per iteration		29	29
Solutions of of linear systems per iteration		58	59
Number of matrix vector multiplication per iteration		29	30
	Iteration		
$\log\left(\|\|\mathbf{F}(\mathbf{u}_q)\|\|_1\right)$	1	$4.63e-3$	$2.47e-3$
	2	$4.78e-8$	$8.605e-4$
	3	$4.98e-26$	$1.83e-7$
Execution time (s)		125.42	158.82

TABLE 6.7: Performance of FTUC and HJ for the classical Blasius flat-plate problem for 30 steps.

6.5.3 Klein-Gordon equation

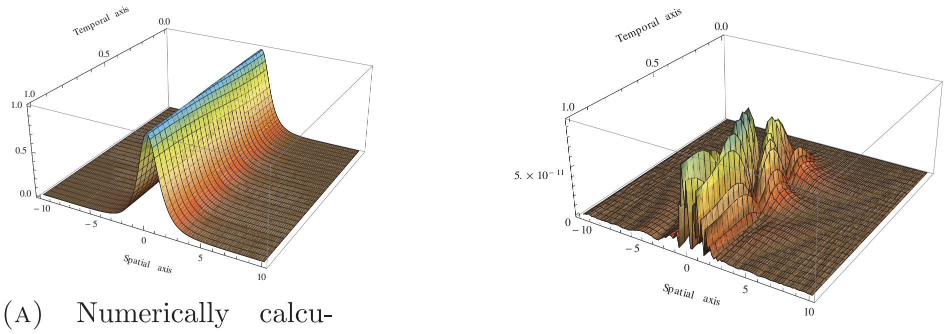
The Klein-Gordon equation[75] is the relativistic case of the Schrödinger equation

$$\begin{aligned} & u_{tt} - c^2 u_{xx} + f\left(u\right) = p - \infty < x < \infty, \ t > 0 \\ & \mathbf{F}\left(\boldsymbol{u}\right) = \left(D_t^2 - c^2 D_x^2\right)\boldsymbol{u} + f\left(\boldsymbol{u}\right) - \boldsymbol{p} \\ & \mathbf{F}' = D_t^2 - c^2 D_x^2 + \operatorname{diag}\left(f'\left(\boldsymbol{u}\right)\right), \end{aligned} \tag{6.29}$$

where $f\left(u\right)$ is an odd function of u and initial conditions are given by

$$\begin{aligned} & u\left(x, 0\right) = g_1\left(x\right) \\ & u_t\left(x, 0\right) = g_2\left(x\right). \end{aligned} \tag{6.30}$$

We chose $f\left(u\right) = ku - \gamma u^3$ and a domain $[-10, 10] \times [0, 1]$. We used Chebyshev pseudospectral collocation method for the discretization of space and time with



(A) Numerically calculated solution.

(B) Absolute error plot .

Figure 6.3: Kein-Gordon eqaution, domain= $[-10,\ 10] \times [0,\ 1]$, grid points in spatial dimension= 120, grid points in temporal dimension= 30.

120 grid points for space and 30 grid points for time. The resulting problem size was 3600. Table 6.8 shows the results. The FTUC method clearly outperforms the HJ method. The exact solution can be written as

$$\begin{aligned}&\delta = \sqrt{\tfrac{2k}{\gamma}}\\&\kappa = \sqrt{\tfrac{k}{c^2-v^2}}\\&u\left(x,t\right) = \delta\,\mathrm{sech}\left(\kappa\left(x-vt\right)\right),\end{aligned} \tag{6.31}$$

where $c = 1$, $\gamma = 1$, $v = 0.5$ and $k = 0.5$.

Iterative methods		FTUC	HJ
Number of iterations		1	1
Number of steps		9	10
Theoretical convergence order		23	20
Number of function evaluations per iteration		8	9
Solutions of linear systems per iteration		16	19
Number of matrix vector multiplication per iteration		8	10
	Iteration		
$\|\|\boldsymbol{u}_q - \boldsymbol{u}^*\|\|_1$	1	$4.08e-1$	$5.99e-1$
	2	$5.67e-1$	$4.15e-2$
	3	$6.45e-2$	$2.34e-3$
	4	$3.62e-3$	$6.72e-5$
	5	$1.06e-4$	$1.15e-6$
	6	$1.84e-6$	$1.31e-8$
	9	$9.63e-11$	$1.24e-10$
	10		$9.12e-11$
Execution time (s)		4.91	5.34

Table 6.8: Performance of comparison between FTUC and HJ for Klein-Gordon problem.

6.5.4 Two-dimensional sinh-Poisson equation

The stationary two-dimensional Euler flow free of body forces satisfy [76]

$$\nabla^2\phi + \sigma \text{sinh}\left(\phi\right) = 0, \quad \sigma > 0. \tag{6.32}$$

The analytical solution of (6.32) for $\sigma = 1$ is given in [101].

$$\phi\left(x, y\right) = 4\text{tanh}^{-1}\left[\frac{\beta\cos\left(\sqrt{1+\beta^2}x\right)}{\sqrt{1+\beta^2}\cosh\left(\beta y\right)}\right]. \tag{6.33}$$

That solution is called the Mallier-Maslowe vortices for $\sigma = 1$. We choose a domain $[-1.3, 1.3] \times [-1.3, 1.3]$ with 30 grid points for each dimension. That resulted in a problem size 500. The computed solution corresponds to $\beta = 0.5$. Table 6.9 shows the obtained results. The FTUC method achieves almost the same accuracy as the HJ method with a smaller number of iterations and, consequently, a smaller execution time. Figure 6.4 depicts the numerically calculated solution and the absolute error over the spatial grid.

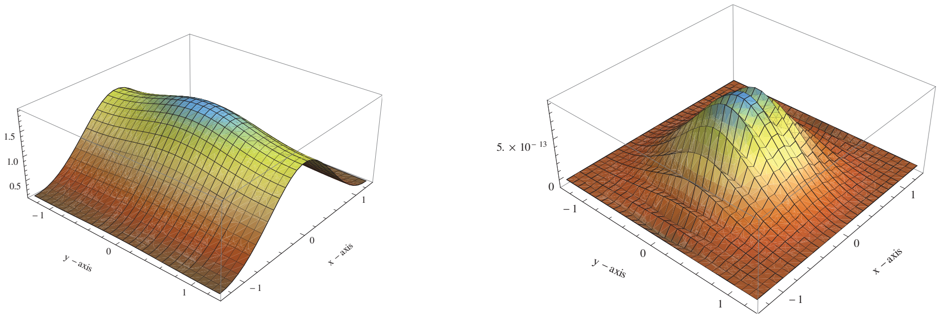
Iterative methods	FTUC	HJ
Number of iterations	1	1
Number of steps	111	183
Theoretical convergence order	329	366
Number of function evaluations per iteration	110	182
Solutions of linear systems per iteration	220	365
Number of matrix vector multiplication per iteration	110	183
$\|\|\boldsymbol{\phi}_q - \boldsymbol{\phi}^*\|\|_1$	$9.22e-13$	$8.81e-13$
Execution time (s)	42.87	68.225

Table 6.9: Performance of FTUC and HJ for the sinh-Poisson equation.

6.5.5 Three-dimensional nonlinear Poisson equation

The numerical solution of nonlinear 3-D nonlinear Poisson equation is treated in [77]. The governing equation is

$$\nabla \cdot \left(K\left(u\right)\nabla u\right) - g = 0\,, \tag{6.34}$$



(A) Solution (B) Absolute error

Figure 6.4: Solution of sinh-Poisson equation andf absolute error in the computed solution.

where $K(u)$ can be any function of u and g is a force term. We will adopt the expression for $K(u)$ from [77]

$$K(u) = \frac{100 + 27u}{300 + 27u}. \tag{6.35}$$

We took a domain $[-1, 1]^3$ and constructed g in a way that makes $u = x^2 + y^2 + z^2$ the solution of (6.34). The grid points were taken so that the problem size is 1331. Table 6.10 gives the obtained results. FTUC achieves a similar accuracy as HJ with smaller number of steps and a smaller execution time. Figure 6.5 plots the maximum absolute error in the solution vector $\mathbf{u}(\mathbf{x})$ against the number of steps.

Iterative methods	FTUC	HJ
Number of iterations	1	1
Number of steps	13	16
Theoretical convergence order	35	32
Number of function evaluations per iteration	12	15
Solutions of linear systems per iteration	24	31
Number of matrix vector multiplication per iteration	12	16
$\|\|\boldsymbol{u}_q - \boldsymbol{u}^*\|\|_1$	$5.77e-15$	$5.32e-15$
Execution time (s)	10.608	11.646

Table 6.10: Performance of FTUC and HJ for nonlinear Poisson equation.

6.6 Summary

Multi-step iterative methods with a large number of steps are computationally efficient because the computational cost required to perform the additional steps

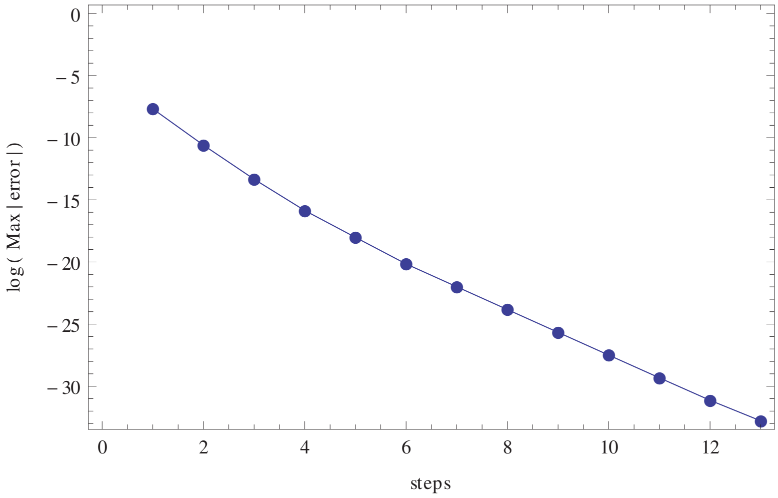


Figure 6.5: $log\left(\|\mathbf{u}\left(\mathbf{x}\right)-\mathbf{u}^{*}\|_{1}\right)$ as a function of the number of steps.

is small. The increase of convergence order in the multi-step part depends on the base method. So, the design of the base method is crucial. All numerically conducted tests conclude that the proposed FTUC multi-step iterative method requires relatively less execution time to achieve same numerical accuracy than HJ.

Chapter 7

Higher order derivative-free iterative methods with memory for systems of nonlinear equations

A derivative-free family of iterations without memory consisting of three steps for solving nonlinear systems of equations is brought forward. The main aim of the chapter consists in proposing several novel schemes with memory, possessing higher R-orders of convergence. Analytical discussions are reported and the theoretical efficiency of the methods is studied in detail. The use of the proposed schemes for solving partial differential equations are finally considered, in order to support the theoretical discussions from a practical viewpoint.

7.1 Introduction

The solution $\boldsymbol{\alpha}$ of a nonlinear equation $\mathbf{F}(\mathbf{x}) = \mathbf{0}$, where $\mathbf{F} : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$ is a sufficiently smooth function, cannot be expressed in a closed form and therefore it is mostly pursued by applying iterative methods of various natures depending on the nonlinear functions, size of the systems and the accuracy of the sought-after results.

By assuming that $\mathbf{F}$ has at least third-order Fréchet derivatives with continuity on a convex set D and by starting from one or several initial solutions of $\boldsymbol{\alpha} \in D$, a sequence of approximations $\{\mathbf{x}_k\}$ is constructed so that it converges to $\boldsymbol{\alpha}$.

The best-known iterative scheme for this purpose is Newton's method defined by [1]:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{F}'(\mathbf{x}_k)^{-1}\mathbf{F}(\mathbf{x}_k), \qquad k = 0, 1, 2, \cdots . \tag{7.1}$$

It is observable that the operator $\mathbf{F}$ must be differentiable Fréchet in order to apply Newton's method. In some practical physical problems, for instance, the simulation of laminar premixed flames [4], it is hard to know the analytical expressions for Jacobian matrix and then only methods come to play are derivative-free iterative methods.

Perhaps, the first try to get rid of the computation of Fréchet derivative was done by Schmidt [50] in 1961 at which he extended the Secant method for nonlinear systems by defining the divided difference operator (DDO). It is now reminded that the definition given by Ortega and Rheinboldt [3] for a first-order divided difference of $\mathbf{F}$ on $\mathbb{R}^n$ is a mapping as follows:

$$[\cdot, \cdot; \mathbf{F}] : D \subset \mathbb{R}^n \times \mathbb{R}^n \to \mathcal{L}(\mathbb{R}^n), \tag{7.2}$$

which satisfies $[\mathbf{y}, \mathbf{x}; \mathbf{F}](\mathbf{y} - \mathbf{x}) = \mathbf{F}(\mathbf{y}) - \mathbf{F}(\mathbf{x})$, $\forall \mathbf{x}, \mathbf{y} \in D$.

Defining $\mathbf{h} = \mathbf{y} - \mathbf{x}$, one may define the DDO of the first-order as follows [41, 45]:

$$[\mathbf{x} + \mathbf{h}, \mathbf{x}; \mathbf{F}] = \int_0^1 \mathbf{F}'(\mathbf{x} + t\,\mathbf{h})dt, \tag{7.3}$$

which is known as Genocchi-Hermite formula. Now by writing the Taylor expansion of $\mathbf{F}'(\mathbf{x} + t\,\mathbf{h})$ at the point x and integrating, we have that

$$\int_0^1 \mathbf{F}'(\mathbf{x} + t\,\mathbf{h})dt = \mathbf{F}'(\mathbf{x}) + \frac{1}{2}\mathbf{F}''(\mathbf{x})\mathbf{h} + \frac{1}{6}\mathbf{F}'''(\mathbf{x})\mathbf{h}^2 + \mathcal{O}(h^3). \tag{7.4}$$

Clearly, if we assume $\mathbf{e} = \mathbf{x} - \boldsymbol{\alpha}$ and the fact that $\mathbf{F}'(\boldsymbol{\alpha})$ is nonsingular, then it would be straightforward to write

$$\mathbf{F}(\mathbf{x}) = \mathbf{F}(\boldsymbol{\alpha} + \mathbf{e}) = \mathbf{A}_1 \left(\mathbf{e} + \sum_{q=2}^{q=8} \mathbf{A}_q\, \mathbf{e}_q + \mathcal{O}(\,\mathbf{e}^9) \right), \tag{7.5}$$

whereas $\mathbf{A}_1 = \mathbf{F}'(\boldsymbol{\alpha})$, $\mathbf{A}_p = \frac{1}{p!}\mathbf{A}_1^{-1}\mathbf{F}^{(p)}(\boldsymbol{\alpha}) \in \mathcal{L}_p(\mathbb{R}^n, \mathbb{R}^n)$. From (7.5) the derivatives of $\mathbf{F}(\mathbf{x})$ can be written as

$$\mathbf{F}'(\mathbf{x}) = \mathbf{A}_1 \left(\mathbf{I} + \sum_{q=2}^{q=7} q\mathbf{A}_q\, \mathbf{e}_{q-1} + \mathcal{O}(\, \mathbf{e}^8) \right), \tag{7.6}$$

$$\mathbf{F}''(\mathbf{x}) = \mathbf{A}_1 \left(\sum_{q=2}^{q=6} q!\mathbf{A}_q\, \mathbf{e}_{q-2} + \mathcal{O}(\, \mathbf{e}^7) \right). \tag{7.7}$$

It is necessary to recall that when studying the theoretical local order of convergence of iterative methods for nonlinear systems, $\mathbf{e}_k = \mathbf{x}_k - \boldsymbol{\alpha}$ is the error in the kth iterate and

$$\mathbf{e}_{k+1} = \mathbf{L}\, \mathbf{e}_{(k)^p} + \mathcal{O}(\, \mathbf{e}_{(k)^{p+1}}), \tag{7.8}$$

is the error equation, where $\mathbf{L}$ is a p-linear function. That is to say, $\mathbf{L} \in \mathcal{L}(\mathbb{R}^n, \mathbb{R}^n, \ldots, \mathbb{R}^n)$, p is the local convergence rate and $\mathcal{L}$ shows the set of bounded linear functions. Furthermore, we have $\mathbf{e}_k^p{=}(\overbrace{\mathbf{e}_k,\ \mathbf{e}_k, \ldots,\ \mathbf{e}_k}^{p})$. The generalized Steffensen's method for nonlinear system of equations can be expressed as [1]:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - [\mathbf{x}_k, \mathbf{w}_k, \mathbf{F}]^{-1}\mathbf{F}(\mathbf{x}_k), \qquad k = 0, 1, 2, \cdots, \tag{7.9}$$

wherein $\mathbf{w}_k = \mathbf{x}_k + \mathbf{F}(\mathbf{x}_k)$. Note that the first order DDO of $\mathbf{F}$ on the points $\mathbf{x}$ and $\mathbf{y}$ can be defined component-to-component as follows:

$$[\mathbf{x}, \mathbf{y}; \mathbf{F}]_{i,j} = \frac{\mathbf{F}_i(x_1, \ldots, x_j, y_{j+1}, \ldots, y_n) - \mathbf{F}_i(x_1, \ldots, x_{j-1}, y_j, \ldots, y_n)}{x_j - y_j}, \qquad 1 \le i, j \le n\,. \tag{7.10}$$

This formula is a bounded linear operator which satisfies $[\mathbf{y}, \mathbf{x}; \mathbf{F}](\mathbf{y}-\mathbf{x}) = \mathbf{F}(\mathbf{y}) - \mathbf{F}(\mathbf{x})$. Due to Potra [49], there is a necessary and sufficient condition to characterize the divided difference operator by means of a Riemann integral. That is, if $\mathbf{F}$ satisfies the following Lipschitz condition $\|[\mathbf{x}, \mathbf{y}; \mathbf{F}] - [\boldsymbol{u}, \mathbf{v}; \mathbf{F}]\| \le H(\|\mathbf{x} - \boldsymbol{u}\| + \|\mathbf{y} - \mathbf{v}\|)$, then equality (7.4) holds for every pair of distinct points $(\mathbf{x} + \mathbf{h}, \mathbf{x}) \in D \times D$ if and only if for all $(\boldsymbol{u}, \mathbf{v}) \in D \times D$ with $\boldsymbol{u} \neq \mathbf{v}$ and $2\mathbf{v} - \boldsymbol{u} \in D$, it is satisfied the relation

$$[\boldsymbol{u}, \mathbf{v}; \mathbf{F}] = 2[\boldsymbol{u}, 2\mathbf{v} - \boldsymbol{u}; \mathbf{F}] - [\mathbf{v}, 2\mathbf{v} - \boldsymbol{u}; \mathbf{F}]. \tag{7.11}$$

Note that authors in [42] developed a symmetric operator for the DDO of the second-order, but of course with more computational cost than the first-order DDO.

In the last years, different procedures have been used in the development of iterative methods for nonlinear systems, see for example the paper of Budzkoa et al. [38], the schemes published in [52] for solving the systems of nonlinear equations obtained during the process of solving stochastic differential equations, findings of the authors in [40] of how to study the dynamical behavior of different methods in multi-dimensional case, the recent paper [57] and the work [37, 65] were the authors applied the designed methods on solving ordinary and partial differential equations.

In 2014, Sharma et al. in [51] proposed a derivative-free iterative method with fourth order of convergence in what follows:

$$\begin{cases} \mathbf{y}_k = \mathbf{x}_k - [\mathbf{x}_k, \mathbf{w}_k, \mathbf{F}]^{-1}\mathbf{F}(\mathbf{x}_k), \\ \mathbf{x}_{k+1} = \mathbf{y}_k - (a\mathbf{I} + \mathbf{G}^{(k)}((3-2a)\mathbf{I} + (a-2)\mathbf{G}^{(k)}))[\mathbf{x}_k, \mathbf{w}_k, \mathbf{F}]^{-1}\mathbf{F}(\mathbf{y}_k), \end{cases} \tag{7.12}$$

wherein $\mathbf{w}_k = \mathbf{x}_k + b\,\mathbf{F}(\mathbf{x}_k)$, $\mathbf{z}_k = \mathbf{y}_k + c\,\mathbf{F}(\mathbf{y}_k)$, $a \in \mathbb{R}$, $b, c \in \mathbb{R}\backslash\{0\}$, $\mathbf{G}^{(k)} = [\mathbf{x}_k, \mathbf{w}_k, \mathbf{F}]^{-1}[\mathbf{z}_k, \mathbf{y}_k, \mathbf{F}]$ and $\mathbf{I}$ is the identity matrix of the appropriate size.

Recently, authors in [48] improved the convergence speed of (7.12) to $2+\sqrt{5}$ when $a \neq 3$ and $2+\sqrt{6}$ when $a = 3$ by considering:

$$\begin{cases} \mathbf{B}^{(k)} = -[\mathbf{w}_{k-1}, \mathbf{x}_{k-1}; \mathbf{F}]^{-1}, \qquad k \geq 1, \\ \mathbf{y}_k = \mathbf{x}_k - [\mathbf{x}_k, \mathbf{w}_k, \mathbf{F}]^{-1}\mathbf{F}(\mathbf{x}_k), \qquad k \geq 0, \\ \mathbf{x}_{k+1} = \mathbf{y}_k - \\ \qquad (a\mathbf{I} + \mathbf{G}^{(k)}((3-2a)\mathbf{I} + (a-2)\mathbf{G}^{(k)}))[\mathbf{x}_k, \mathbf{w}_k, \mathbf{F}]^{-1}\mathbf{F}(\mathbf{y}_k), \end{cases} \tag{7.13}$$

where $\mathbf{w}_k = \mathbf{x}_k + \mathbf{B}^{(k)}\mathbf{F}(\mathbf{x}_k)$.

There are a few works related to iterative methods with memory of high orders for nonlinear systems. Accordingly, in this work we design a new iterative family of methods without memory which is also economic in terms of computational cost. Then, accelerations of this scheme are derived using the idea of with memorization without any *additional* computational cost per cycle. We manifest that the proposed methods are more efficient than several existing derivative-free methods with and without memory in the literature.

7.2 A new family of iteration schemes

The iterative methods (7.1), (7.9) and (7.12) belong to the class of methods without memory since they use only data from the current iteration. On the other hand, (7.12) requires evaluations of four functions, two divided differences and one inverse operator per one cycle and thus it is not that efficient in terms of computational costs.

The proposed iterative family of methods can be constructed as a generalization of the scheme (7.12) but with three sub-steps and a much more care to use the data as follows:

$$\begin{cases} \mathbf{y}_k = \mathbf{x}_k - \boldsymbol{\phi}_1, \\ \mathbf{z}_k = \mathbf{y}_k - a_0\boldsymbol{\phi}_2 - (3-2a_0)\boldsymbol{\phi}_3 - (a_0-2)\boldsymbol{\phi}_4, \\ \mathbf{x}_{k+1} = \mathbf{z}_k - a_1\boldsymbol{\phi}_5 - a_2\boldsymbol{\phi}_6 - a_3\boldsymbol{\phi}_7 - a_4\boldsymbol{\phi}_8 - a_5\boldsymbol{\phi}_9, \end{cases} \tag{7.14}$$

wherein $\mathbf{w}_k = \mathbf{x}_k + \mathbf{b}_0\mathbf{F}(\mathbf{x}_k)$, $\mathbf{M}^{(k)} = [\mathbf{x}_k, \mathbf{w}_k, \mathbf{F}]$, $\mathbf{h}_k = \mathbf{y}_k + \mathbf{b}_1\mathbf{F}(\mathbf{y}_k)$, $\mathbf{N}^{(k)} = [\mathbf{h}_k, \mathbf{y}_k, \mathbf{F}]$, $\mathbf{l}_k = \mathbf{z}_k + \mathbf{b}_2\mathbf{F}(\mathbf{z}_k)$, and $\mathbf{Q}^{(k)} = [\mathbf{l}_k, \mathbf{z}_k, \mathbf{F}]$, while the following linear systems must also be solved per cycle

$$\begin{cases} \mathbf{M}^{(k)}\boldsymbol{\phi}_1 = \mathbf{F}(\mathbf{x}_k), \\ \mathbf{M}^{(k)}\boldsymbol{\phi}_2 = \mathbf{F}(\mathbf{y}_k), \\ \mathbf{M}^{(k)}\boldsymbol{\phi}_3 = \mathbf{N}^{(k)}\boldsymbol{\phi}_2, \\ \mathbf{M}^{(k)}\boldsymbol{\phi}_4 = \mathbf{N}^{(k)}\boldsymbol{\phi}_3, \end{cases} \tag{7.15}$$

and

$$\begin{cases} \mathbf{M}^{(k)}\boldsymbol{\phi}_5 = \mathbf{F}(\mathbf{z}_k), \\ \mathbf{M}^{(k)}\boldsymbol{\phi}_6 = \mathbf{Q}^{(k)}\boldsymbol{\phi}_5, \\ \mathbf{M}^{(k)}\boldsymbol{\phi}_7 = \mathbf{Q}^{(k)}\boldsymbol{\phi}_6, \\ \mathbf{M}^{(k)}\boldsymbol{\phi}_8 = \mathbf{Q}^{(k)}\boldsymbol{\phi}_7, \\ \mathbf{M}^{(k)}\boldsymbol{\phi}_9 = \mathbf{Q}^{(k)}\boldsymbol{\phi}_8, \end{cases} \tag{7.16}$$

where $a_1 = 4 + a_5$, $a_2 = -6 - 4a_5$, $a_3 = 4 + 6a_5$ and $a_4 = -1 - 4a_5$. The eminent point is that due to the same coefficient matrix $\mathbf{M}^{(k)}$, only one LU decomposition could be performed for solving linear systems with multiple right hand sides.

Hence, the iteration scheme (7.14) without memory includes three steps, and five free (non-zero matrix) parameters which make it quite general and useful for solving nonlinear systems and of course for with memorization, as will be seen in the next section.

Theorem 1. Let $\mathbf{F}$ have at least three times Fréchet differentiable in the nonempty open convex domain D. Also suppose that $[\boldsymbol{u}, \mathbf{v}; \mathbf{F}] \in \mathcal{L}(D, D)$, for all $\boldsymbol{u}, \mathbf{v} \in D(\boldsymbol{u} \neq \mathbf{v})$ and $(\mathbf{x}^{(0)}$ is close enough to $\boldsymbol{\alpha}$. Then, the sequence $\{\mathbf{x}_k\}_{k\geq 0}$ obtained using the iterative expression (7.14) converges to $\boldsymbol{\alpha}$ with at least eight order of convergence.

Proof. The proof of this theorem can be followed by writing the Taylor expansions of $\mathbf{F}$ around the appropriate points. Now, it is straightforward to write

$$\mathbf{F}(\boldsymbol{\alpha} + \mathfrak{h}) = \mathbf{F}'(\boldsymbol{\alpha}) \left(\mathfrak{h} + \sum_{q=2}^{p-1} \mathbf{A}_q \mathfrak{h}^q \right) + \mathcal{O}\left(\mathfrak{h}^p\right). \tag{7.17}$$

We observe that $\mathbf{A}_q\mathfrak{h}^q \in \mathbb{R}^n$ since $\mathbf{F}^{(q)}(\boldsymbol{\alpha}) \in \mathcal{L}(\mathbb{R}^n \times \cdots \times \mathbb{R}^n, \mathbb{R}^n)$ and $\mathbf{F}'(\boldsymbol{\alpha})^{-1} \in \mathcal{L}(\mathbb{R}^n)$. Note that matrix-matrix products are not commutative. Subsequently, we acquire $\mathbf{F}(\mathbf{x}_k)= \mathbf{F}(\mathbf{x}_k - \boldsymbol{\alpha} + \boldsymbol{\alpha}) = \mathbf{F}(\,\mathbf{e}_k + \boldsymbol{\alpha}) = \mathbf{F}(\boldsymbol{\alpha}) + \mathbf{F}'(\boldsymbol{\alpha})\;\mathbf{e}_k + \mathbf{F}''(\boldsymbol{\alpha})/2!\;\mathbf{e}_k^2 + \cdots + \mathcal{O}(\,\mathbf{e}_k^9) = \mathbf{F}'(\boldsymbol{\alpha})(\,\mathbf{e}_k + \mathbf{A}_2\;\mathbf{e}_k^2 + \mathbf{A}_3\;\mathbf{e}_k^3 + \mathbf{A}_4\;\mathbf{e}_k^4) + \cdots + \mathcal{O}(\,\mathbf{e}_k^9)$, by taking into consideration $\mathbf{w}_k = \mathbf{x}_k + \mathbf{b}_0\,\mathbf{F}(\mathbf{x}_k)$, we write

$$\begin{aligned}\mathbf{F}(\mathbf{w}_k) = \mathbf{F}'(\boldsymbol{\alpha})\Big(&\mathbf{d}_1\;\mathbf{e}_k + \Big(- \mathbf{A}_2 + \mathbf{d}_1\mathbf{A}_2 + \mathbf{A}_2\mathbf{d}_1^2\Big)\,\mathbf{e}_k^2 + \\ &\Big(- \mathbf{A}_3 - \mathbf{A}_2^2\,\mathbf{d}_1 - \mathbf{A}_2\,\mathbf{d}_1\mathbf{A}_2 + \mathbf{A}_2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 \\ &+ \mathbf{d}_1\mathbf{A}_3 + \mathbf{A}_2\mathbf{d}_1^2\mathbf{A}_2 + \mathbf{A}_3\mathbf{d}_1^3\Big)\,\mathbf{e}_k^3 + \cdots\Big) + \mathcal{O}\left(\,\mathbf{e}_k^9\right),\end{aligned} \tag{7.18}$$

where $\mathbf{d}_1 = \mathbf{I} + \mathbf{b}_0\mathbf{F}'(\boldsymbol{\alpha})$. We define also $\mathbf{d}_2 = \mathbf{I} + \mathbf{b}_1\mathbf{F}'(\boldsymbol{\alpha})$ and $\mathbf{d}_3 = \mathbf{I} + \mathbf{b}_2\mathbf{F}'(\boldsymbol{\alpha})$. The detail of further computations is given below in sequence. We can obtain

$$\begin{aligned}\mathbf{M}^{(k)} = \mathbf{F}'(\alpha)\Big(&\mathbf{I} + \Big(\mathbf{A}_2\mathbf{d}_1 + \mathbf{A}_2\Big)\,\mathbf{e}_k + \Big(\mathbf{A}_3\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_1\mathbf{A}_2 + \mathbf{A}_3\mathbf{d}_1^2 - \mathbf{A}_2^2 \\ &+ \mathbf{A}_3\Big)\,\mathbf{e}_k^2 + \Big(\mathbf{A}_4\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_1\mathbf{A}_3 + \mathbf{A}_4\mathbf{d}_1^2 + \mathbf{A}_3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 + \mathbf{A}_3\mathbf{d}_1^2\mathbf{A}_2 + \\ &\mathbf{A}_4\mathbf{d}_1^3 - \mathbf{A}_3\mathbf{A}_2 - \mathbf{A}_2\mathbf{A}_3 - \mathbf{A}_3\mathbf{A}_2\mathbf{d}_1 + \mathbf{A}_4\Big)\,\mathbf{e}_k^3 + \cdots\Big) + \mathcal{O}\left(\,\mathbf{e}_k^9\right),\end{aligned} \tag{7.19}$$

and obviously

$$\begin{aligned}\mathbf{M}^{(k)^{-1}} =&\Big(\mathbf{I}+\Big(-\mathbf{A}_2\mathbf{d}_1-\mathbf{A}_2\Big)\mathbf{e}_k+\Big(-\mathbf{A}_3\mathbf{d}_1+2\mathbf{A}_2^2-\mathbf{A}_3\mathbf{d}_1^2+\\&(\mathbf{A}_2\mathbf{d}_1)^2+\mathbf{A}_2^2\mathbf{d}_1-\mathbf{A}_3\Big)\mathbf{e}_k^2+\Big(-\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2-\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1-\\&3\mathbf{A}_2^3-2\mathbf{A}_2^3\mathbf{d}_1-(\mathbf{A}_2\mathbf{d}_1)^3-\mathbf{A}_4\mathbf{d}_1+2\mathbf{A}_3\mathbf{A}_2+2\mathbf{A}_2\mathbf{A}_3-\\&\mathbf{A}_4\mathbf{d}_1^2+2\mathbf{A}_3\mathbf{A}_2\mathbf{d}_1-\mathbf{A}_4\mathbf{d}_1^3+\mathbf{A}_3\mathbf{d}_1\mathbf{A}_2+\mathbf{A}_3\mathbf{d}_1^2\mathbf{A}_2\mathbf{d}_1+\\&\mathbf{A}_2\mathbf{A}_3\mathbf{d}_1+\mathbf{A}_2\mathbf{A}_3\mathbf{d}_1^2+\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1+\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1^2-\mathbf{A}_4\Big)\mathbf{e}_k^3+\\&\cdots\Big)\mathbf{F}'(\boldsymbol{\alpha})^{-1}+\mathcal{O}\left(\mathbf{e}_k^9\right).\end{aligned}\tag{7.20}$$

It is remarked that we used the notation $\cdots$ in order to avoid huge cumbersome terms obtained for high degree terms in Taylor expansion, since they will finally be vanished as could be observed below. Now, the error equation at the end of the first sub-step can be written in what follows:

$$\begin{aligned}\mathbf{y}_k-\boldsymbol{\alpha}=&\ \mathbf{A}_2\mathbf{d}_1\,\mathbf{e}_k^2-\Big(\mathbf{A}_2^2\mathbf{d}_1+(\mathbf{A}_2\mathbf{d}_1)^2-\mathbf{A}_3\mathbf{d}_1^2+\mathbf{A}_2^2-\mathbf{A}_3\mathbf{d}_1\\&-\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2\Big)\mathbf{e}_k^3-\Big(-\mathbf{A}_3\mathbf{d}_1^2\mathbf{A}_2+2\mathbf{A}_3\mathbf{A}_2\mathbf{d}_1-\mathbf{A}_4\mathbf{d}_1^2+\mathbf{A}_3\mathbf{A}_2+\\&\mathbf{A}_2\mathbf{A}_3-\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3-\mathbf{A}_4\mathbf{d}_1-\mathbf{A}_4\mathbf{d}_1^3+(\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2-\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2-\\&\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1+\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2+\mathbf{A}_3\mathbf{d}_1^2\mathbf{A}_2\mathbf{d}_1-\mathbf{A}_2^3-2\mathbf{A}_2^3\mathbf{d}_1+\mathbf{A}_2\mathbf{A}_3\mathbf{d}_1\\&-(\mathbf{A}_2\mathbf{d}_1)^3+\mathbf{A}_2\mathbf{A}_3\mathbf{d}_1^2+\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1+\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1^2\Big)\mathbf{e}_k^4+\\&\cdots+\mathcal{O}\left(\mathbf{e}_k^9\right).\end{aligned}\tag{7.21}$$

Using (7.21), one may attain

$$\begin{aligned}\mathbf{F}(\mathbf{y}_k)=&\ \mathbf{F}'(\boldsymbol{\alpha})\Big(\mathbf{A}_2\mathbf{d}_1\,\mathbf{e}_k^2+\Big(-\mathbf{A}_2^2\mathbf{d}_1-(\mathbf{A}_2\mathbf{d}_1)^2+\mathbf{A}_3\mathbf{d}_1^2-\mathbf{A}_2^2+\\&\mathbf{A}_3\mathbf{d}_1+\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2\Big)\mathbf{e}_k^3+\Big(\mathbf{A}_3\mathbf{d}_1^2\mathbf{A}_2-2\mathbf{A}_3\mathbf{A}_2\mathbf{d}_1+\mathbf{A}_4\mathbf{d}_1^2\\&-\mathbf{A}_3\mathbf{A}_2-\mathbf{A}_2\mathbf{A}_3+\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3+\mathbf{A}_4\mathbf{d}_1+\mathbf{A}_4\mathbf{d}_1^3-(\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2+\\&\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2+2\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1-\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2-\mathbf{A}_3\mathbf{d}_1^2\mathbf{A}_2\mathbf{d}_1+\mathbf{A}_2^3+2\mathbf{A}_2^3\mathbf{d}_1\\&-\mathbf{A}_2\mathbf{A}_3\mathbf{d}_1+(\mathbf{A}_2\mathbf{d}_1)^3-\mathbf{A}_2\mathbf{A}_3\mathbf{d}_1^2-\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1-\\&\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1^2\Big)\mathbf{e}_k^4+\cdots\Big)+\mathcal{O}\left(\mathbf{e}_k^9\right),\end{aligned}\tag{7.22}$$

and subsequently

$$\begin{aligned}\mathbf{h}_k = \mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\,\mathbf{e}_k^2 + \Big(&-\mathbf{d}_2\mathbf{A}_2^2\mathbf{d}_1 - \mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^2 - \mathbf{d}_2\mathbf{A}_2^2 + \mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2 \\ &+ \mathbf{d}_2\mathbf{A}_3\mathbf{d}_1 + \mathbf{d}_2\mathbf{A}_3\mathbf{d}_1^2\Big)\,\mathbf{e}_k^3 + \Big(-\mathbf{d}_2\mathbf{A}_3\mathbf{A}_2 - \mathbf{d}_2\mathbf{A}_2\mathbf{A}_3 \\ &- \mathbf{d}_2\mathbf{A}_3\mathbf{d}_1^2\mathbf{A}_2\mathbf{d}_1 + 2\mathbf{d}_2\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 - \mathbf{d}_2\mathbf{A}_2\mathbf{A}_3\mathbf{d}_1^2 - \mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1^2 + \\ &2\mathbf{d}_2\mathbf{A}_2^3\mathbf{d}_1 - \mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2 - \mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1 - \mathbf{d}_2\mathbf{A}_2\mathbf{A}_3\mathbf{d}_1 - \mathbf{d}_2\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2 \\ &- \mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 - 2\mathbf{d}_2\mathbf{A}_3\mathbf{A}_2\mathbf{d}_1 + \mathbf{d}_2\mathbf{A}_4\mathbf{d}_1^2 + \mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3 + \mathbf{d}_2\mathbf{A}_4\mathbf{d}_1^3 + \\ &\mathbf{d}_2\mathbf{A}_4\mathbf{d}_1 + \mathbf{d}_2\mathbf{A}_3\mathbf{d}_1^2\mathbf{A}_2 + \mathbf{d}_2\mathbf{A}_2^3 + \mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^3 + \mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2\Big)\,\mathbf{e}_k^4 \\ &+ \cdots + \mathcal{O}\left(\mathbf{e}_k^9\right).\end{aligned} \tag{7.23}$$

It is now necessary to obtain the error equation of $\mathbf{N}^{(k)}$ by applying (7.23) as comes next

$$\begin{aligned}\mathbf{N}^{(k)} = \mathbf{F}'(\boldsymbol{\alpha})\Big(\mathbf{I} + \Big(&\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1 + \mathbf{A}_2^2\mathbf{d}_1\Big)\,\mathbf{e}_k^2 + \Big(-\mathbf{A}_2^3 - \mathbf{A}_2^3\mathbf{d}_1 \\ &- \mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 - \mathbf{A}_2\mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^2 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1^2 + \\ &\mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1 + \mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2 + \mathbf{A}_2\mathbf{A}_3\mathbf{d}_1^2 + \mathbf{A}_2\mathbf{A}_3\mathbf{d}_1 - \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2^2\mathbf{d}_1 \\ &- \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2^2\Big)\,\mathbf{e}_k^3 + \cdots\Big) + \mathcal{O}\left(\mathbf{e}_k^9\right).\end{aligned} \tag{7.24}$$

Using the above expansions, we have

$$\begin{aligned}\boldsymbol{\phi}_3 =& \mathbf{A}_2\mathbf{d}_1\,\mathbf{e}_k^2 + \Big(-3\mathbf{A}_2^2\mathbf{d}_1 - 3(\mathbf{A}_2\mathbf{d}_1)^2 - \mathbf{A}_2^2 + \mathbf{A}_3\mathbf{d}_1^2 + \mathbf{A}_3\mathbf{d}_1 + \\ &\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2\Big)\,\mathbf{e}_k^3 + \cdots + \mathcal{O}\left(\mathbf{e}_k^9\right),\end{aligned} \tag{7.25}$$

and

$$\begin{aligned}\boldsymbol{\phi}_4 = \mathbf{A}_2\mathbf{d}_1\,\mathbf{e}_k^2 + \Big(&-4(\mathbf{A}_2\mathbf{d}_1)^2 - 4\mathbf{A}_2^2\mathbf{d}_1 + \mathbf{A}_3\mathbf{d}_1^2 + \mathbf{A}_2\mathbf{d}_1\mathbf{A}_2 \\ &- \mathbf{A}_2^2 + \mathbf{A}_3\mathbf{d}_1\Big)\,\mathbf{e}_k^3 + \cdots + \mathcal{O}\left(\mathbf{e}_k^9\right).\end{aligned} \tag{7.26}$$

Thus, the error terms for the second sub-step can be attained as follows:

$$\begin{aligned}\mathbf{z}_k-\boldsymbol{\alpha}=\mathbf{d}_4\,\mathbf{e}_k^4+\Big(&-\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2^3\mathbf{d}_1+(\mathbf{A}_2\mathbf{d}_1)^3\mathbf{A}_2\mathbf{d}_1+(\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2^2\mathbf{d}_1\\&+\mathbf{A}_2^2\mathbf{d}_1(\mathbf{A}_2\mathbf{d}_1)^2+\mathbf{A}_2^4\mathbf{d}_1+\mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1^2\mathbf{A}_2\mathbf{d}_1+\\&\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1^2+\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^3\mathbf{d}_1+\mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1+\mathbf{A}_2^3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1\\&+\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1+\mathbf{A}_2\mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2+\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1\\&-2\mathbf{A}_2\mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^3-\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1+\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1\\&-\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2\Big)\,\mathbf{e}_k^5+\cdots+\mathcal{O}\left(\mathbf{e}_k^9\right),\end{aligned}\tag{7.27}$$

where $\mathbf{d}_4=(3-a)\left(\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1+(\mathbf{A}_2\mathbf{d}_1)^3+\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1+\mathbf{A}_2^3\mathbf{d}_1\right)+\mathbf{A}_2\mathbf{d}_2\left(\mathbf{A}_2\mathbf{d}_1\right)^2$. Now, we have

$$\begin{aligned}\mathbf{l}_k=\mathbf{d}_3\mathbf{d}_4\,\mathbf{e}_k^4+\Big(&\mathbf{d}_3\mathbf{A}_2^2\mathbf{d}_1(\mathbf{A}_2\mathbf{d}_1)^2+\mathbf{d}_3(\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2^2\mathbf{d}_1+\\&\mathbf{d}_3(\mathbf{A}_2\mathbf{d}_1)^3\mathbf{A}_2\mathbf{d}_1-\mathbf{d}_3\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2^3\mathbf{d}_1+\mathbf{d}_3\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^3\mathbf{d}_1+\\&\mathbf{d}_3\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1+\mathbf{d}_3\mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1^2\mathbf{A}_2\mathbf{d}_1+\mathbf{d}_3\mathbf{A}_2^4\mathbf{d}_1+\mathbf{d}_3\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1\\&+\mathbf{d}_3\mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1-2\mathbf{d}_3\mathbf{A}_2\mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^3-\mathbf{d}_3\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2+\\&\mathbf{d}_3\mathbf{A}_2\mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2+\mathbf{d}_3\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1+\mathbf{d}_3\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1^2+\\&\mathbf{d}_3\mathbf{A}_2^3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1-\mathbf{d}_3\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1\Big)\,\mathbf{e}_k^5\\&+\cdots+\mathcal{O}\left(\mathbf{e}_k^9\right),\end{aligned}\tag{7.28}$$

and

$$\begin{aligned}\boldsymbol{\phi}_5=\mathbf{d}_4\,\mathbf{e}_k^4+\Big(&-\mathbf{A}_2\mathbf{d}_1\mathbf{d}_4-\mathbf{A}_2\mathbf{d}_4-\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2^3\mathbf{d}_1+(\mathbf{A}_2\mathbf{d}_1)^3\mathbf{A}_2\mathbf{d}_1+\\&(\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2^2\mathbf{d}_1+\mathbf{A}_2^2\mathbf{d}_1(\mathbf{A}_2\mathbf{d}_1)^2+\mathbf{A}_2^4\mathbf{d}_1+\mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1^2\mathbf{A}_2\mathbf{d}_1+\\&\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1^2+\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^3\mathbf{d}_1+\mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1+\mathbf{A}_2^3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1+\\&\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1+\mathbf{A}_2\mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2+\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1-\\&2\mathbf{A}_2\mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^3-\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1+\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1-\\&\mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2\Big)\,\mathbf{e}_k^5+\cdots+\mathcal{O}\left(\mathbf{e}_k^9\right).\end{aligned}\tag{7.29}$$

Using the above formulas, we attain that

$$\mathbf{Q}^{(k)}=\mathbf{F}'(\boldsymbol{\alpha})\Big(\mathbf{I}+\Big(\mathbf{A}_2\mathbf{d}_4+\mathbf{A}_2\mathbf{d}_3\mathbf{d}_4\Big)\,\mathbf{e}_k^4+\cdots\Big)+\mathcal{O}\left(\mathbf{e}_k^9\right),\tag{7.30}$$

and subsequently

$$\begin{aligned}
\boldsymbol{\phi}_6 = \mathbf{d}_4\,\mathbf{e}_k^4 + \Big(& -2\mathbf{A}_2\mathbf{d}_1\mathbf{d}_4 - 2\mathbf{A}_2\mathbf{d}_4 - \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2^3\mathbf{d}_1 + (\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2^2\mathbf{d}_1 + \\
& \mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1^2\mathbf{A}_2\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1^2 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 + \mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1 + \\
& \mathbf{A}_2\mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1 - 2\mathbf{A}_2\mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^3 - \\
& \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 - \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2 + (\mathbf{A}_2\mathbf{d}_1)^3\mathbf{A}_2\mathbf{d}_1 + \\
& \mathbf{A}_2^2\mathbf{d}_1(\mathbf{A}_2\mathbf{d}_1)^2 + \mathbf{A}_2^4\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^3\mathbf{d}_1 + \mathbf{A}_2^3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1\Big)\,\mathbf{e}_k^5 + \\
& \cdots + \mathcal{O}\left(\,\mathbf{e}_k^9\right),
\end{aligned} \tag{7.31}$$

$$\begin{aligned}
\boldsymbol{\phi}_7 = \mathbf{d}_4\,\mathbf{e}_k^4 + \Big(& -3\mathbf{A}_2\mathbf{d}_1\mathbf{d}_4 - 3\mathbf{A}_2\mathbf{d}_4 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 + \mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1 + \\
& \mathbf{A}_2\mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1 - 2\mathbf{A}_2\mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^3 - \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2 \\
& \mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 - \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2 - \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2^3\mathbf{d}_1 + (\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2^2\mathbf{d}_1 + \\
& \mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1^2\mathbf{A}_2\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1^2 + \mathbf{A}_2^4\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^3\mathbf{d}_1 \\
& + \mathbf{A}_2^3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 + (\mathbf{A}_2\mathbf{d}_1)^3\mathbf{A}_2\mathbf{d}_1 + \mathbf{A}_2^2\mathbf{d}_1(\mathbf{A}_2\mathbf{d}_1)^2\Big)\,\mathbf{e}_k^5 + \\
& \cdots + \mathcal{O}\left(\,\mathbf{e}_k^9\right),
\end{aligned} \tag{7.32}$$

$$\begin{aligned}
\boldsymbol{\phi}_8 = \mathbf{d}_4\,\mathbf{e}_k^4 + \Big(& -4\mathbf{A}_2\mathbf{d}_1\mathbf{d}_4 - 4\mathbf{A}_2\mathbf{d}_4 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 + \mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1 + \\
& \mathbf{A}_2\mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1 - 2\mathbf{A}_2\mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^3 - \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2^2\mathbf{d}_1 \\
& \mathbf{A}_2\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 - \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2 - \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2^3\mathbf{d}_1 + (\mathbf{A}_2\mathbf{d}_1)^2 \\
& \mathbf{A}_2^2\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1^2\mathbf{A}_2\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1^2 + (\mathbf{A}_2\mathbf{d}_1)^3\mathbf{A}_2\mathbf{d}_1 + \\
& \mathbf{A}_2^2\mathbf{d}_1(\mathbf{A}_2\mathbf{d}_1)^2 + \mathbf{A}_2^4\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^3\mathbf{d}_1 + \mathbf{A}_2^3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1\Big)\,\mathbf{e}_k^5 + \\
& \cdots + \mathcal{O}\left(\,\mathbf{e}_k^9\right),
\end{aligned} \tag{7.33}$$

$$\begin{aligned}
\boldsymbol{\phi}_9 = \mathbf{d}_4\,\mathbf{e}_k^4 + \Big(& -5\mathbf{A}_2\mathbf{d}_1\mathbf{d}_4 - 5\mathbf{A}_2\mathbf{d}_4 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 + \\
& \mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1 - 2\mathbf{A}_2\mathbf{d}_2(\mathbf{A}_2\mathbf{d}_1)^3 - \\
& Ab_2\mathbf{d}_2\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1 - \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2 - \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2^3\mathbf{d}_1 + \\
& (\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2^2\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_3\mathbf{d}_1^2\mathbf{A}_2\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_2\mathbf{A}_2\mathbf{d}_1\mathbf{A}_3\mathbf{d}_1^2 + (\mathbf{A}_2\mathbf{d}_1)^3\mathbf{A}_2\mathbf{d}_1 + \\
& \mathbf{A}_2^2\mathbf{d}_1(\mathbf{A}_2\mathbf{d}_1)^2 + \mathbf{A}_2^4\mathbf{d}_1 + \mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^3\mathbf{d}_1 + \mathbf{A}_2^3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1\Big)\,\mathbf{e}_k^5 + \\
& \cdots + \mathcal{O}\left(\,\mathbf{e}_k^9\right).
\end{aligned} \tag{7.34}$$

Therefore, it is now easy to combine the expressions (7.31)-(7.34) and obtain the final error equation

$$\begin{aligned}\mathbf{e}_{k+1} = \Big(&(1-a_5)(\mathbf{A}_2^4+\mathbf{A}_2^4\mathbf{d}_1+\mathbf{A}_2^3\mathbf{d}_1\mathbf{A}_2+\mathbf{A}_2^3\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1+\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2^2+\\ &\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1+\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2+\mathbf{A}_2^2\mathbf{d}_1(\mathbf{A}_2\mathbf{d}_1)^2+\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^3+\\ &\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^3\mathbf{d}_1+\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2+\mathbf{A}_2\mathbf{d}_1\mathbf{A}_2^2\mathbf{d}_1\mathbf{A}_2\mathbf{d}_1+(\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2^2\\ &+(\mathbf{A}_2\mathbf{d}_1)^2\mathbf{A}_2^2\mathbf{d}_1+(\mathbf{A}_2\mathbf{d}_1)^3\mathbf{A}_2+(\mathbf{A}_2\mathbf{d}_1)^3\mathbf{A}_2\mathbf{d}_1)\mathbf{d}_4+\\ &\mathbf{A}_2\mathbf{d}_3\mathbf{d}_4^2\Big)\,\mathbf{e}_k^8+\mathcal{O}\big(\,\mathbf{e}_k^9\big),\end{aligned} \tag{7.35}$$

which shows eighth order of convergence. The proof is now complete. □

It is remarked that by choosing $a_0 = 3$ and $a_5 = 0$, we reduce to the following iterative expression

$$\begin{cases}\mathbf{y}_k = \mathbf{x}_k - \boldsymbol{\phi}_1,\\ \mathbf{z}_k = \mathbf{y}_k - 3\boldsymbol{\phi}_2 + 3\boldsymbol{\phi}_3 - \boldsymbol{\phi}_4,\\ \mathbf{x}_{k+1} = \mathbf{z}_k - 4\boldsymbol{\phi}_5 + 6\boldsymbol{\phi}_6 - 4\boldsymbol{\phi}_7 + \boldsymbol{\phi}_8,\end{cases} \tag{7.36}$$

and by choosing $a_0 = 3$ and $a_5 = 1$, we obtain the following iterative method which is quite fruitful for with memorization due to its fined error equation

$$\begin{cases}\mathbf{y}_k = \mathbf{x}_k - \boldsymbol{\phi}_1,\\ \mathbf{z}_k = \mathbf{y}_k - 3\boldsymbol{\phi}_2 + 3\boldsymbol{\phi}_3 - \boldsymbol{\phi}_4,\\ \mathbf{x}_{k+1} = \mathbf{z}_k - 5\boldsymbol{\phi}_5 + 10\boldsymbol{\phi}_6 - 10\boldsymbol{\phi}_7 + 5\boldsymbol{\phi}_8 - \boldsymbol{\phi}_9,\end{cases} \tag{7.37}$$

where

$$\mathbf{e}_{k+1} = \mathbf{A}_2\mathbf{d}_3\left(\mathbf{A}_2\mathbf{d}_2\left(\mathbf{A}_2\mathbf{d}_1\right)^2\right)^2\,\mathbf{e}_k{}^8 + \mathcal{O}\big(\,\mathbf{e}_k{}^9\big). \tag{7.38}$$

7.3 Methods with memory

Now a question may arise that is it possible to accelerate the convergence rate of the family of methods (7.14) without incorporating the evaluation of new functions or divided difference operators of various orders? To respond such a thing, a notion which is first discussed formally in [1] can be pursued which is known as with memorization.

During the process of with memorization of an iteration scheme, one may find appropriate approximations for the parameters using the current and previous saved data of the iterative steps so as to speed up the convergence rate without any special new evaluations. This could increase the computational efficiency of the methods in most cases. For getting a background and a recent theory in this category, one may consult [46] and the references cited therein.

In this section, we present several new derivative-free methods with memory with increased r-orders of convergence based on the iterative family (7.14) without memory. According to the obtained error equations of the previous section, one way is to approximate the parameters b_0, b_1 and b_2 using the available data. That is to say to accelerate the whole process, e.g., one may solve the matrix problem

$$\mathbf{I} + \mathbf{b}\mathbf{F}'(\boldsymbol{\alpha}) = 0. \tag{7.39}$$

Due to that fact that the value of $\boldsymbol{\alpha}$ is not known, we can use an approximation of $\mathbf{F}'(\boldsymbol{\alpha})$, calculated by available information to accelerate the convergence rate to a certain extent. Therefore, we may write $\mathbf{b} \simeq -\mathbf{F}'(\bar{\boldsymbol{\alpha}})^{-1}$, wherein $\bar{\boldsymbol{\alpha}}$ is an approximation of the solution (per cycle).

Remark 1. When developing iteration schemes with memory for nonlinear systems of equations, the main notion consists in calculating the parameter matrix $\mathbf{b} := \mathbf{B}_k (k \geq 1)$ as the iterative method proceeds by using some approximations to $-\mathbf{F}'(\boldsymbol{\alpha})$ evaluated by available data.

Now, if we approximate

$$\mathbf{b}_0 = \mathbf{b}_1 = \mathbf{b}_2 := \mathbf{B}_k = -[\mathbf{w}_{k-1}, \mathbf{x}_{k-1}; \mathbf{F}]^{-1} \approx -\mathbf{F}'(\boldsymbol{\alpha})^{-1}, \tag{7.40}$$

we can achieve a convergence r-order higher than eight. Therefore, our first iterative method with memory is defined by

$$\begin{cases} \mathbf{B}_{0k} = \mathbf{B}_{1k} = \mathbf{B}_{2k} = -[\mathbf{w}_{k-1}, \mathbf{x}_{k-1}; \mathbf{F}]^{-1} = -(\mathbf{M}_{k-1})^{-1}, \qquad k \geq 1, \\ \mathbf{y}_k = \mathbf{x}_k - \boldsymbol{\phi}_1, \qquad k \geq 0, \\ \mathbf{z}_k = \mathbf{y}_k - 3\boldsymbol{\phi}_2 + 3\boldsymbol{\phi}_3 - \boldsymbol{\phi}_4, \\ \mathbf{x}_{k+1} = \mathbf{z}_k - 4\boldsymbol{\phi}_5 + 6\boldsymbol{\phi}_6 - 4\boldsymbol{\phi}_7 + \boldsymbol{\phi}_8, \end{cases} \tag{7.41}$$

wherein $\mathbf{w}_k = \mathbf{x}_k + \mathbf{B}_{0k}\mathbf{F}(\mathbf{x}_k)$, $\mathbf{h}_k = \mathbf{y}_k + \mathbf{B}_{1k}\mathbf{F}(\mathbf{y}_k)$, and $\mathbf{l}_k = \mathbf{z}_k + \mathbf{B}_{2k}\,\mathbf{F}(\mathbf{z}_k)$. Before going to the main theorem, it is requisite to provide the following lemma so as to ease up the recognition of the numerical analysis of with memorization for the new methods.

Lemma 1. Let us define the same conditions as in Theorem 1. Moreover, if we define $\mathbf{B}_k = -[\mathbf{w}_{k-1},\ \mathbf{x}_{k-1};\ \mathbf{F}]\ ^{-1}$ and $\mathbf{d}^{(k)} := \mathbf{I} + \mathbf{B}_k\,\mathbf{F}'(\boldsymbol{\alpha})$, then we obtain the following asymptotic error relation

$$\mathbf{d}^{(k)} \sim\ \mathbf{e}_{k-1}. \tag{7.42}$$

Proof. To prove this, we need to apply the Taylor expansion again. The expansion of $[\mathbf{w}_{k-1}, \mathbf{x}_{k-1}; \mathbf{F}]^{-1}$ around the simple zero can be written as

$$\begin{aligned}\mathbf{B}_k &= -[\mathbf{w}_{k-1}, \mathbf{x}_{k-1}; \mathbf{F}]^{-1} = -\big(\mathbf{I} - \mathbf{A}_2\big(\mathbf{I} + \mathbf{d}^{(k-1)}\big)\mathbf{e}_{k-1} + \mathcal{O}\big(\mathbf{e}_{k-1}^2\big)\big)\mathbf{F}'(\boldsymbol{\alpha})^{-1},\\ &= -\big(\mathbf{I} - \mathbf{A}_2(2\mathbf{I} + \mathbf{B}_{k-1}\mathbf{F}'(\boldsymbol{\alpha}))\,\mathbf{e}_{k-1} + \mathcal{O}\big(\mathbf{e}_{k-1}^2\big)\big)\mathbf{F}'(\boldsymbol{\alpha})^{-1}.\end{aligned} \tag{7.43}$$

Now by simplifying, one may get that

$$\begin{aligned}\mathbf{d}^{(k)} =&\mathbf{I} + \mathbf{B}_k\,\mathbf{F}'(\boldsymbol{\alpha}) = \mathbf{A}_2\big(2\mathbf{I} + \mathbf{B}^{(k-1)}\mathbf{F}'(\boldsymbol{\alpha})\big)\,\mathbf{e}_{k-1} + \mathcal{O}\big(\mathbf{e}_{k-1}^2\big),\\ &\sim\ \mathbf{e}_{k-1}\,.\end{aligned} \tag{7.44}$$

We also use the symbol $\sim$ so as to show the errors asymptotically without any unnecessary coefficients. The proof is complete. □

At this moment, we address the convergence r-order of (7.41) in what follows.

Theorem 2. Consider the same conditions as in Theorem 1 as well as the initial matrix $\mathbf{B}_0^{(0)}$, which is close enough to $\mathbf{F}'(\boldsymbol{\alpha})$. Then, the sequence $\{\mathbf{x}_k\}_{k\geq 0}$ obtained using the iterative expression (7.41) converges to $\boldsymbol{\alpha}$ with at least $4+\sqrt{19} \simeq 8.3589$ r-order of convergence.

Proof. Let $\{\mathbf{x}_k\}$ be a sequence generated by iterative method (7.41). It converges to simple root $\boldsymbol{\alpha}$ of $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ with $r-$order r. Now, we can write the error relation as

$$\mathbf{e}_{k+1} \sim D_{k,r}\ \mathbf{e}_k^r, \tag{7.45}$$

where $\lim_{k\to\infty} D_{k,r} = D_r$ and D_r is asymptotic error constant of (7.41). By using (7.45), we can write

$$\mathbf{e}_{k+1} \sim D_{k,r}\left(D_{k-1,r}\left(\mathbf{e}^{(k-1)^r}\right)\right)^r = D_{k,r}\, D^r_{k-1,r}\, \mathbf{e}^{r^2}_{k-1} \sim \mathbf{e}^{r^2}_{k-1}. \tag{7.46}$$

By using Lemma 1, (7.46) for (7.41) can be written as

$$\mathbf{e}_{k+1} \sim \mathbf{e}^3_{k-1}\, \mathbf{e}^{8r}_{k-1}. \tag{7.47}$$

By comparing (7.47) and (7.46), we get the equation

$$\begin{cases} r^2 - 8\,r - 3 = 0, \\ r = 4 + \sqrt{19} = 8.358898944\,. \end{cases} \tag{7.48}$$

Therefore, we conclude that the r-order for (7.41) is at least $4 + \sqrt{19}$. □

Remark 2. It is requisite to remind that there is no need for any attempt in order to compute $[\mathbf{w}_{k-1}, \mathbf{x}_{k-1}; \mathbf{F}]$ per cycle, since it has already be calculated at the end of the first sub-step and can directly be used as the acceleration matrix for approximating $\mathbf{B}_0$, $\mathbf{B}_1$, $\mathbf{B}_2$ in the subsequent cycle.

Similarly, we can now present a much more efficient method with memory in contrast to (7.41) by using (7.37) as follows:

$$\begin{cases} \mathbf{B}_{0k} = \mathbf{B}_{1k} = \mathbf{B}_{2k} = -[\mathbf{w}_{k-1}, \mathbf{x}_{k-1}; \mathbf{F}]^{-1} = -(\mathbf{M}_{k-1})^{-1}, \qquad k \geq 1, \\ \mathbf{y}_k = \mathbf{x}_k - \boldsymbol{\phi}_1, \qquad k \geq 0, \\ \mathbf{z}_k = \mathbf{y}_k - 3\boldsymbol{\phi}_2 + 3\boldsymbol{\phi}_3 - \boldsymbol{\phi}_4, \\ \mathbf{x}_{k+1} = \mathbf{z}_k - 5\boldsymbol{\phi}_5 + 10\boldsymbol{\phi}_6 - 10\boldsymbol{\phi}_7 + 5\boldsymbol{\phi}_8 - \boldsymbol{\phi}_9. \end{cases} \tag{7.49}$$

Theorem 3. Consider the same conditions as in Theorem 2. Then, the sequence $\{\mathbf{x}_k\}_{k\geq 0}$ obtained using the iterative expression (7.49) converges to $\boldsymbol{\alpha}$ with at least $4 + \sqrt{23} \simeq 8.79583$ r-order.

Proof. We assume that $\{\mathbf{x}_k\}$ be a sequence generated by iterative method (7.49). We can write the error relation as

$$\mathbf{e}_{k+1} \sim D_{k,r}\, \mathbf{e}^r_k, \tag{7.50}$$

where $\lim_{k\to\infty} D_{k,r} = D_r$ and D_r is asymptotic error constant of (7.49). By using (7.51), we can write

$$\mathbf{e}_{k+1} \sim D_{k,r}\left(D_{k-1,r}\left(\mathbf{e}_{k-1}{}^{r}\right)\right)^{r} = D_{k,r}\, D_{k-1,r}^{r}\, \mathbf{e}_{k-1}^{r^2} \sim \mathbf{e}_{k-1}^{r^2}. \tag{7.51}$$

By using Lemma 1, (7.52) for (7.49) can be written as

$$\mathbf{e}_{k+1} \sim \mathbf{e}_{k-1}^{7}\, \mathbf{e}_{k-1}^{8r}, \tag{7.52}$$

By comparing (7.52) and (7.51), we get the equation

$$\begin{cases} r^2 - 8\,r - 7 = 0, \\ r = 4 + \sqrt{23} = 8.795831523\,. \end{cases} \tag{7.53}$$

Thus, we conclude that the r-order for (7.49) is $4 + \sqrt{23}$. This ends the proof. □

Now, it is eminent to study that how we accelerate the convergence rate more? It is of crystal clear nature that a better approximation for the parameter matrices may accelerate the schemes with memory more. However, we have an obstacle in front, which is the definition of the divided difference operator for three points in the multidimensional case. Such a definition would be too much costly and could not be of practical interest.

Hence, a better approach is to use the already computed data more carefully so as to obtain better approximation matrices. Accordingly, we could write

$$\mathbf{B}_{0k} = -(\mathbf{M}_{k-1})^{-1}, \mathbf{B}_{2k} = -(\mathbf{N}_{k-1})^{-1}, \mathbf{B}_{3k} = -(\mathbf{Q}_{k-1})^{-1}, \tag{7.54}$$

whereas $\mathbf{M}_{k-1}, \mathbf{N}_{k-1}, \mathbf{Q}_{k-1}$ have already been computed.

However, in this way, we are still using an old approximation such as $-(\mathbf{M}_{k-1})^{-1}$ for $\mathbf{B}_{0k}$! This means that, $-(\mathbf{Q}_{k-1})^{-1}$ can be a better approximation for all parameter matrices. As a result, we may present a method with memory for nonlinear systems with the following structure

$$\begin{cases} \mathbf{B}_{0k} = \mathbf{B}_{1k} = \mathbf{B}_{2k} = -(\mathbf{Q}_{k-1})^{-1}, \qquad k \geq 1, \\ \mathbf{y}_k = \mathbf{x}_k - \boldsymbol{\phi}_1, \qquad k \geq 0, \\ \mathbf{z}_k = \mathbf{y}_k - 3\boldsymbol{\phi}_2 + 3\boldsymbol{\phi}_3 - \boldsymbol{\phi}_4, \\ \mathbf{x}_{k+1} = \mathbf{z}_k - 5\boldsymbol{\phi}_5 + 10\boldsymbol{\phi}_6 - 10\boldsymbol{\phi}_7 + 5\boldsymbol{\phi}_8 - \boldsymbol{\phi}_9. \end{cases} \tag{7.55}$$

Although the scheme (7.55) has better r-order than (7.41) and (7.49), it suffers of computing a new matrix inversion which could restrict the application (7.55) in large dimensions. To obtain higher r-order with low computational cost, i.e., only one LU decomposition in each cycle, we apply a new trick and propose the following

$$\begin{cases} \mathbf{x}^{(0)} \text{ and } \mathbf{B}_0^{(0)} \textit{ are given}, \\ \mathbf{y}_k = \mathbf{x}_k - \boldsymbol{\phi}_1, \qquad k \geq 0, \\ \mathbf{B}_0^{(k+1)} = \mathbf{B}_{1k} = \mathbf{B}_{2k} = -(\mathbf{M}^{(k)})^{-1}, \qquad k \geq 0, \\ \mathbf{z}_k = \mathbf{y}_k - 3\boldsymbol{\phi}_2 + 3\boldsymbol{\phi}_3 - \boldsymbol{\phi}_4, \\ \mathbf{x}_{k+1} = \mathbf{z}_k - 5\boldsymbol{\phi}_5 + 10\boldsymbol{\phi}_6 - 10\boldsymbol{\phi}_7 + 5\boldsymbol{\phi}_8 - \boldsymbol{\phi}_9. \end{cases} \tag{7.56}$$

Theorem 4. Consider the same conditions as in Theorem 3. Then, the sequence $\{\mathbf{x}_k\}_{k\geq 0}$ obtained using the iterative expression (7.56) converges to $\boldsymbol{\alpha}$ with at least $\frac{1}{2}(9+5\sqrt{5}) \simeq 10.0902$ r-order.

Proof. Let us first re-write (7.38) in the asymptotical form as follows:

$$\mathbf{e}_{k+1} \sim \mathbf{d}_3^{(k)} \mathbf{d}_1^{(k)^4} \mathbf{d}_2^{(k)^2} \mathbf{e}_k^8. \tag{7.57}$$

Following a same terminology as in the proof of Theorems 2 and 3, we have

$$\mathbf{d}_1^{(k)} \sim \mathbf{e}_{k-1}, \qquad \mathbf{d}_2^{(k)} \sim \mathbf{e}_{k-1}, \qquad \mathbf{d}_3^{(k)} \sim \mathbf{e}_{k-1}, \qquad \forall k \geq 1. \tag{7.58}$$

But here in (7.56), $\mathbf{B}_{1k}$ and $\mathbf{B}_{2k}$ are updated for all $k \geq 0$. Therefore, we obtain

$$\mathbf{d}_1^{(k)^4} \sim \mathbf{e}_{k-1}^4 \mathbf{I}, \qquad \mathbf{d}_2^{(k)^2} \sim \mathbf{e}_{k-1}^2 \mathbf{e}_{k-1}^4, \qquad \mathbf{d}_3^{(k)} \sim \mathbf{e}_{k-1} \mathbf{e}_{k-1}^8, \qquad \forall k \geq 0. \tag{7.59}$$

Combining (7.58) and (7.59) into (7.57), we attain

$$\mathbf{e}_{k+1} \sim \mathbf{e}_{k-1}^{19}\, \mathbf{e}_k{}^8 \sim \mathbf{e}_{k-1}^{11} \mathbf{e}_k^9. \tag{7.60}$$

Iterative methods	ρ	FE	$\mathcal{C}$	$\mathcal{E}$
NM	2	$n+n^2$	$(n+n^2)+u((2/3)n^3+2n^2)$	$2^{\frac{1}{\frac{2n^3u}{3}+n^2(2u+1)+n}}$
ST	2	$2n+n^2-n$	$(2n+n^2-n)+u((2/3)n^3+2n^2)$	$2^{\frac{1}{\frac{2n^3u}{3}+n^2(2u+1)+n}}$
SH	4	$4n+2(n^2-n)$	$(4n+2(n^2-n))+u((2/3)n^3+4(2n^2))$	$2^{\frac{3}{n\left(n^2u+3n(4u+1)+3\right)}}$
SHM	$2+\sqrt{6}$	$4n+2(n^2-n)$	$(4n+2(n^2-n))+u((2/3)n^3+4(2n^2))$	$\left(2+\sqrt{6}\right)^{\frac{3}{2n\left(n^2u+3n(4u+1)+3\right)}}$
PM1	8	$4n+3(n^2-n)$	$(6n+3(n^2-n))+u((2/3)n^3+8(2n^2))$	$8^{\frac{3}{n\left(2n^2u+n(48u+9)+9\right)}}$
PM2	8	$6n+3(n^2-n)$	$(6n+3(n^2-n))+u((2/3)n^3+9(2n^2))$	$8^{\frac{3}{n\left(2n^2u+9n(6u+1)+9\right)}}$
PM3	8.3589	$4n+3(n^2-n)$	$(6n+3(n^2-n))+u((2/3)n^3+8(2n^2))$	$8.3589^{\frac{3}{n\left(2n^2u+n(48u+9)+9\right)}}$
PM4	8.79583	$6n+3(n^2-n)$	$(6n+3(n^2-n))+u((2/3)n^3+9(2n^2))$	$8.79583^{\frac{3}{n\left(2n^2u+9n(6u+1)+9\right)}}$
PM5	10.0902	$6n+3(n^2-n)$	$(6n+3(n^2-n))+u((2/3)n^3+9(2n^2))$	$10.0902^{\frac{3}{n\left(2n^2u+9n(6u+1)+9\right)}}$

TABLE 7.1: Comparison of efficiency indices for different methods.

This implies that $\frac{11}{p}+9=p$, where the r-order of convergence would then be $p=\frac{1}{2}(9+5\sqrt{5})\simeq 10.0902$. This completes the proof. □

Finally, we point out that the easy structure and high r-order of (7.56) by using only one LU decomposition, makes it not only interesting from a theoretical point of view but is also in practice.

7.4 Numerical analysis

Efficiency is generally a crucial aspect to pay heed of when choosing an iterative process to calculate a solution of nonlinear algebraic systems.

The classic efficiency index $E=\rho^{\frac{1}{\theta}}$ [1], provides a balance between the order of convergence ρ and the number of functional evaluations θ. However, it is interesting to note that when we deal with a system of n nonlinear equations, the computational cost of evaluating the operators $\mathbf{F}$, $\mathbf{F}'$ or DDO are not similar, as it happens in the case of scalar equations.

Typically, the costs are as follows:

- For one evaluation of $\mathbf{F}$, n functional evaluations are required.
- The evaluation of the associated Jacobian matrix $\mathbf{F}'$ requires n^2 functional evaluations.
- One evaluation of the first-order DDO requires n^2-n functional evaluations.

- Moreover, we consider that the cost of LU decomposition is $u(2\frac{n^3}{3})$ plus $u(2n^2)$ for solving two triangular systems,

where u is a weight which relates the cost of one functional evaluation and one flops.

Note that another way is to estimate the efficiency of the methods by using the index [43, 51]:

$$E = \frac{1}{\log \tau} \frac{\log \rho}{\theta}, \tag{7.61}$$

where ρ is the order of convergence, θ is the computational cost per iteration and τ is the number of significant decimal digits of the approximation $\mathbf{x}_k$.

Let us denote the methods (7.1), (7.9), (7.12), (7.13), (7.36), (7.37), (7.41), (7.49) and (7.56) by NM, ST, SH, SHM, PM1, PM2, PM3, PM4 and PM5, respectively. Here, the computational flops-like efficiency index is computed [102]

$$\mathcal{E} = \rho^{\frac{1}{C}}, \tag{7.62}$$

where ρ is the r-order of convergence and C stands for the *total* computational cost per iteration in terms of the number of functional evaluations and the costs needed for LU decompositions, matrix multiplications, number of triangular systems, etc. See for more [56].

We tried to compare different methods in a similar environment in this work. In fact, we considered that the cost of scalar function evaluation is nearly five times more than the cost of doing typical flops ($u = 1/5$). Note that it is only an assumption and in general the relation between the cost of scalar function evaluation and the cost of doing operations are related to the specifications of the computer. Furthermore, the cost of matrix to vector products has been ignored for simplicity.

In general, studying the computational complexity of a nonlinear system might be a challenging problem, if one uses some other strategies in the process of doing a cycle such as applying GMRES, BiCGSTAB, etc.

In the iterative method PM1, one LU decomposition is needed. In fact, we solve eight linear systems, but due to the fact that the coefficient matrices are same, only one LU decomposition could be done in the programs. There is a similar procedure for PM2-PM5.

The numbers of functional evaluations to obtain the theoretical efficiency index for different methods are illustrated in Table 7.1. Thus, the results of comparison for different values of n are given in Figures 7.1-7.2, which show the efficient behavior for the proposed methods with memory, particularly PM5.

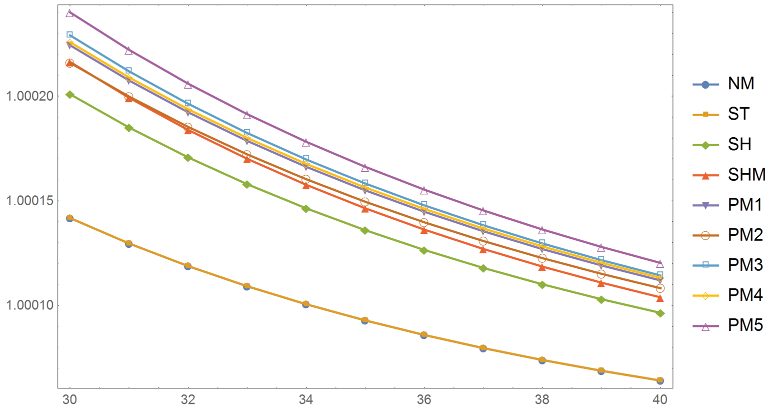


Figure 7.1: The plot of the efficiency indices for different methods in the case $n = 30, \ldots, 40$.

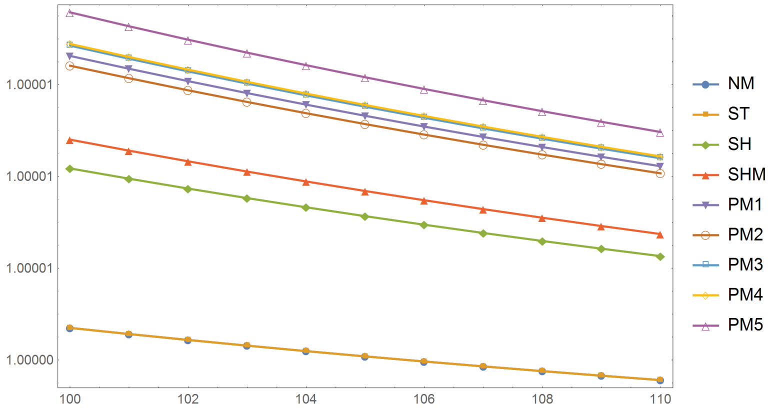


Figure 7.2: The plot of the efficiency indices for different methods in the case $n = 100, \ldots, 110$.

7.5 Simulations to illustrate the analytical results

In this section, we test the contributed methods using Mathematica 10.0 [58]. For the first following experiment which has some academical nature, high precision computing using 600 fixed point arithmetics is considered to check the high rates of convergence and to calculate the computational r-order of convergence, which is defined by [39, 48]

$$\text{CCO} \approx \frac{\ln(||\mathbf{F}(\mathbf{x}_{k+1})||_2/||\mathbf{F}(\mathbf{x}_k)||_2)}{\ln(||\mathbf{F}(\mathbf{x}_k)||_2/||\mathbf{F}(\mathbf{x}_{k-1})||_2)}. \tag{7.63}$$

In other application problems, double precision arithmetic has been used which is quite enough for large systems resulting from the discretization of differential equations. The computer specifications are: Intel(R) Core(TM) i5-2430M CPU 2.40 GHz with 8.00 GB of RAM on Windows 7 Ultimate. We have chosen the tolerance $\|\mathbf{F}(\mathbf{x}_{k+1})\|_2 < \epsilon$ for implementing the methods, where ϵ is the tolerance.

Example 1. We begin with the system $\mathbf{F}(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}) = 0$ defined by:

$$\mathbf{F}(\mathbf{x}) = \begin{cases} 5\exp{(x_1-2)}x_2 + 2{x_7}^{x_{10}} + 8{x_3}^{x_4} - 5{x_6}^3 - x_9, \\ 5\tan{(x_1+2)} + \cos{({x_9}^{x_{10}})} + {x_2}^3 + 7{x_3}^4 - 2\sin^3{(x_6)}, \\ {x_1}^2 - x_{10}x_5x_6x_7x_8x_9 + \tan{(x_2)} + 2{x_3}^{x_4} - 5{x_6}^3, \\ 2\tan{({x_1}^2)} + 2^{x_2} + {x_3}^2 - 5{x_5}^3 - x_6 + {x_8}^{\cos{(x_9)}}, \\ 10{x_1}^2 - x_{10} + \cos{(x_2)} + {x_3}^2 - 5{x_6}^3 - 2x_8 - 4^{x_9}, \\ \cos^{-1}({x_1}^2)\sin{(x_2)} - 2x_{10}{x_5}^4x_6x_9 + {x_3}^2, \\ x_1{x_2}^{x_7} - {x_8}^{x_{10}} + {x_3}^5 - 5{x_5}^3 + x_7, \\ \cos^{-1}{(-10x_{10} + x_8 + x_9)} + x_4\sin{(x_2)} + x_3 - 15{x_5}^2 + x_7, \\ 10x_1 + {x_3}^2 - 5{x_5}^2 + 10{x_6}^{x_8} - \sin{(x_7)} + 2x_9, \\ x_1\sin{(x_2)} - 2{x_{10}}^{x_8} + x_{10} - 5x_6 - 10x_9, \end{cases} \tag{7.64}$$

where $\boldsymbol{\alpha} \simeq$ (1.3273490437+0.3502924960i, 1.058599346-1.748724664i,1.0276186794−0.0141308051i,3.27395 0008+0.127828308i, 0.8318243937+0.0017551949i,−0.4853245912+0.6848776400i,0.1693667630+0.184091 7580i,1. 5344 19958-0.321214766i, 2.086379651+0.426342755i, $-1.989592331 + 1.478395393i)^*$, and $(\mathbf{x}^{(0)} = (1.4 + 0.5I, 1.1 - 2.0I, 1.0 - 0.2I, 2.5 + 0.5I, 0.8 - 0.1I, -0.4 + 1.I, 0.1 + 0.1I, 1.4 - 0.6I, 2.0 + 0.5I, -2.0 + 1.45I)^*$.

The results of comparisons for this example are brought forward in Table 7.2. Here, IT, $R^{(k+1)}$, CCO and Time stand for the number of iterations, the residual norm $\|R^{(k+1)}\|_2 = \|\mathbf{F}(\mathbf{x}_{k+1})\|_2$, (7.63), and the elapsed computational time in

Iterative methods	IT	$R^{(k+1)}$	CCO	Time
ST	12	1.241×10^{-471}	2.000	1.542088
SH	6	1.477×10^{-349}	4.001	1.610092
SHM	6	4.300×10^{-498}	4.448	1.667095
PM1	4	7.251×10^{-209}	8.007	1.601092
PM2	4	6.581×10^{-260}	8.028	1.615092
PM3	4	6.695×10^{-298}	8.372	1.603092
PM4	4	2.249×10^{-431}	8.844	1.646094
PM5	4	1.561×10^{-502}	10.11	1.642094

Table 7.2: Results of comparisons for different methods in Example 1.

seconds to run the iterative method, respectively. Note that the criterion (7.63) is independent of the knowledge of the root.

Throughout this chapter, the following diagonal matrix of the appropriate size

$$B^{(0)^{-1}} = \begin{pmatrix} -\frac{1}{100} & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & -\frac{1}{100} \end{pmatrix}, \tag{7.65}$$

has been used for starting the parameter matrices whenever required.

The results for another initial approximation ($\mathbf{x}^{(0)} = (1.3+0.6I, 1.1-2.0I,\ 1.1-0.1I, 2.5+0.5I,\ 0.85-0.25I, -0.5+1.1I,\ 0.1+0.1I, 1.4-0.6I,\ 2.0+0.4I, -2.0+1.45I)^*$ have also been provided in Table 7.3.

From the results presented in Tables 7.2-7.3, we observe that the accuracy of approximations to the solution increases as the iteration process proceeds, showing stable character of the methods. The new methods with memory show robust character in terms of the accuracy when compared with the other methods.

Note that a line search could be done in solving nonlinear systems of equations to let the initial guesses arrive at trust region. Robust strategies for providing enough accurate initial guesses have been discussed in [44] and [55]. We also remind that in case of facing with singular matrices if a badly chosen initial approximation be chosen, then one might use the generalized outer inverses (see e.g. [54]) instead of the regular inverse.

Basically, the process of solving a nonlinear partial differential equation (PDE) would be reduced to solving a set of nonlinear algebraic equations. Another point

Iterative methods	IT	$R^{(k+1)}$	CCO	Time
ST	12	1.760×10^{-370}	2.005	1.530088
SH	6	1.014×10^{-251}	4.002	1.627093
SHM	6	2.463×10^{-440}	4.461	1.666095
PM1	4	9.232×10^{-161}	7.993	1.593091
PM2	4	7.496×10^{-209}	7.991	1.604092
PM3	4	8.155×10^{-217}	8.368	1.633093
PM4	4	9.863×10^{-317}	8.794	1.625093
PM5	4	2.744×10^{-464}	10.01	1.615092

Table 7.3: Results of comparisons for different methods in Example 1 with the second initial approximation.

is that for such cases, basically numerical results of lower accuracy in terms of the precision are needed.

Example 2. *The Nonlinear Coupled Burgers Equations [5].* In this experiment, we consider the two-dimensional nonlinear coupled Burgers equations as follows:

$$\begin{aligned} \frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial(\mathbf{x}^2} - 2u\frac{\partial u}{\partial x} + \frac{\partial u}{\partial x}v = 0, \\ \frac{\partial v}{\partial t} - \frac{\partial^2 v}{\partial(\mathbf{x}^2} - 2v\frac{\partial v}{\partial x} + \frac{\partial u}{\partial x}v = 0, \end{aligned} \tag{7.66}$$

on the interval $x \in [0, 5]$, where $0 \leq t \leq T = 1/4$ and the initial conditions are

$$u(\mathbf{x}, 0) = \sin x, \qquad v(\mathbf{x}, 0) = \sin x. \tag{7.67}$$

Note that the boundary conditions are set to the exact solution $u(x, t) = \exp(-t)\sin(x)$ and $v(x, t) = \exp(-t)\sin(x)$.

These nonlinear coupled equations with parameters derivative contain many important mathematical physics equations and reaction diffusion equations. To solve this system of two PDEs by using finite difference (FD) discretization with the stopping criterion $||\mathbf{F}(\mathbf{x})||_2 < 10^{-8}$ and the interval $x \in [0, 5]$, we use the backward finite difference for the first derivative along the time (the independent variable t):

$$u_t(\mathbf{x}_i, t_j) \simeq \frac{w_{i,j} - w_{i,j-1}}{k}, \tag{7.68}$$

where k is the step size, and the central finite difference for the other involved pieces of the equations, i.e.,

$$u_x(\mathbf{x}_i, t_j) \simeq \frac{w_{i+1,j} - w_{i-1,j}}{2h}, \tag{7.69}$$

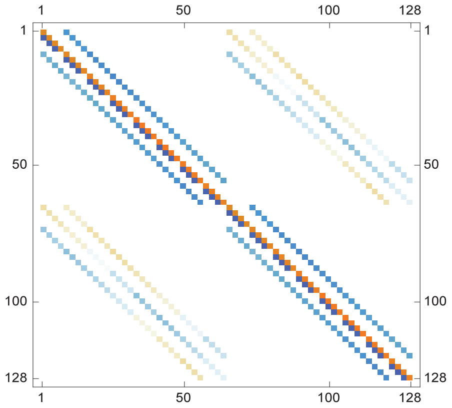


FIGURE 7.3: The matrix plot of the divided difference operator $[\mathbf{x}_k, \mathbf{w}_k; \mathbf{F}]$ in Example 2.

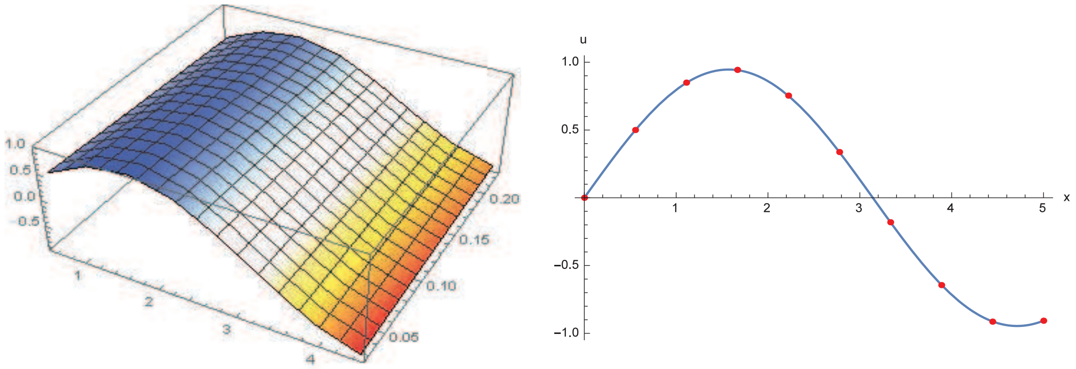


FIGURE 7.4: The curve of the numerical solution $v(\mathbf{x}, t)$ using FD and PM5 (left), and comparison of the exact $u(\mathbf{x}, t)$ (solid blue) and approximate (dotted red) solutions obtained via FD and PM5 (right).

and

$$u_{xx}(\mathbf{x}_i, t_j) \simeq \frac{w_{i+1,j} - 2w_{i,j} + w_{i-1,j}}{h^2}, \tag{7.70}$$

wherein h is the step size along the space. In solving (7.66), the procedure will end in a nonlinear system of algebraic equations having a large sparse form for the DDO matrix (illustrated in Figure 7.3). The solution has been plotted in Figure 7.4 using PM5 with one full iteration since its high order is enough to demonstrate the final accuracy. For this test we have chosen $M = N = 9$, to obtain a nonlinear system of the size 128 × 128, with the starting vector $\texttt{left} = 0.8; \texttt{right} = -0.8; \texttt{pts} = \texttt{Range}[\texttt{aa}, \texttt{bb}, (\texttt{bb} - \texttt{aa})/((\texttt{M} - 1) * (\texttt{N} - 1))];$ $\texttt{x0} = \texttt{Drop}[\texttt{Drop}[\texttt{Join}[\texttt{pts}, \texttt{pts}], 1], -1];$ in the Mathematica environment with the machine precision.

7.6 Summary

Iteration is a repeated application of a function, and can be viewed as a discrete dynamical system, in which the continuous time variable has been quantized to assume integer values. In fact, iterative methods are the only practical ways to solve nonlinear systems of equations.

Here, a method without the need of computing Fréchet derivative per cycle had been introduced which is without memory and of high order. The main goal of the chapter had also been provided by introducing new methods with memory of higher r-orders for nonlinear systems.

The term method with memory points to the use of data from the current and previous iterations. Iteration schemes with memory for solving systems of nonlinear equations have been considered very seldom in the literature and so the presented methods may be regarded as an advancement in the topic.

The numerical analysis of the schemes, i.e., their complexity and rates of convergence have been studied and showed a fast, efficient and stable behavior for the new methods with memory, particularly PM5.

The theoretical orders were verified by calculating the computational order of convergence. Numerical experiments in solving some applicable problems, such as partial differential equations have also been furnished and confirmed the theoretical discussions. Note that in real scenarios, derivative-free iterative methods are the only option even they are computationally expensive due to the filling process of the DDO matrix.

The attention for future works might be focused on providing better approximations of the parameter matrices using first-order DDO so as to increase the r-order of convergence without imposing further computational burden. Extension of such schemes with frozen DDOs have also not yet been discussed in deep. Such developments are now under investigation in our research group and they can be done as future works in this field of study.

Chapter 8

Are the eigenvalues of preconditioned banded symmetric Toeplitz matrices known in almost closed form?

Bogoya, Böttcher, Grudsky, and Maximenko have recently obtained the precise asymptotic expansion for the eigenvalues of a sequence of Toeplitz matrices $\{T_n(f)\}$, under suitable assumptions on the associated generating function f. We provide numerical evidence that some of these assumptions can be relaxed and extended to the case of a sequence of preconditioned Toeplitz matrices $\{T_n^{-1}(g)T_n(f)\}$, for f trigonometric polynomial, g nonnegative, not identically zero trigonometric polynomial, $r = f/g$, and where the ratio $r(\cdot)$ plays the same role as $f(\cdot)$ in the nonpreconditioned case. Moreover, based on the eigenvalue asymptotics, we devise an extrapolation algorithm for computing the eigenvalues of preconditioned banded symmetric Toeplitz matrices with a high level of accuracy, with a relatively low computational cost, and with potential application to the computation of the spectrum of differential operators.

8.1 Introduction

A matrix of size n, having a fixed entry along each diagonal, is called Toeplitz and enjoys the expression

$$[a_{i-j}]_{i,j=1}^{n} = \begin{bmatrix} a_0 & a_{-1} & a_{-2} & \cdots & \cdots & a_{-(n-1)} \\ a_1 & \ddots & \ddots & \ddots & & \vdots \\ a_2 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & a_{-2} \\ \vdots & & \ddots & \ddots & \ddots & a_{-1} \\ a_{n-1} & \cdots & \cdots & a_2 & a_1 & a_0 \end{bmatrix}.$$

Given a complex-valued Lebesgue integrable function $\phi : [-\pi, \pi] \to \mathbb{C}$, the n-th Toeplitz matrix generated by ϕ is defined as

$$T_n(\phi) = \left[\hat{\phi}_{i-j}\right]_{i,j=1}^{n},$$

where the quantities $\hat{\phi}_k$ are the Fourier coefficients of ϕ, which means

$$\hat{\phi}_k = \frac{1}{2\pi}\int_{-\pi}^{\pi} \phi(\theta)\, \mathrm{e}^{-\mathbf{i}k\theta}\mathrm{d}\theta, \qquad k \in \mathbb{Z}.$$

We refer to $\{T_n(\phi)\}_n$ as the Toeplitz sequence generated by ϕ, which in turn is called the generating function of $\{T_n(\phi)\}_n$. In the case where ϕ is real-valued, all the matrices $T_n(\phi)$ are Hermitian and much is known about their spectral properties, from the localization of the eigenvalues to the asymptotic spectral distribution in the Weyl sense: in particular ϕ is the spectral symbol of $\{T_n(\phi)\}_n$, see [9, 11] and the references therein.

More in detail, if ϕ is real-valued and not identically constant, then any eigenvalue of $T_n(\phi)$ belongs to the open set (m_ϕ, M_ϕ), with m_ϕ, M_ϕ being the essential infimum, the essential supremum of ϕ, respectively. The case of a constant ϕ is trivial: in that case if $\phi = m$ almost everywhere then $T_n(\phi) = m\mathbb{I}_n$ with $\mathbb{I}_n$ denoting the identity of size n. Hence if $M_\phi > 0$ and ϕ is nonnegative almost everywhere, then $T_n(\phi)$ is Hermitian positive definite.

We focus our attention on the following setting.

- We consider two real-valued cosine trigonometric polynomials (RCTPs) f, g, that is
$$f(\theta) = \hat{f}_0 + 2\sum_{k=1}^{m_1} \hat{f}_k \cos(k\theta), \qquad \hat{f}_0, \hat{f}_1, \ldots, \hat{f}_{m_1} \in \mathbb{R}, \qquad m_1 \in \mathbb{N},$$
$$g(\theta) = \hat{g}_0 + 2\sum_{k=1}^{m_2} \hat{g}_k \cos(k\theta), \qquad \hat{g}_0, \hat{g}_1, \ldots, \hat{g}_{m_2} \in \mathbb{R}, \qquad m_2 \in \mathbb{N},$$
so that $T_n(f)$, $T_n(g)$ are both real symmetric.
- We assume that $M_g = \max g > 0$ and $m_g = \min g \geq 0$, so that $T_n(g)$ is positive definite.
- We consider $\mathcal{P}_n(f, g) = T_n^{-1}(g)T_n(f)$ the "preconditioned" matrix and we define the new symbol $r = f/g$.

The n-th Toeplitz matrix generated by $\phi \in \{f, g\}$ is the real symmetric banded matrix of bandwidth $2m+1$, $m \in \{m_1, m_2\}$ ($m = m_1$ if $\phi = f$ and $m = m_2$ if $\phi = g$), given by

$$T_n(\phi) = \left[\begin{array}{ccccccccccc}
\hat{\phi}_0 & \hat{\phi}_1 & \cdots & \hat{\phi}_m & & & & & & & \\
\hat{\phi}_1 & \ddots & \ddots & & \ddots & & & & & & \\
\vdots & \ddots & \ddots & \ddots & & \ddots & & & & & \\
\hat{\phi}_m & & \ddots & \ddots & \ddots & & \ddots & & & & \\
 & \ddots & & \ddots & \ddots & \ddots & & \ddots & & & \\
 & & \multicolumn{7}{c}{\boxed{\hat{\phi}_m \quad \cdots \quad \hat{\phi}_1 \quad \hat{\phi}_0 \quad \hat{\phi}_1 \quad \cdots \quad \hat{\phi}_m}} & & \\
 & & & \ddots & & \ddots & \ddots & \ddots & & \ddots & \\
 & & & & \ddots & & \ddots & \ddots & \ddots & & \hat{\phi}_m \\
 & & & & & \ddots & & \ddots & \ddots & \ddots & \vdots \\
 & & & & & & \ddots & & \ddots & \ddots & \hat{\phi}_1 \\
 & & & & & & & \hat{\phi}_m & \cdots & \hat{\phi}_1 & \hat{\phi}_0
\end{array}\right].$$

Matrices of the form $\mathcal{P}_n(f, g)$ are important for the fast solution of large Toeplitz linear systems (in connection with the preconditioned conjugate gradient method [134–136, 142] or of more general preconditioned Krylov methods [139, 140]). Furthermore, up to low rank corrections, they appear in the context of the spectral approximation of differential operators in which a low rank correction of $T_n(g)$ is the mass matrix and a low rank correction of $T_n(f)$ is the stiffness matrix.

Their spectral features have been studied in detail. More precisely, under the assumption that $r = m$ identically $\mathcal{P}_n(f, g) = r\mathbb{I}_n$, while if $m_r < M_r$, then any eigenvalue of $\mathcal{P}_n(f, g)$ belongs to the open set (m_r, M_r), see [136], and the whole sequence $\{\mathcal{P}_n(f, g)\}_n$ is spectrally distributed in the Weyl sense as $r = f/g$ (see [143]).

In our context, we say that a function is monotone if it is either increasing or decreasing over the interval $[0, \pi]$.

Under the assumption that $r = f/g$ is monotone, We show experimentally that for every integer $\alpha \geq 0$, every n and every $j = 1, \ldots, n$, the following asymptotic expansion holds:

$$\lambda_j(\mathcal{P}_n(f, g)) = r(\theta_{j,n}) + \sum_{k=1}^{\alpha} c_k(\theta_{j,n})h^k + E_{j,n,\alpha}, \tag{8.1}$$

where:

- the eigenvalues of $\mathcal{P}_n(f, g)$ are arranged in nondecreasing or nonincreasing order, depending on whether r is increasing or decreasing;
- $\{c_k\}_{k=1,2,\ldots}$ is a sequence of functions from $[0, \pi]$ to $\mathbb{R}$ which depends only on r;
- $h = \frac{1}{n+1}$ and $\theta_{j,n} = \frac{j\pi}{n+1} = j\pi h$;
- $E_{j,n,\alpha} = O(h^{\alpha+1})$ is the remainder (the error), which satisfies the inequality $|E_{j,n,\alpha}| \leq C_\alpha h^{\alpha+1}$ for some constant C_α depending only on α and r.

In the pure Toeplitz case, that is for $g = 1$ identically, so that $\mathcal{P}_n(f, g) = T_n(f)$ and $r = f$, the result is proven in [131–133], if the RCTP f is monotone and satisfies certain additional assumptions, which include the requirements that $f'(\theta) \neq 0$ for $\theta \in (0, \pi)$ and $f''(\theta) \neq 0$ for $\theta \in \{0, \pi\}$. The symbols

$$f_q(\theta) = (2 - 2\cos\theta)^q, \qquad q = 1, 2, \ldots, \tag{8.2}$$

arise in the discretization of differential equations and are therefore of particular interest. Unfortunately, for these symbols the requirement that $f''(0) \neq 0$ is not satisfied if $q \geq 2$. In [138] several numerical evidences are reported, showing that the higher order approximation (8.1) holds even in this "degenerate case".

Here, as first purpose, we show numerically the same for the preconditioned matrices $\mathcal{P}_n(f,g)$ and, from a theoretical point of view, the numerical testing is complemented by the proof of the above conjecture in the basic case of $\alpha = 0$.

Furthermore, in [138], the authors employed the asymptotic expansion (8.1) for computing an accurate approximation of $\lambda_j(T_n(f))$ for very large n, provided that the values $\lambda_{j_1}(T_{n_1}(f)), \ldots, \lambda_{j_s}(T_{n_s}(f))$ are available for moderate sizes $n_1, \ldots, n_s$ with $\theta_{j_1,n_1} = \cdots = \theta_{j_s,n_s} = \theta_{j,n}$, $s \geq 2$. The second and main purpose of present research is to carry out this idea and to support it by numerical experiments, accompanied by an appropriate error analysis in the more general case of the preconditioned matrices $\mathcal{P}_n(f,g)$. In particular, we devise an algorithm to compute $\lambda_j(\mathcal{P}_n(f,g))$ with a high level of accuracy and a relatively low computational cost. The algorithm is completely analogous to the extrapolation procedure, which is employed in the context of Romberg integration (to obtain high precision approximations of an integral from a few coarse trapezoidal approximations [144, Section 3.4], see also [10] for more advanced algorithms). In this regard, the asymptotic expansion (8.1) plays here the same role as the Euler–Maclaurin summation formula [144, Section 3.3].

The third and last purpose of this research work is to formulate, on the basis of numerical experiments, a conjecture on the higher-order asymptotic of the eigenvalues if the monotonicity assumption on $r = f/g$ is not in force. We also illustrate how this conjecture can be used along with our extrapolation algorithm in order to compute some of the eigenvalues of $\mathcal{P}_n(f,g)$ in the case where r is nonmonotone.

8.2 Error Bounds for the Coefficients c_k in the Asymptotic Expansion

We start this section by manipulating the error expression implicitly given in (8.1), the goal being that of using extrapolation methods [10]. In fact, if we assume that the relations in (8.1) hold, then we can write

$$E_{j,n,0} = \sum_{k=1}^{\alpha} c_k(\theta_{j,n})\, h^k + E_{j,n,\alpha}\,, \tag{8.3}$$

where $E_{j,n,0} = \lambda_j(\mathcal{P}_n(f,g)) - r(\theta_{j,n})$.

We now suppose to know the eigenvalues for different (small) n_i namely $\{(n_1, \lambda_{j_1}(\mathcal{P}_{n_1}(f,g))),\ (n_2, \lambda_{j_2}(\mathcal{P}_{n_2}(f,g))),\ \cdots,\ (n_\alpha, \lambda_{j_\alpha}(\mathcal{P}_{n_\alpha}(f,g)))\}$, where $n_1, n_2, \cdots, n_\alpha$ and $j_1, j_2, \cdots, j_\alpha$ are chosen in such a way that $j_1/(n_1+1) = j_2/(n_2+1) = \cdots = j_\alpha/(n_\alpha+1)$.

By defining $h_1 = 1/(n_1+1), h_2 = 1/(n_2+1), \ldots, h_\alpha = 1/(n_\alpha+1)$, for a given set of eigenvalues, equation (8.3) can be written as

$$\begin{aligned}
E_{j_1,n_1,0} &= \sum_{k=1}^{\alpha} c_k(\theta_{j_1,n_1})\, h_1^k + E_{j_1,n_1,\alpha},\\
E_{j_2,n_2,0} &= \sum_{k=1}^{\alpha} c_k(\theta_{j_2,n_2})\, h_2^k + E_{j_2,n_2,\alpha},\\
E_{j_3,n_3,0} &= \sum_{k=1}^{\alpha} c_k(\theta_{j_3,n_3})\, h_3^k + E_{j_3,n_3,\alpha},\\
&\ \vdots\\
E_{j_\alpha,n_\alpha,0} &= \sum_{k=1}^{\alpha} c_k(\theta_{j_\alpha,n_\alpha})\, h_\alpha^k + E_{j_\alpha,n_\alpha,\alpha}.
\end{aligned} \tag{8.4}$$

Let c, $\tilde{c}$ be the vectors

$$c = [c_1, c_2, \ldots, c_\alpha]^T; \qquad \tilde{c} = [\tilde{c}_1, \tilde{c}_2, \ldots, \tilde{c}_\alpha]^T,$$

and let A be the coefficient matrix of size $\alpha \times \alpha$ with $(A)_{i,j} = h_i^j$. Hence the set of equations (8.4) can be written in matrix form as

$$Ac = b_0 - b_\alpha\,, \tag{8.5}$$

where $b_0 = [E_{j_1,n_1,0}, E_{j_2,n_2,0}, \ldots, E_{j_\alpha,n_\alpha,0}]^T$ and $b_\alpha = [E_{j_1,n_1,\alpha}, E_{j_2,n_2,\alpha}, \ldots, E_{j_\alpha,n_\alpha,\alpha}]^T$. Furthermore, by neglecting the higher order errors, we may define an approximation $\tilde{c}$ of c according to the expression below

$$A\tilde{c} = b_0\,. \tag{8.6}$$

By solving the linear system of equations above, the approximation of c is easily obtained since the matrix size is very small. In a subsequent step we derive upper-bounds for $|\tilde{c}-c|$: in reality, equations (8.5) and (8.6) leads to

$$A(\tilde{c}-c) = b_\alpha. \tag{8.7}$$

If we define $\Delta c = \tilde{c}-c$ and $\eta_i = \frac{E_{j_i,n_i,\alpha}}{h_i^{\alpha+1}}$ for $i=1,\ldots,\alpha$, then the system (8.7) can be written as

$$A\Delta c = \begin{bmatrix} \eta_1 h_1^{\alpha+1} \\ \eta_2 h_2^{\alpha+1} \\ \vdots \\ \eta_\alpha h_\alpha^{\alpha+1} \end{bmatrix}, \tag{8.8}$$

with $|\eta_i| \leq C_\alpha$ for $i=1,\ldots,\alpha$, where C_α is a constant. The coefficient matrix can be expressed as

$$A = \begin{bmatrix} h_1 & h_1^2 & \ldots & h_1^\alpha \\ h_2 & h_2^2 & \ldots & h_2^\alpha \\ \vdots & \vdots & & \vdots \\ h_\alpha & h_\alpha^2 & \ldots & h_\alpha^\alpha \end{bmatrix} = \begin{bmatrix} h_1 & & & \\ & h_2 & & \\ & & \ddots & \\ & & & h_\alpha \end{bmatrix} V(h_1,\ldots,h_\alpha),$$

where $V(h_1,\ldots,h_\alpha)$ is the Vandermonde matrix of order α corresponding to $h_1,\ldots,h_\alpha$.

By assuming $W = V^{-1}(h_1,\ldots,h_\alpha)$, we deduce

$$(W)_{i,j} = \begin{cases} (-1)^{\alpha-i}\left(\dfrac{\displaystyle\sum_{\substack{1\leq k_1<\ldots<k_{\alpha-i}\leq\alpha \\ k_1,\ldots,k_{\alpha-i}\neq j}} h_{k_1}\cdots h_{k_{\alpha-i}}}{\displaystyle\prod_{\substack{1\leq k\leq\alpha \\ k\neq j}}(h_j-h_k)}\right) & 1\leq i<\alpha, \\ \dfrac{1}{\displaystyle\prod_{\substack{1\leq k\leq\alpha \\ k\neq j}}(h_j-h_k)} & i=\alpha. \end{cases} \tag{8.9}$$

Therefore for the inversion of the matrix A we have

$$(A^{-1})_{i,j} = \begin{cases} (-1)^{\alpha-i}\left(\dfrac{\displaystyle\sum_{\substack{1\le k_1<\ldots<k_{\alpha-i}\le\alpha\\ k_1,\ldots,k_{\alpha-i}\neq j}} h_{k_1}\cdots h_{k_{\alpha-i}}}{h_j\displaystyle\prod_{\substack{1\le k\le\alpha\\ k\neq j}}(h_j-h_k)}\right) & 1\le i<\alpha,\\ \dfrac{1}{h_j\displaystyle\prod_{\substack{1\le k\le\alpha\\ k\neq j}}(h_j-h_k)} & i=\alpha, \end{cases} \tag{8.10}$$

and we can obtain an explicit expression for $(\Delta c)_i$, $i=1,\ldots,\alpha$, that is

$$(\Delta c)_i = \sum_{j=1}^{\alpha}(A^{-1})_{i,j}\eta_j h_j^{\alpha+1}. \tag{8.11}$$

Case 1. If $i=\alpha$, then

$$(\Delta c)_\alpha = \sum_{j=1}^{\alpha}\frac{\eta_j h_j^{\alpha+1}}{h_j\displaystyle\prod_{\substack{1\le k\le\alpha\\ k\neq j}}(h_j-h_k)}.$$

Whence, from the fact that $|\eta_i|\le C_\alpha$ for $i=1,\ldots,\alpha$,

$$|(\Delta c)_\alpha| \le \sum_{j=1}^{\alpha}\frac{|\eta_j| h_j^{\alpha+1}}{h_j\displaystyle\prod_{\substack{1\le k\le\alpha\\ k\neq j}}|h_j-h_k|} \le \sum_{j=1}^{\alpha}\frac{C_\alpha h_j^{\alpha}}{\displaystyle\prod_{\substack{1\le k\le\alpha\\ k\neq j}}|h_j-h_k|}.$$

With the choice $h_j=\frac{1}{m^{j-1}}h_1$ for $j=1,\ldots,\alpha$, m positive integer, we have

$$\begin{aligned} |(\Delta c)_\alpha| &\le C_\alpha\sum_{j=1}^{\alpha}\frac{(\frac{h_1}{m^{j-1}})^\alpha}{\displaystyle\prod_{\substack{1\le k\le\alpha\\ k\neq j}} h_1\left|\frac{1}{m^{j-1}}-\frac{1}{m^{k-1}}\right|}\\ &= C_\alpha h_1^\alpha\sum_{j=1}^{\alpha}\frac{(\frac{1}{m^{j-1}})^\alpha}{h_1^{\alpha-1}\displaystyle\prod_{\substack{1\le k\le\alpha\\ k\neq j}}\left|\frac{1}{m^{j-1}}-\frac{1}{m^{k-1}}\right|}=\\ &= h_1 C_\alpha\sum_{j=1}^{\alpha}\frac{(\frac{1}{m^{j-1}})^\alpha}{\displaystyle\prod_{\substack{1\le k\le\alpha\\ k\neq j}}\left|\frac{1}{m^{j-1}}-\frac{1}{m^{k-1}}\right|}=O(h_1). \end{aligned}$$

Case 2. If $i = 1, \ldots, \alpha - 1$, then

$$(\Delta c)_i = \sum_{j=1}^{\alpha} (-1)^{\alpha-i} \eta_j h_j^{\alpha+1} \frac{\displaystyle\sum_{\substack{1 \le k_1 < \ldots < k_{\alpha-i} \le \alpha \\ k_1, \ldots, k_{\alpha-i} \neq j}} h_{k_1} \cdots h_{k_{\alpha-i}}}{h_j \displaystyle\prod_{\substack{1 \le k \le \alpha \\ k \neq j}} (h_j - h_k)},$$

that is different from the case $i = \alpha$ just for the numerator

$$\sum_{\substack{1 \le k_1 < \ldots < k_{\alpha-i} \le \alpha \\ k_1, \ldots, k_{\alpha-i} \neq j}} h_{k_1} \cdots h_{k_{\alpha-i}}.$$

As a consequence

$$|(\Delta c)_i| \le C_\alpha \sum_{j=1}^{\alpha} h_j^{\alpha} \frac{\displaystyle\sum_{\substack{1 \le k_1 < \ldots < k_{\alpha-i} \le \alpha \\ k_1, \ldots, k_{\alpha-i} \neq j}} h_{k_1} \cdots h_{k_{\alpha-i}}}{\displaystyle\prod_{\substack{1 \le k \le \alpha \\ k \neq j}} |h_j - h_k|}.$$

With the choice $h_j = \frac{1}{m^{j-1}} h_1$ for $j = 1, \ldots, \alpha$, we infer

$$\begin{aligned}
|(\Delta c)_i| &\le C_\alpha \sum_{j=1}^{\alpha} \left(\frac{h_1}{m^{j-1}} \right)^{\alpha} \frac{\displaystyle\sum_{\substack{1 \le k_1 < \ldots < k_{\alpha-i} \le \alpha \\ k_1, \ldots, k_{\alpha-i} \neq j}} h_1^{\alpha-i} \left(\frac{1}{m^{k_1-1}} \frac{1}{m^{k_2-1}} \cdots \frac{1}{m^{k_{\alpha-i}-1}} \right)}{\displaystyle\prod_{\substack{1 \le k \le \alpha \\ k \neq j}} h_1 \left| \frac{1}{m^{j-1}} - \frac{1}{m^{k-1}} \right|} \\
&= C_\alpha \sum_{j=1}^{\alpha} \left(\frac{1}{m^{j-1}} \right)^{\alpha} \left(\frac{h_1^{\alpha} h_1^{\alpha-i}}{h_1^{\alpha-1}} \right) \\
&\frac{\displaystyle\sum_{\substack{1 \le k_1 < \ldots < k_{\alpha-i} \le \alpha \\ k_1, \ldots, k_{\alpha-i} \neq j}} \left(\frac{1}{m^{k_1-1}} \frac{1}{m^{k_2-1}} \cdots \frac{1}{m^{k_{\alpha-i}-1}} \right)}{\displaystyle\prod_{\substack{1 \le k \le \alpha \\ k \neq j}} \left| \frac{1}{m^{j-1}} - \frac{1}{m^{k-1}} \right|}
\end{aligned}$$

$$= h_1^{\alpha-i+1} C_\alpha \sum_{j=1}^{\alpha} \left(\frac{1}{m^{j-1}}\right)^{\alpha} \frac{\displaystyle\sum_{\substack{1\le k_1<\ldots<k_{\alpha-i}\le\alpha\\ k_1,\ldots,k_{\alpha-i}\neq j}} \left(\frac{1}{m^{k_1-1}}\frac{1}{m^{k_2-1}}\cdots\frac{1}{m^{k_{\alpha-i}-1}}\right)}{\displaystyle\prod_{\substack{1\le k\le\alpha\\ k\neq j}} \left|\frac{1}{m^{j-1}}-\frac{1}{m^{k-1}}\right|}$$

$$= O(h_1^{\alpha-i+1}).$$

As a conclusion, with the choice $h_j = \frac{1}{m^{j-1}} h_1$ for $j = 1, \ldots, \alpha$ and under the assumption that the asymptotic expansion reported in (8.1) is true, we deduce

$$|(\Delta c)_i| = O(h_1^{\alpha-i+1}), \tag{8.12}$$

for $i = 1, \ldots, \alpha$.

8.3 Error Bounds for Numerically Approximated Eigenvalues

The goal of this short section is to provide error bounds based on the linear system in (8.6) for the computation of the eigenvalues of $\mathcal{P}_n(f,g)$: of course these error bounds are based on the conjecture that the relations reported in (8.1) are true. However, as we can see in Section 8.4, the numerical tests fully support the existence of the considered asymptotic expansion.

Indeed, as already observed, by solving (8.6), we can approximate c_k. Once we have the values of c_k, we can approximate the eigenvalues λ_{j_β} of a large dimension matrix of size n_β, here $n_\beta + 1 = m^{\beta-1}(n_1+1)$. The asymptotic expansion (8.3) can be written as

$$E_{j_\beta,n_\beta,0} = \bar{h}_\beta^T c + E_{j_\beta,n_\beta,\alpha}\,. \tag{8.13}$$

By subtraction $\bar{h}_\beta^T\tilde{c}$ from both sides of the equation above, we find

$$\begin{aligned} E_{j_\beta,n_\beta,0} - \bar{h}_\beta^T\tilde{c} &= \bar{h}_\beta^T(c-\tilde{c}) + E_{j_\beta,n_\beta,\alpha}, \\ \lambda_j(\mathcal{P}_{n_\beta}(f,g)) - r(\theta_{j,n_\beta}) - \bar{h}_\beta^T\tilde{c} &= \bar{h}_\beta^T\Delta c + E_{j_\beta,n_\beta,\alpha}, \\ \left|\lambda_j(\mathcal{P}_{n_\beta}(f,g)) - r(\theta_{j,n_\beta}) - \bar{h}_\beta^T\tilde{c}\right| &\leq \sum_{i=1}^{\alpha} h_\beta^i|(\Delta c)_i| + |E_{j_\beta,n_\beta,\alpha}|, \\ \left|\lambda_j(\mathcal{P}_{n_\beta}(f,g)) - r(\theta_{j,n_\beta}) - \bar{h}_\beta^T\tilde{c}\right| &\leq \sum_{i=1}^{\alpha} h_\beta^i|(\Delta c)_i| + C_\alpha h_\beta^{\alpha+1}, \end{aligned} \tag{8.14}$$

where $\bar{h}_\beta = [h_\beta, h_\beta^2, \cdots, h_\beta^\alpha]^T$, $|E_{j_\beta,n_\beta,\alpha}| \leq C_\alpha h_\beta^{\alpha+1}$ for some constant C_α and $|(\Delta c)_i|$ is given in (8.12).

8.4 Numerical Tests

In this section we want to present a few numerical experiments to support the asymptotic expansion (8.1) in the case where one or more properties of the following list are satisfied:

1. $f''(0) \neq 0$ (see Example 1, Example 3, and Example 5),
2. $f''(0) = 0$ (see Example 2 and Example 4),
3. $\min g > 0$ (see Example 1, Example 2, and Example 5),
4. $\min g = 0$ (see Example 3 and Example 4),
5. $r = f/g$ is non monotone (see Example 5).

The approximation of eigenvalues of large matrices in each case is also computed. The expansion (8.1) for $\alpha = 4$ is

$$\begin{aligned} \lambda_j(\mathcal{P}_n(f,g)) &= r(\theta_{j,n}) + c_1(\theta_{j,n})\,h + c_2(\theta_{j,n})\,h^2 + c_3(\theta_{j,n})\,h^3 \\ &\quad + c_4(\theta_{j,n})\,h^4 + E_{j,n,4}, \\ E_{j,n,0} &= \lambda_j(\mathcal{P}_n(f,g)) - r(\theta_{j,n}) = c_1(\theta_{j,n})\,h + c_2(\theta_{j,n})\,h^2 \\ &\quad + c_3(\theta_{j,n})\,h^3 + c_4(\theta_{j,n})\,h^4 + E_{j,n,4}\,. \end{aligned} \tag{8.15}$$

In all numerical examples we choose four matrix-size values, that is n_i for $i \in \{1,2,3,4\}$, in a way that they satisfy $n_i = m^{i-1}(n_1+1)-1$, with m being a positive integer. The expansion (8.15) for the set of the four dimensions n_i can be written as

$$\begin{aligned}
E_{j_1,n_1,0} &= c_1(\theta_{j_1,n_1})\, h_1 + c_2(\theta_{j_1,n_1})\, h_1^2 + c_3(\theta_{j_1,n_1})\, h_1^3 + c_4(\theta_{j_1,n_1})\, h_1^4 \\
&\quad + E_{j_1,n_1,4}, \\
E_{j_2,n_2,0} &= c_1(\theta_{j_2,n_2})\, h_2 + c_2(\theta_{j_2,n_2})\, h_2^2 + c_3(\theta_{j_2,n_2})\, h_2^3 + c_4(\theta_{j_2,n_2})\, h_2^4 \\
&\quad + E_{j_2,n_2,4}, \\
E_{j_3,n_3,0} &= c_1(\theta_{j_3,n_3})\, h_3 + c_2(\theta_{j_3,n_3})\, h_3^2 + c_3(\theta_{j_3,n_3})\, h_3^3 + c_4(\theta_{j_3,n_3})\, h_3^4 \\
&\quad + E_{j_3,n_3,4}, \\
E_{j_4,n_4,0} &= c_1(\theta_{j_4,n_4})\, h_4 + c_2(\theta_{j_4,n_4})\, h_4^2 + c_3(\theta_{j_4,n_4})\, h_4^3 + c_4(\theta_{j_4,n_4})\, h_4^4 \\
&\quad + E_{j_4,n_4,4},
\end{aligned} \tag{8.16}$$

where $h_i = \frac{1}{n_i+1}$ and $j_i = m^{i-1} j_1$ for $i \in \{1,2,3,4\}$. Notice that $\theta_{j_i,n_i} = \theta_{j_1,n_1} = \bar{\theta}$ for a fixed $j_1 \in \{1, 2, \cdots, n_1\}$. We are interested in the numerical approximation of $c_i(\bar{\theta})$ for $i \in \{1,2,3,4\}$ and then in the precise numerical approximation of the eigenvalue of $\mathcal{P}_n(f,g)$ for large n. The set of equations (8.16) can be written as

$$\begin{aligned}
E_{j_1,n_1,0} &= \tilde{c}_1(\bar{\theta})\, h_1 + \tilde{c}_2(\bar{\theta})\, h_1^2 + \tilde{c}_3(\bar{\theta})\, h_1^3 + \tilde{c}_4(\bar{\theta})\, h_1^4, \\
E_{j_2,n_2,0} &= \tilde{c}_1(\bar{\theta})\, h_2 + \tilde{c}_2(\bar{\theta})\, h_2^2 + \tilde{c}_3(\bar{\theta})\, h_2^3 + \tilde{c}_4(\bar{\theta})\, h_2^4, \\
E_{j_3,n_3,0} &= \tilde{c}_1(\bar{\theta})\, h_3 + \tilde{c}_2(\bar{\theta})\, h_3^2 + \tilde{c}_3(\bar{\theta})\, h_3^3 + \tilde{c}_4(\bar{\theta})\, h_3^4, \\
E_{j_4,n_4,0} &= \tilde{c}_1(\bar{\theta})\, h_4 + \tilde{c}_2(\bar{\theta})\, h_4^2 + \tilde{c}_3(\bar{\theta})\, h_4^3 + \tilde{c}_4(\bar{\theta})\, h_4^4.
\end{aligned} \tag{8.17}$$

We solve the system of linear equations above for $j_1 \in \{1, 2, \cdots, n_1\}$ to compute $\tilde{c}_i(\bar{\theta})$. The computed $\tilde{c}_i$ are used to approximate the eigenvalues of large size n_β by exploiting the following relation

$$\tilde{\lambda}_{j_\beta}(\mathcal{P}_{n_\beta}(f,g)) = r(\theta_{j_\beta,n_\beta}) + \bar{h}_\beta^T \tilde{c}\,. \tag{8.18}$$

Example 8.1. *Let g, f, and r be the functions defined as*

$$\begin{aligned}
f(\theta) &= 4 - 2\cos(\theta) - 2\cos(2\theta) \\
&= (2 - 2\cos(\theta))(3 + 2\cos(\theta))\,, \\
g(\theta) &= 3 + 2\cos(\theta)\,, \\
r(\theta) &= \frac{f(\theta)}{g(\theta)} = 2 - 2\cos(\theta)\,,
\end{aligned}$$

where $\theta \in [0,\pi]$. The graphs of generating functions are shown in left panel of

Figure 8.1, and the approximations $\tilde{c}_k$, for $k = 1, 2, 3, 4$ are shown in the right panel. Note that $g(\theta) > 0$, $\forall\, \theta \in [0, \pi]$, $f''(0) \neq 0$, and furthermore $r(\theta)$ is monotone. We set $n = n_1 \in \{40, 60, 80, 100\}$ and $m = 2$.

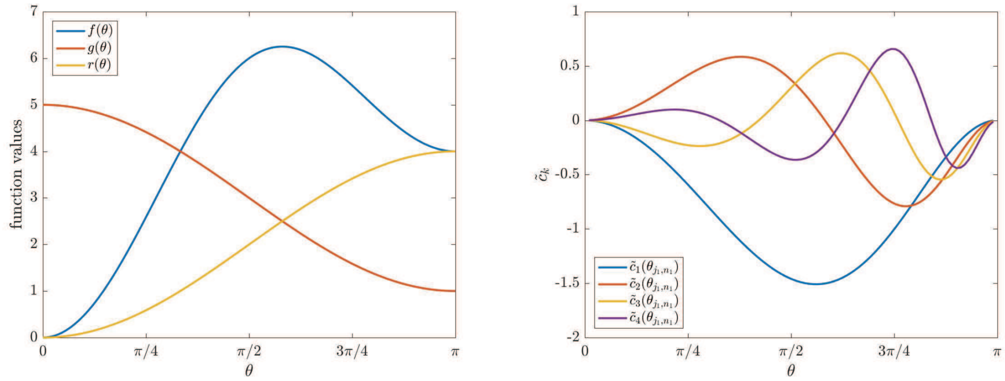


FIGURE 8.1: Example 1: Generating functions (f, g, and r) and $\tilde{c}_k$ for $k = 1, 2, 3, 4$.

Example 8.2. *Let g, f, and r be the functions defined as*

$$\begin{aligned} f(\theta) &= 20 - 30\cos(\theta) + 12\cos(2\theta) - 2\cos(3\theta) = (2 - 2\cos(\theta))^3\,, \\ g(\theta) &= 3 + 2\cos(\theta)\,, \\ r(\theta) &= \frac{f(\theta)}{g(\theta)} = \frac{(2 - 2\cos(\theta))^3}{3 + 2\cos(\theta)}\,, \end{aligned}$$

where $\theta \in [0, \pi]$. The graphs of generating functions are shown in left panel of Figure 8.2, and the approximations $\tilde{c}_k$, for $k = 1, 2, 3, 4$ are shown in the right panel. Remark that $g(\theta) > 0$, $\forall\, \theta \in [0, \pi]$, $f''(0) = 0$, and furthermore $r(\theta)$ is monotone. We set $n = n_1 \in \{40, 60, 80, 100\}$ and $m = 2$.

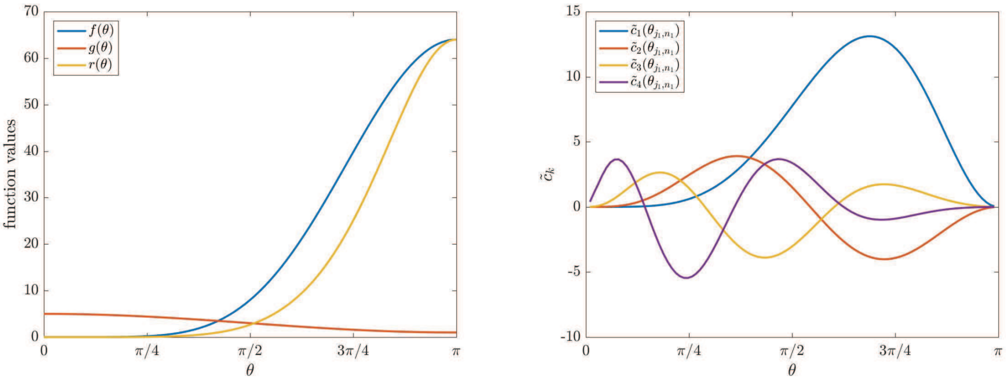


FIGURE 8.2: Example 2: Generating functions (f, g, and r) and $\tilde{c}_k$ for $k = 1, 2, 3, 4$.

There is an important issue to discuss here. Both the functions f and r attain the minimum at $\theta = 0$ with a very high order. Indeed we have $f(\theta), r(\theta) \approx \theta^6$, with $\phi_1 \approx \phi_2$ being the symmetric, transitive relation telling that there exist positive constants $c, C > 0$ such that $c\phi_1 \leq \phi_2 \leq C\phi_1$ on the whole definition domain $[0,\pi]$. Therefore for fixed j (independent of n) the j-th smallest eigenvalue of $\mathcal{P}_n(f,g)$ is asymptotic to $k_j h^6$, k_j positive constant depending on j but not on n: the reader is refereed to [141] for the preconditioned case with the limitation $j = 1$ and to [128] and references therein for very elegant and precise estimates regarding the pure Toeplitz case.

Now if we fix j and we put together $\lambda_j\left(\mathcal{P}_n(f,g)\right) \approx h^6$ with relations (8.3)–(8.4) then the only possibility for avoiding a contradiction is that the functions $c_1(\theta), c_2(\theta), c_3(\theta), c_4(\theta), c_5(\theta)$ all vanish at $\theta = 0$.

The approximations $\tilde{c}_k$, for $k = 1,2,3,4$ shown in the right panel of Figure 8.2 are coherent with the above mathematical conclusion and in fact all these approximations vanish simultaneously at $\theta = 0$ (the fifth is not displayed, but we computed it and it also equals to zero at $\theta = 0$, while, as expected from an extension of the results by [128] to the preconditioned Toeplitz case, the sixth is nonzero at $\theta = 0$).

Since the argument and the conclusions are the very same, we anticipate that the discussion can be repeated verbatim for Example 4, where the functions f and r attain the minimum at $\theta = 0$ with order 10. As a consequence, we expect that the functions $c_1(\theta), \ldots, c_9(\theta)$ all simultaneously vanish at $\theta = 0$, while $c_{10}(0) \neq 0$: this is confirmed for the first four of them as reported in the right panel of Figure 8.4.

Example 8.3. *Let g, f, and r be the functions defined as*

$$\begin{aligned}
f(\theta) &= 1 + \cos(\theta) + \frac{1}{4}\cos(2\theta) + \frac{1}{5}\cos(3\theta) + \frac{1}{10}\cos(4\theta) + \frac{1}{10}\cos(5\theta)\,, \\
g(\theta) &= 2 - 2\cos(\theta)\,, \\
r(\theta) &= \frac{f(\theta)}{g(\theta)} = \frac{1 + \cos(\theta) + \frac{1}{4}\cos(2\theta) + \frac{1}{5}\cos(3\theta) + \frac{1}{10}\cos(4\theta) + \frac{1}{10}\cos(5\theta)}{2 - 2\cos(\theta)}\,,
\end{aligned}$$

where $\theta \in [0,\pi]$. The graphs of generating functions are shown in left panel of Figure 8.3, and the approximations $\tilde{c}_k$, for $k = 1,2,3,4$ are shown in the right panel. Note that $\min g(\theta) = 0$, $\forall\, \theta \in [0,\pi]$, $f''(0) \neq 0$, and furthermore $r(\theta)$ is monotone. We set $n = n_1 \in \{40, 60, 80, 100\}$ and $m = 2$.

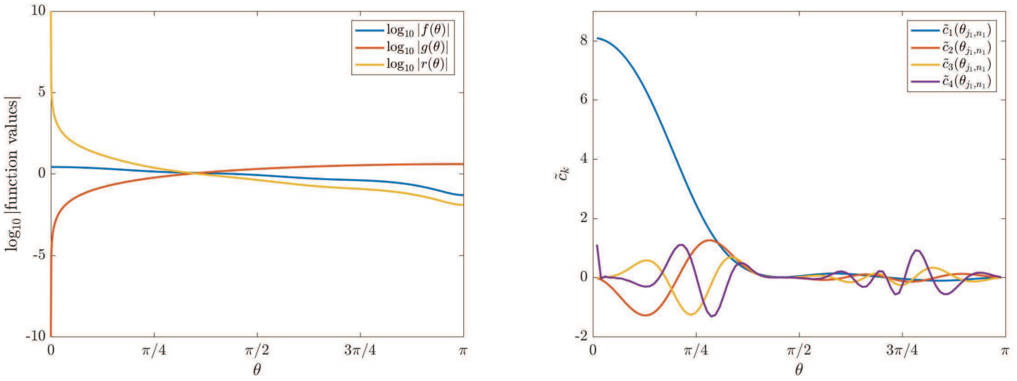


FIGURE 8.3: Example 3: Generating functions $(f, g,$ and $r)$ and $\tilde{c}_k$ for $k = 1, 2, 3, 4$.

Example 8.4. *Let g, f, and r be the functions defined as*

$$\begin{aligned} f(\theta) &= 252 - 420\cos(\theta) + 240\cos(2\theta) - 90\cos(3\theta) + 20\cos(4\theta) - 2\cos(5\theta) \\ &= (2 - 2\cos(\theta))^5 , \\ g(\theta) &= 2 + 2\cos(\theta) , \\ r(\theta) &= \frac{f(\theta)}{g(\theta)} = \frac{(2-2\cos(\theta))^5}{2+2\cos(\theta)} , \end{aligned}$$

where $\theta \in [0, \pi]$. The graphs of generating functions are shown in left panel of Figure 8.4, and the approximations $\tilde{c}_k$, for $k = 1, 2, 3, 4$ are shown in the right panel. Remark that $\min g(\theta) = 0$, $\forall\, \theta \in [0, \pi]$, $f''(0) = 0$, and furthermore $r(\theta)$ is monotone. We set $n = n_1 \in \{40, 60, 80, 100\}$ and $m = 2$.

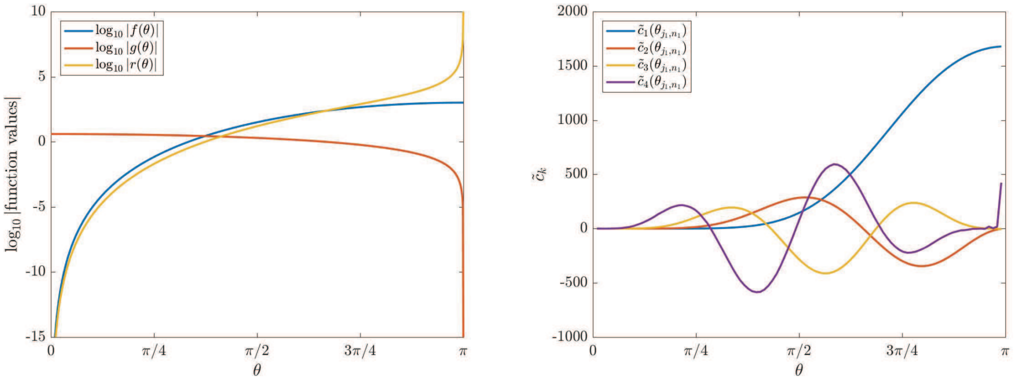


FIGURE 8.4: Example 4: Generating functions $(f, g,$ and $r)$ and $\tilde{c}_k$ for $k = 1, 2, 3, 4$.

Example 8.5. *Let g, f, and r be the functions defined as*

$$\begin{aligned} f(\theta) &= \frac{136}{17} + \frac{56}{17}\cos(\theta) - \frac{2}{17}\cos(2\theta) \\ &\quad + \frac{5}{17}\cos(3\theta) = (3-\cos(\theta) + \frac{5}{17}\cos(2\theta))(3+2\cos(\theta))\,, \\ g(\theta) &= 3+2\cos(\theta)\,, \\ r(\theta) &= \frac{f(\theta)}{g(\theta)} = 3-\cos(\theta)+\frac{5}{17}\cos(2\theta)\,, \end{aligned}$$

where $\theta \in [0,\pi]$. The graphs of generating functions are shown in left panel of Figure 8.5, and the approximations $\tilde{c}_k$, for $k = 1,2,3,4$ are shown in the right panel. Notice that $\min g(\theta) > 0$, $\forall\, \theta \in [0,\pi]$, $f''(0) \neq 0$, and furthermore $r(\cdot)$ is non monotone. We set $n = n_1 \in \{40, 60, 80, 100\}$ and $m = 2$.

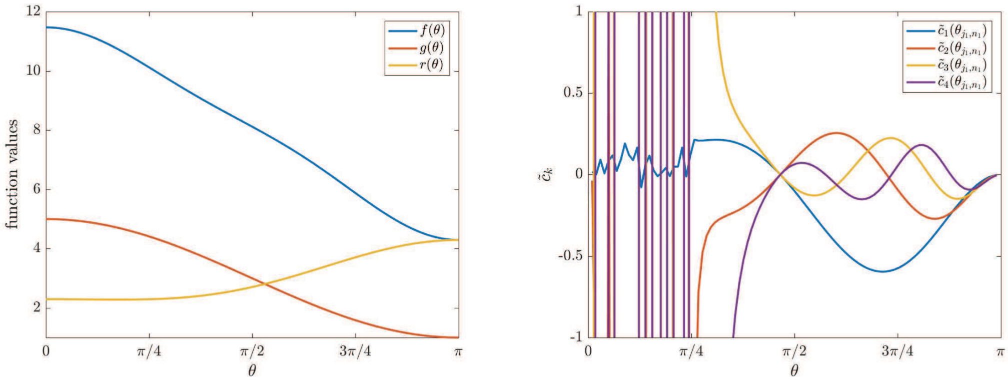


Figure 8.5: Example 5: Generating functions (f, g, and r) and $\tilde{c}_k$ for $k = 1,2,3,4$.

The numerical tests related to Examples 1 and 2, as in Figures 8.6 and 8.7, show that the error expansion (8.1) behaves as expected. In Figure 8.11 we also see that the approximated $\tilde{c}_k$ can be used for a large n to approximate the error term to (or almost to) machine precision.

In the numerical tests associated with Examples 3 and 4, as in Figures 8.8 and 8.9, we observe again that the error expansion is in accordance with (8.1). We also note a slight deviation for the largest eigenvalue and this has to be expected since we have $r(\theta_{1,n}) \to \infty$ as $n \to \infty$ for Example 3 (on the other hand for Example 4 we notice $r(\theta_{n,n}) \to \infty$ as $n \to \infty$). However, the approximation of the eigenvalues of $\mathcal{P}_n(f,g)$ is excellent and almost to machine precision as reported in Figure 8.12.

In the numerical test related to Example 5 we have a non monotone region for $\theta \in [0, 2\tan^{-1}(\sqrt{3/17})]$ where the proposed expansion does not work. Indeed

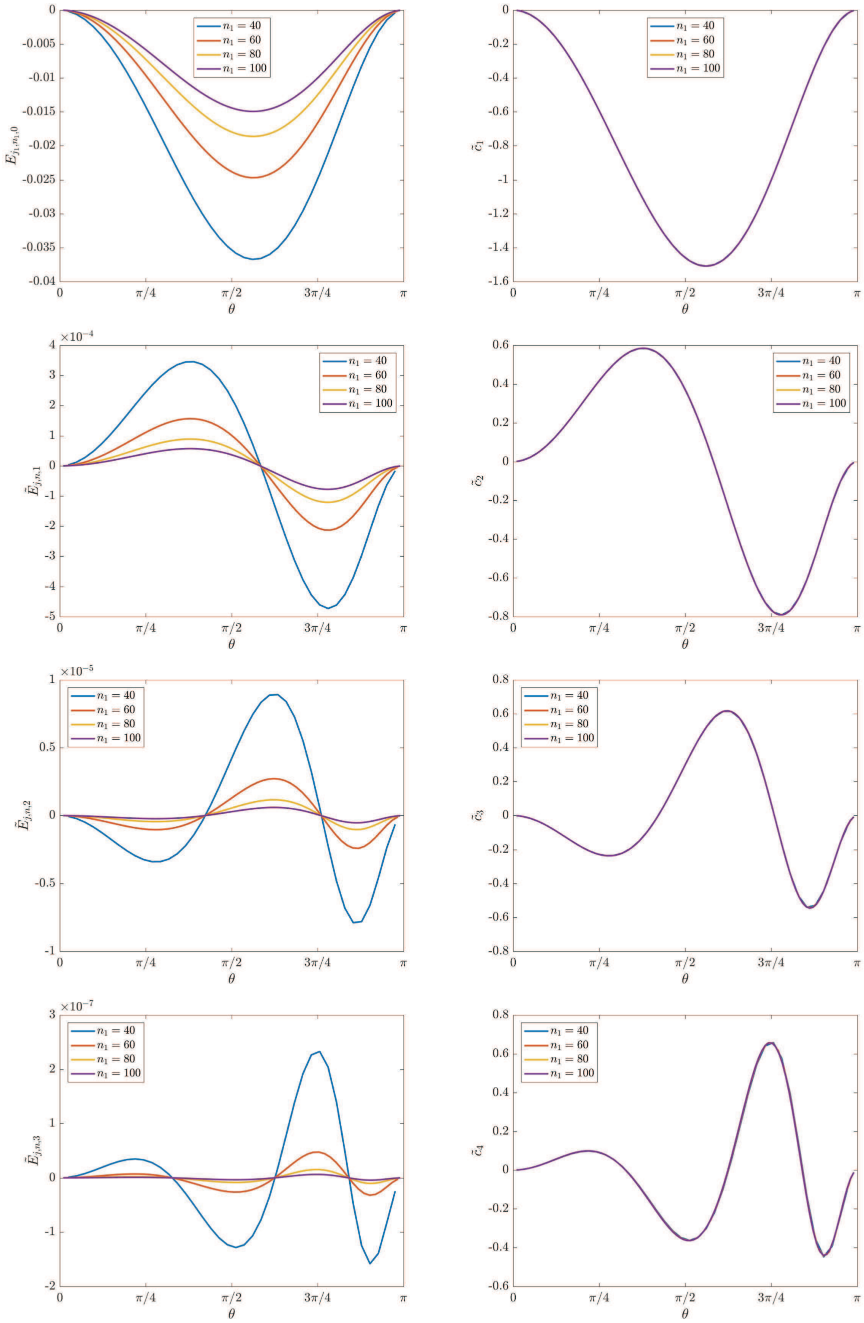


FIGURE 8.6: Example 1: $E_{j,n,0}$, $\tilde{E}_{j,n,k}$ $(k = 1, 2, 3)$, and $\tilde{c}_k$ $(k = 1, 2, 3, 4)$, for $n = n_1 = \{40, 60, 80, 100\}$.

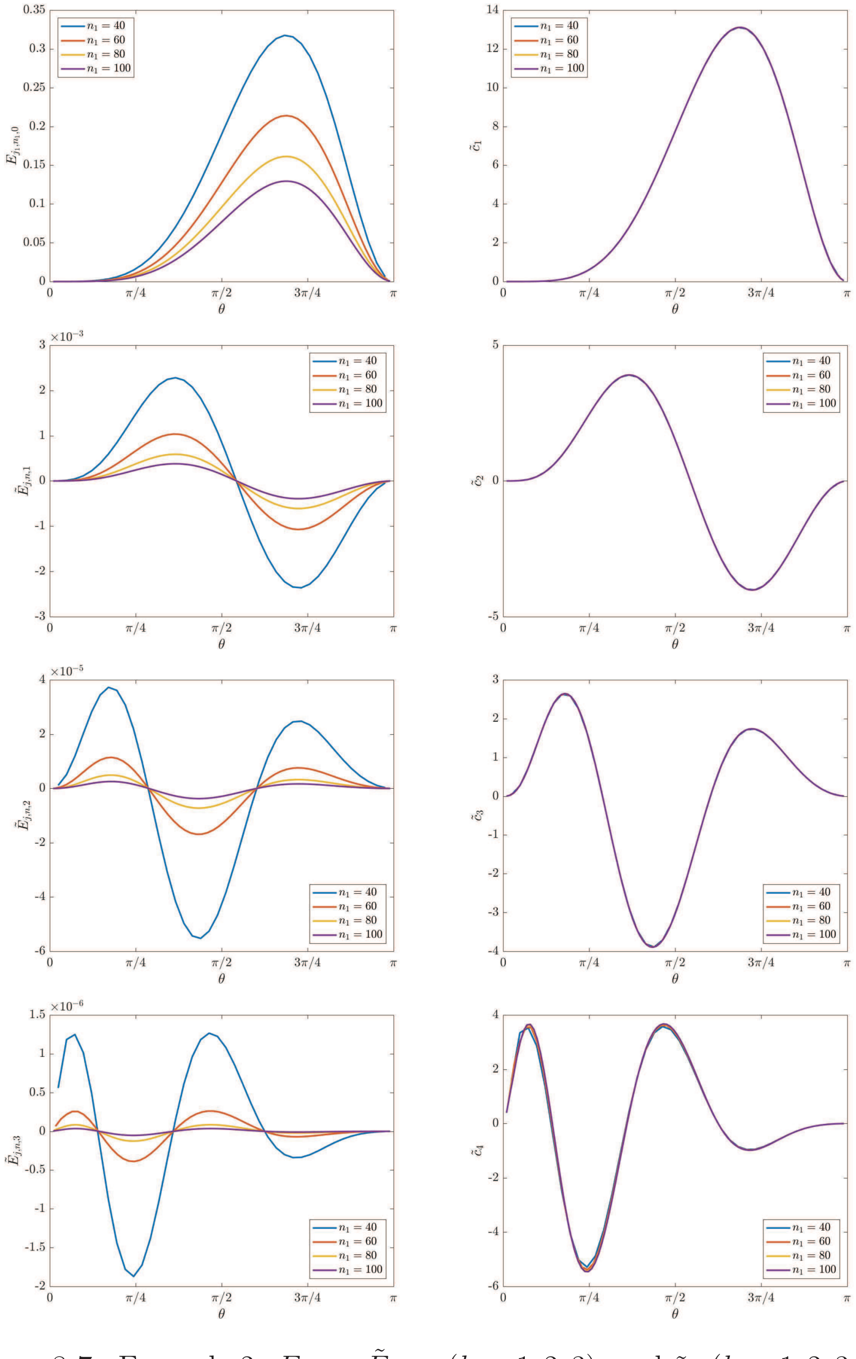


FIGURE 8.7: Example 2: $E_{j,n,0}$, $\tilde{E}_{j,n,k}$ $(k = 1, 2, 3)$, and $\tilde{c}_k$ $(k = 1, 2, 3, 4)$, for $n = n_1 = \{40, 60, 80, 100\}$.

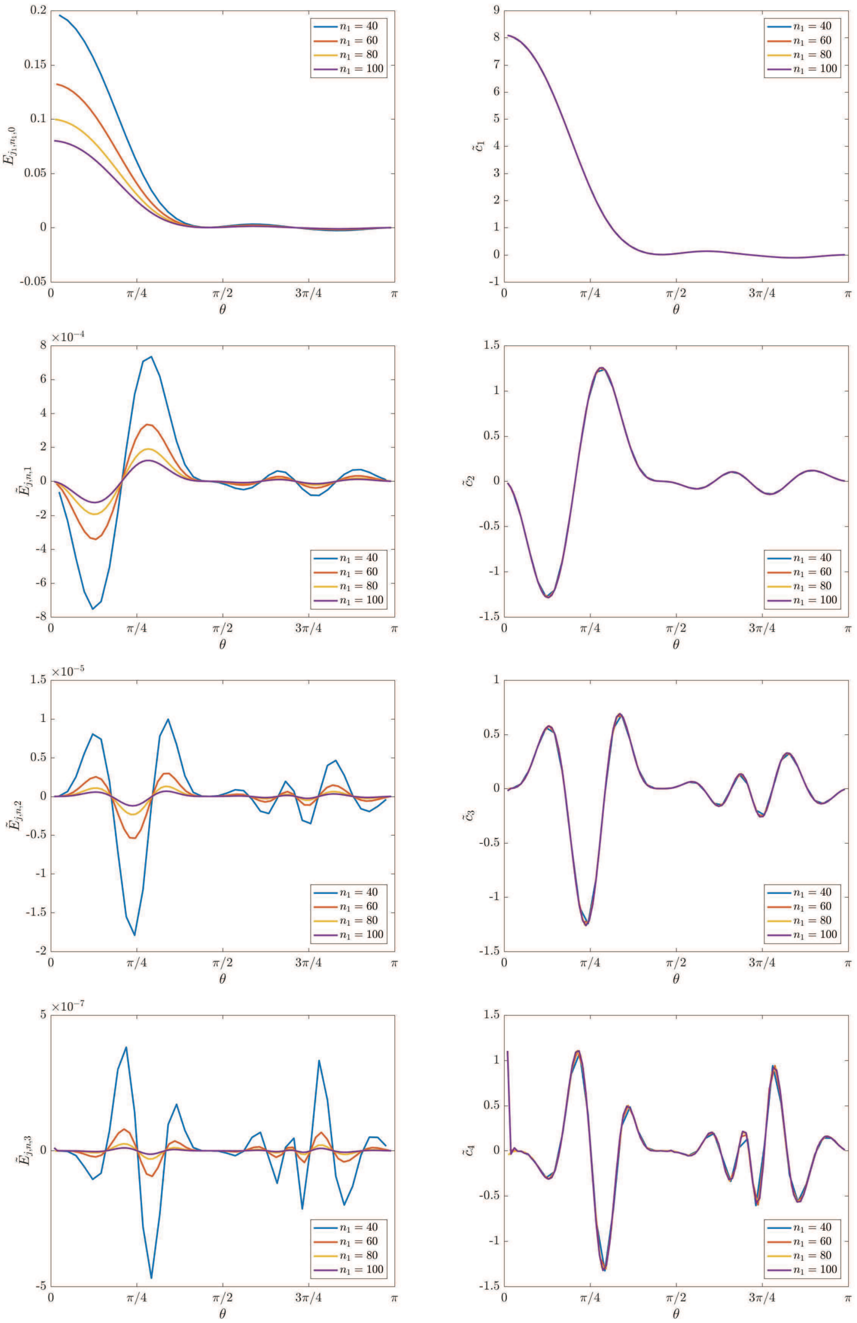


FIGURE 8.8: Example 3: $E_{j,n,0}$, $\tilde{E}_{j,n,k}$ $(k = 1, 2, 3)$, and $\tilde{c}_k$ $(k = 1, 2, 3, 4)$, for $n = n_1 = \{40, 60, 80, 100\}$.

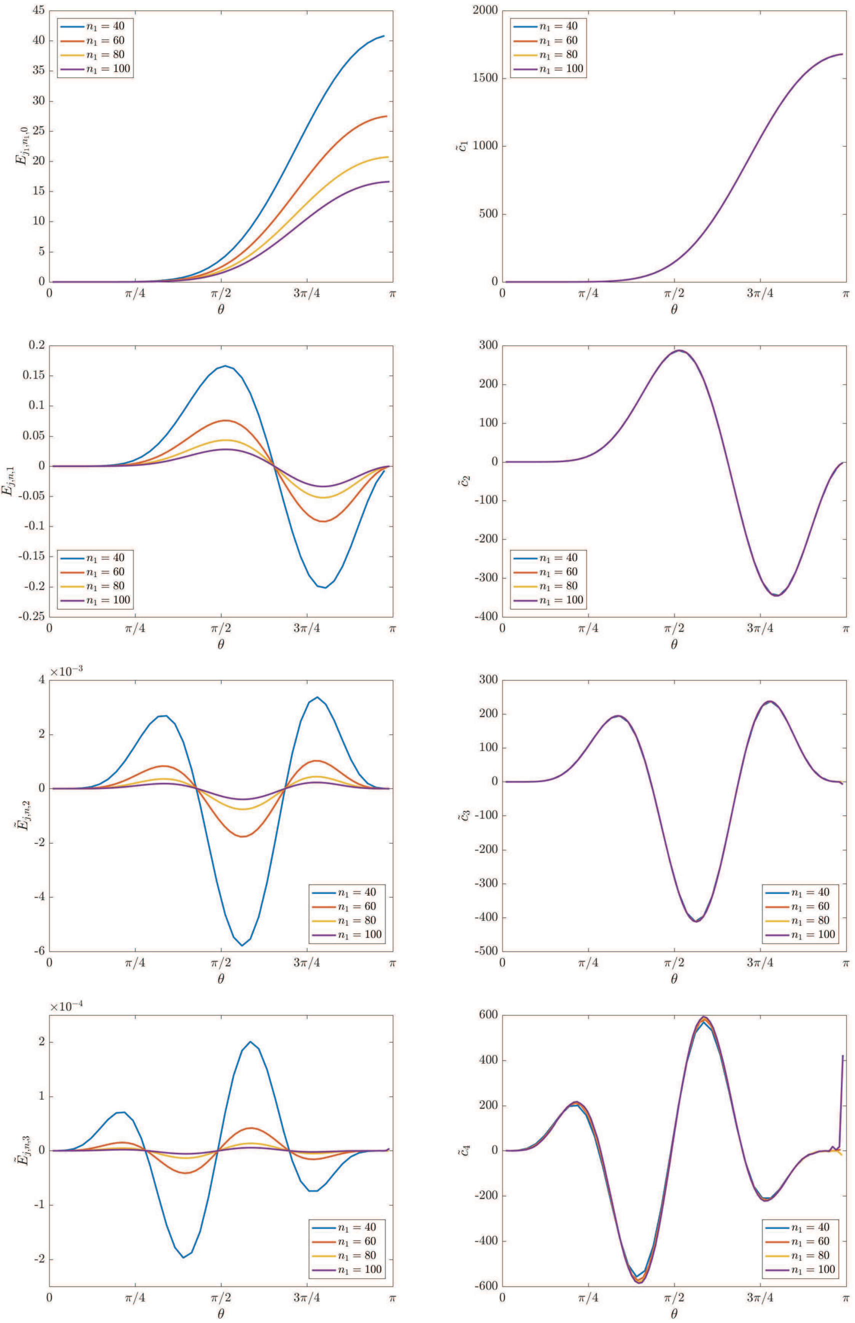


FIGURE 8.9: Example 4: $E_{j,n,0}$, $\tilde{E}_{j,n,k}$ $(k = 1, 2, 3)$, and $\tilde{c}_k$ $(k = 1, 2, 3, 4)$, for $n = n_1 = \{40, 60, 80, 100\}$.

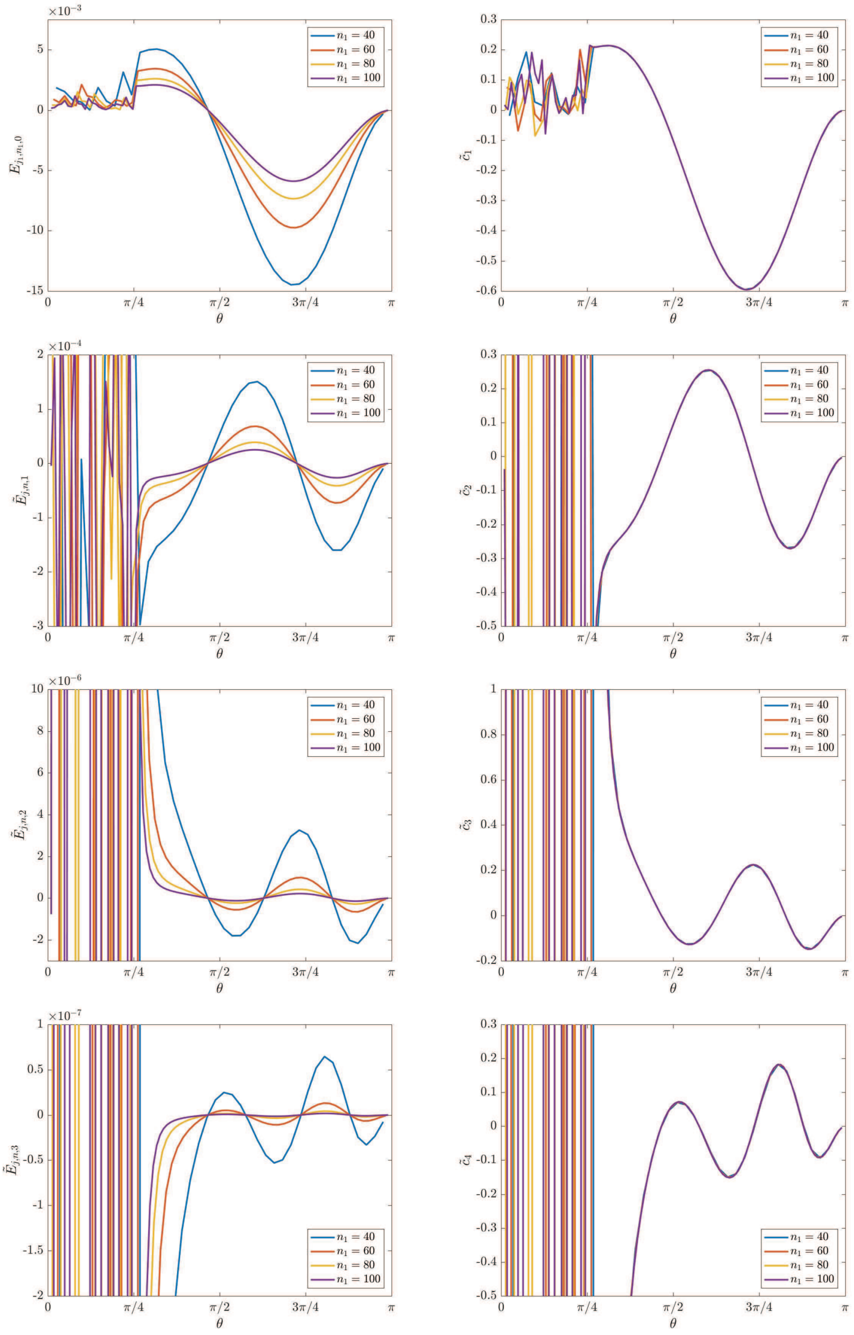


FIGURE 8.10: Example 5: $E_{j,n,0}$, $\tilde{E}_{j,n,k}$ $(k = 1, 2, 3)$, and $\tilde{c}_k$ $(k = 1, 2, 3, 4)$, for $n = n_1 = \{40, 60, 80, 100\}$.

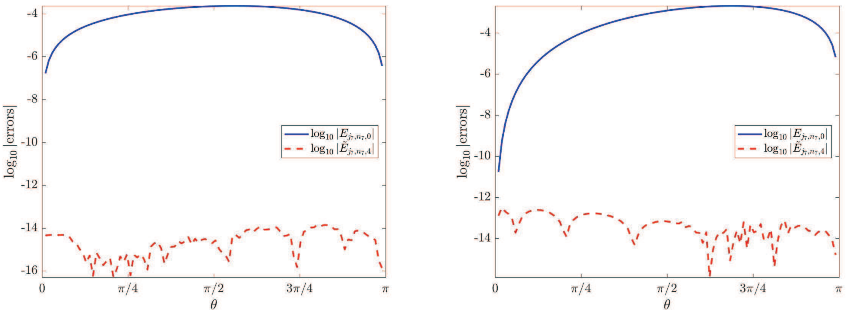


FIGURE 8.11: Example 1 and 2: The errors $\log_{10}|E_{j7,n_7,0}|$ and $\log_{10}|\tilde{E}_{j7,n_7,4}|$ for the 100 indices j_7 of $n_7 = 6463$ in (8.18), corresponding to $n_1 = 100$, and using $\tilde{c}_k, k = 1,2,3,4$, computed with $m = 2$.

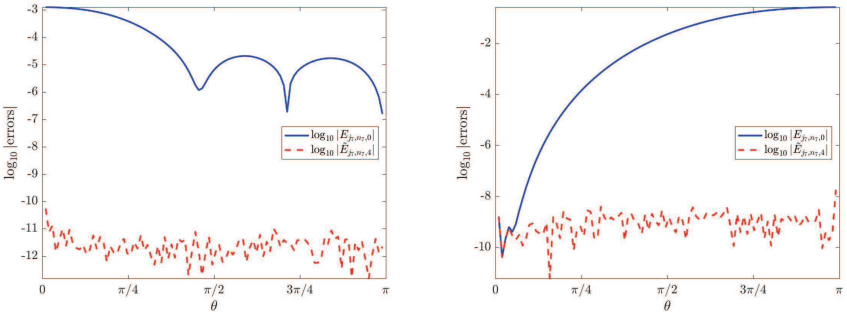


FIGURE 8.12: Example 3 and 4: The errors $\log_{10}|E_{j7,n_7,0}|$ and $\log_{10}|\tilde{E}_{j7,n_7,4}|$ for the 100 indices j_7 of $n_7 = 6463$ in (8.18), corresponding to $n_1 = 100$, and using $\tilde{c}_k, k = 1,2,3,4$, computed with $m = 2$.

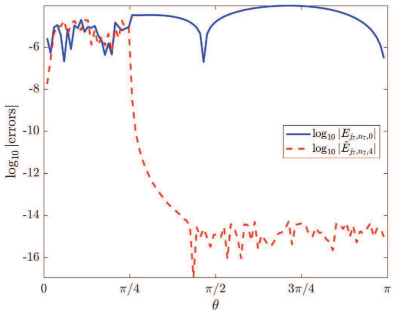


FIGURE 8.13: Example 5: The errors $\log_{10}|E_{j7,n_7,0}|$ and $\log_{10}|\tilde{E}_{j7,n_7,4}|$ for the 100 indices j_7 of $n_7 = 6463$ in (8.18), corresponding to $n_1 = 100$, and using $\tilde{c}_k, k = 1,2,3,4$, computed with $m = 2$. Note the non monotone part, $\theta \in [0, 2\tan^{-1}(\sqrt{3/17})]$, where the error is not improved.

additional errors are introduced when compared to $E_{j,n,0}$, since the sampling of $r(\theta_{j_1,n_1})$ leads to a poorer approximation after ordering than the procedure given by sampling $r(\theta_{j,n_7})$ first and then picking samples after ordering. However, the expansion is confirmed for the rest of the domain, as seen in Figure 8.10. Furthermore, in Figure 8.13 the expansion works well again for the monotone part, by allowing an approximation almost to machine precision of the eigenvalues of $\mathcal{P}_n(f,g)$.

However, even if the eigenvalues lying in the non monotone region give raise to an irregular error pattern, it seems that there exists a kind of 'deformed' periodicity in the error, like it is formally proven, without deformations, for the eigenvalues of $T_n(f)$, $f(\theta) = 2 - 2\cos(\omega\theta)$, $\omega \geq 2$ integer, and $g(\theta) = 1$ (see [137]). The latter observation indicates that a more complete study of this 'deformed' periodicity has to be considered in the future.

We finally observe that remarkable numerical results for the eigenvalues of $\mathcal{P}_n(f,g)$, as reported in Figures 8.11, 8.12, 8.13, really answer in the positive to the question posed in the title of this research work. In fact, we obtain almost machine precision for the computation of the spectrum of $\mathcal{P}_n(f,g)$, for large n and only working with few really small matrices.

8.5 Summary

Bogoya et al. [131–133] have recently obtained the precise asymptotic expansion for the eigenvalues of a sequence of Toeplitz matrices $\{T_n(f)\}$, under suitable assumptions on the associated generating function f. In this research work we have shown numerical evidence that some of these assumptions can be relaxed and extended to the case of a sequence of preconditioned Toeplitz matrices $\{\mathcal{P}_n(f,g) = T_n^{-1}(g)T_n(f)\}$, for f trigonometric polynomial, g nonnegative, not identically zero trigonometric polynomial, $r = f/g$, and where the ratio $r(\cdot)$ plays the same role as $f(\cdot)$ in the nonpreconditioned case. The first order asymptotic term of the expansion has been also proven using purely linear algebra tools.

Moreover, based on the eigenvalue asymptotics, we devised an extrapolation algorithm for computing the eigenvalues of preconditioned banded symmetric Toeplitz matrices with a high level of accuracy, with a relatively low computational cost, and with potential application to the computation of the spectrum of differential operators. In fact, up to low rank corrections, matrices of the form $\mathcal{P}_n(f,g)$ appear in the context of the spectral approximation of differential operators in which a low rank correction of $T_n(g)$ is the mass matrix and a low rank correction of $T_n(f)$ is the stiffness matrix. We carried out also preliminary numerical tests confirming that the same kind of asymptotic expansion holds, at least in the context of the Isogeometric approximation of second order differential operators.

Therefore a plan for the future has to include:

- the theoretical proof of the asymptotic expansion in (8.1) for $\alpha > 1$;
- the analysis of the non monotone case and its relations with the study in [137] for the special case where $f(\theta) = 2 - 2\cos(\omega\theta)$, $\omega \geq 2$ integer, and $g(\theta) = 1$;
- the extension of the results by [128] to the preconditioned Toeplitz case and the study of its connection with the general expansion in (8.1);
- the extension of the numerical and theoretical study to a multidimensional, block setting, with special attention to the matrices coming from the approximation of elliptic differential operators.

Appendix

Theorem 8.1. *Let f, g be real-valued cosine trigonometric polynomials (RCTP) on $[0,\pi]$ with $M_g = \max g > 0$ and $m_g = \min g \geq 0$. If $r = \frac{f}{g}$ is monotone on $[0,\pi]$ then $\exists\, C > 0$ such that*

$$\left|\lambda_j(\mathcal{P}_n(f,g)) - r\left(\frac{j\pi}{n+1}\right)\right| \leq Ch \quad \forall\, j, \forall\, n, \tag{8.19}$$

where

- *$\mathcal{P}_n(f,g)$ is the "preconditioned" matrix $\mathcal{P}_n(f,g) = T_n^{-1}(g)T_n(f)$,*
- *$\lambda_1(\mathcal{P}_n(f,g)), \lambda_2(\mathcal{P}_n(f,g)), \ldots, \lambda_n(\mathcal{P}_n(f,g))$ are the eigenvalues of $\mathcal{P}_n(f,g)$, arranged in nondecreasing or nonincreasing order, depending on whether r is increasing or decreasing,*

- *$h = \frac{1}{n+1}$ and $\theta_{j,n} = \frac{j\pi}{n+1} = j\pi h$.*

Proof. For the sake of simplicity, we assume that r is nondecreasing (the other case has a similar proof).

Notice that the conditions on f and g imply that $T_n(g)$ is positive definite and, by setting $\sim$ the symbol representing similarity between matrices, we find $\mathcal{P}_n(f,g) \sim T_n^{-1/2}(g)T_n(f)T_n^{-1/2}(g)$ so we can order the eigenvalues of $\mathcal{P}_n(f,g)$ as follows

$$\lambda_1(\mathcal{P}_n(f,g)) \leq \lambda_2(\mathcal{P}_n(f,g)) \leq \cdots \leq \lambda_n(\mathcal{P}_n(f,g)).$$

We remark that

$$\begin{aligned} T_n(f) &= \tau_n(f) + H_n(f), \\ T_n(g) &= \tau_n(g) + H_n(g), \end{aligned} \tag{8.20}$$

where, for ψ RCTP of degree m and $Q = \left(\sqrt{\frac{2}{n+1}}\sin\left(\frac{ij\pi}{n+1}\right)\right)_{i,j=1}^{n}$, $\tau_n(\psi)$ is the following τ matrix [130] of size n generated by ψ

$$\tau_n(\psi) = Q \operatorname*{diag}_{1\leq j\leq n}\left(\psi\left(\frac{j\pi}{n+1}\right)\right)Q, \qquad Q = Q^T = Q^{-1},$$

and $H_n(\psi)$ is the Hankel matrix

$$H_n(\phi) = \begin{bmatrix} \hat{\psi}_2 & \hat{\psi}_3 & \cdots & \hat{\psi}_m & & & \\ \hat{\psi}_3 & & \iddots & & & & \\ \vdots & \iddots & & & & & \\ \hat{\psi}_m & & & & & & \\ & & & & & & \\ & & & & & & \hat{\psi}_m \\ & & & & & \iddots & \vdots \\ & & & & \iddots & & \hat{\psi}_3 \\ & & & \hat{\psi}_m & \cdots & \hat{\psi}_3 & \hat{\psi}_2 \end{bmatrix}.$$

with $\mathrm{rank}(H_n(\psi)) \leq 2(m-1)$.

Hence,

$$\begin{aligned} R_f &:= \mathrm{rank}(H_n(f)) \leq 2(\deg(f)-1), \\ R_g &:= \mathrm{rank}(H_n(g)) \leq 2(\deg(g)-1), \\ R_{f,g} &:= \max\{R_f, R_g\} \leq 2\left(\max\{\deg(f), \deg(g)\}-1\right). \end{aligned} \tag{8.21}$$

Let P_n^τ be the matrix $\tau_n^{-1}(g)\tau_n(f)$,

$$\begin{aligned} P_n^\tau &= Q\left(\operatorname*{diag}_{1\leq j\leq n}\left(g\left(\frac{j\pi}{n+1}\right)\right)\right)^{-1} QQ \operatorname*{diag}_{1\leq j\leq n}\left(f\left(\frac{j\pi}{n+1}\right)\right)Q \\ &= Q\operatorname*{diag}_{1\leq j\leq n}\left(\frac{f}{g}\left(\frac{j\pi}{n+1}\right)\right)Q \\ &= Q\operatorname*{diag}_{1\leq j\leq n}\left(r\left(\frac{j\pi}{n+1}\right)\right)Q. \end{aligned}$$

Hence, for $j = 1, \ldots, n$

$$\lambda_j(P_n^\tau) = r\left(\frac{j\pi}{n+1}\right). \tag{8.22}$$

By observing that $T_n^{-1}(g)T_n(f)$ is similar to $T_n^{-1/2}(g)T_n(f)T_n^{-1/2}(g)$, using the Min-Max spectral characterization for Hermitian matrices [129], fixed $j \in \{R_{f,g}+1, \ldots, n-R_{f,g}\}$ and $T \subset \mathbb{C}^n$, $\dim(T) = n+1-j$, we obtain

$$\begin{aligned} \lambda_j(\mathcal{P}_n(f,g)) &= \lambda_j\left(T_n^{-1}(g)T_n(f)\right) \\ &= \lambda_j\left(T_n^{-1/2}(g)T_n(f)T_n^{-1/2}(g)\right) \\ &= \max_{\dim(T)=n+1-j}\left(\min_{\substack{x\in T,\\ x\neq \underline{0}}}\left(\frac{x^*T_n^{-1/2}(g)T_n(f)T_n^{-1/2}(g)x}{x^*x}\right)\right) \\ &= \max_{\dim(T)=n+1-j}\left(\min_{\substack{x\in T,\\ x\neq 0\\ y=T_n^{-1/2}(g)x}}\left(\frac{y^*T_n(f)y}{y^*T_n(g)y}\right)\right) \\ &= \max_{\dim(\hat{T})=n+1-j}\left(\min_{\substack{y\in \hat{T},\\ y\neq \underline{0}}}\left(\frac{y^*T_n(f)y}{y^*T_n(g)y}\right)\right), \end{aligned} \tag{8.23}$$

because $T_n^{-1/2}(g)$ is a full rank matrix and, if $\dim(T) = n+1-j$, then $\hat{T} := \{y : y = T_n^{-1/2}(g)x,\ x \neq \underline{0}, x\ \in T\}$ is a new vector space having the same dimension $n+1-j$ as T.

Let F be the subspace of $\mathbb{C}^n$ generated by the union of the columns of matrices

$H_n(f)$ and $H_n(g)$. Because of the particular structure of the columns of Hankel matrices $H_n(f)$ and $H_n(g)$, we deduce

$$\dim(F) = \max\left\{\mathrm{rank}(H_n(g)), \mathrm{rank}(H_n(f))\right\} = R_{f,g},$$

so that

$$\dim(F^{\perp}) = n - R_{f,g}.$$

Let us define $W_{f,g} = \hat{T} \cap F^{\perp}$,

$$\begin{aligned} n+1-j \geq \dim(W_{f,g}) &\geq \max\{0, \dim(\hat{T}) + \dim(F^{\perp}) - n\} \\ &= n+1-j+n-R_{f,g}-n = n+1-(j+R_{f,g}), \end{aligned}$$

because $n+1-(j+R_{f,g}) \geq 1$ for $j \leq n-R_{f,g}$.The latter implies in particular that $W_{f,g} \neq \emptyset$. Thus, due the orthogonality, $\forall\, y \neq \underline{0} \in W_{f,g}$, we find

$$H_n(f)y = \underline{0}, \quad H_n(g)y = \underline{0},$$

so that

$$y^* H_n(f) y = 0, \quad y^* H_n(g) y = 0.$$

Hence, from (8.23)

$$\begin{aligned} \lambda_j(\mathcal{P}_n(f,g)) &= \max_{\dim(\hat{T})=n+1-j} \left(\min_{\substack{y\in\hat{T},\\ y\neq\underline{0}}} \left(\frac{y^*(\tau_n(f)+H_n(f))y}{y^*(\tau_n(g)+H_n(g))y} \right) \right) \\ &\leq \max_{\dim(\hat{T})=n+1-j} \left(\min_{\substack{y\in W_{f,g}\\ y\neq\underline{0}}} \left(\frac{y^*(\tau_n(f)+H_n(f))y}{y^*(\tau_n(g)+H_n(g))y} \right) \right) \\ &= \max_{\dim(\hat{T})=n+1-j} \left(\min_{\substack{y\in W_{f,g}\\ y\neq\underline{0}}} \left(\frac{y^*\tau_n(f)y}{y^*\tau_n(g)y} \right) \right) \\ &= \max_{\substack{W_{f,g}=\hat{T}\cap F^{\perp}\\ \dim(\hat{T})=n+1-j}} \left(\min_{\substack{y\in W_{f,g},\\ y\neq\underline{0}}} \left(\frac{y^*\tau_n(f)y}{y^*\tau_n(g)y} \right) \right) \\ &\leq \max_{n+1-j\geq\dim(\hat{W}_{f,g})\geq n+1-(j+R_{f,g})} \left(\min_{\substack{y\in\hat{W}_{f,g},\\ y\neq\underline{0}}} \left(\frac{y^*\tau_n(f)y}{y^*\tau_n(g)y} \right) \right) \end{aligned} \tag{8.24}$$

$$
\begin{aligned}
&= \max_{n+1-j \geq \dim(\hat{W}) \geq n+1-(j+R_{f,g})} \left(\min_{\substack{y \in \hat{W}_{f,g}, \\ y \neq \underline{0} \\ x = \tau_n^{1/2}(g) y}} \left(\frac{x^* \tau_n^{-1/2}(g) \tau_n(f) \tau_n^{-1/2}(g) x}{x^* x} \right) \right) \\
&= \max\{\lambda_j(P_n^\tau), \lambda_{j+1}(P_n^\tau), \ldots, \lambda_{j+R_{f,g}}(P_n^\tau)\} \\
&= \lambda_{j+R_{f,g}}(P_n^\tau).
\end{aligned}
$$

By fixing $j \in \{R_{f,g}+1, \ldots, n-R_{f,g}\}$ and $T \subset \mathbb{C}^n$, $\dim(T) = j$, analogously we obtain

$$
\begin{aligned}
\lambda_j(\mathcal{P}_n(f,g)) &= \min_{\dim(T)=j} \left(\max_{\substack{x \in T, \\ x \neq \underline{0}}} \left(\frac{x^* T_n^{-1/2}(g) T_n(f) T_n^{-1/2}(g) x}{x^* x} \right) \right) \\
&= \min_{\dim(T)=j} \left(\max_{\substack{x \in T, \\ x \neq \underline{0} \\ y = T_n^{-1/2}(g) x}} \left(\frac{y^* T_n(f) y}{y^* T_n(g) y} \right) \right) \\
&= \min_{\dim(\hat{T})=j} \left(\max_{\substack{y \in \hat{T}, \\ y \neq \underline{0}}} \left(\frac{y^* T_n(f) y}{y^* T_n(g) y} \right) \right) \\
&= \min_{\dim(\hat{T})=j} \left(\max_{\substack{y \in \hat{T}, \\ y \neq \underline{0}}} \left(\frac{y^*(\tau_n(f) + H_n(f)) y}{y^*(\tau_n(g) + H_n(g)) y} \right) \right).
\end{aligned} \tag{8.25}
$$

Let us define $W_{f,g} = \hat{T} \cap F^\perp$,

$$j \geq \dim(W_{f,g}) \geq \max\{0, \dim(\hat{T}) + \dim(F^\perp) - n\} = j + n - R_{f,g} - n = j - R_{f,g},$$

because $j - R_{f,g} \geq 1$ for $j \geq R_{f,g}+1$. The latter implies in particular that $W_{f,g} \neq \emptyset$, and hence, due the orthogonality, $\forall\, y \neq \underline{0} \in W_{f,g}$, we have

$$H_n(f) y = \underline{0}, \quad H_n(g) y = \underline{0},$$

and therefore

$$y^* H_n(f) y = 0, \quad y^* H_n(g) y = 0.$$

Thus, from (8.25)

$$\begin{aligned}
\lambda_j(\mathcal{P}_n(f,g)) &\geq \min_{\dim(\hat{T})=j}\left(\max_{\substack{y\in W_{f,g},\\ y\neq \underline{0}}}\left(\frac{y^*(\tau_n(f)+H_n(f))y}{y^*(\tau_n(g)+H_n(g))y}\right)\right)\\
&= \min_{\dim(\hat{T})=j}\left(\max_{\substack{y\in W_{f,g},\\ y\neq \underline{0}}}\left(\frac{y^*\tau_n(f)y}{y^*\tau_n(g)y}\right)\right)\\
&= \min_{\substack{W_{f,g}=\hat{T}\cap F^{\perp}\\ \dim(\hat{T})=j}}\left(\max_{\substack{y\in W_{f,g},\\ x\neq \underline{0}}}\left(\frac{y^*\tau_n(f)y}{y^*\tau_n(g)y}\right)\right) \qquad (8.26)\\
&\geq \min_{j\geq \dim(\hat{W}_{f,g})\geq j-R_{f,g}}\left(\max_{\substack{y\in W_{f,g},\\ y\neq \underline{0}}}\left(\frac{y^*\tau_n(f)y}{y^*\tau_n(g)y}\right)\right)\\
&= \min\{\lambda_j(P_n^\tau),\lambda_{j-1}(P_n^\tau),\ldots,\lambda_{j-R_{f,g}}(P_n^\tau)\}\\
&= \lambda_{j-R_{f,g}}(P_n^\tau).
\end{aligned}$$

By exploiting the previous inequality, relations (8.22) and (8.24), we obtain for $j = R_{f,g}+1,\ldots,n-R_{f,g}$

$$r\left(\frac{(j-s)\pi}{n+1}\right) = \lambda_{j-s}(P_n^\tau) \leq \lambda_j(\mathcal{P}_n(f,g)) \leq \lambda_{j+s}(P_n^\tau) = r\left(\frac{(j+s)\pi}{n+1}\right), \qquad (8.27)$$

where $s = R_{f,g}$.

The function r is a RCTP on $[0,\pi]$ and a monotone increasing function so we have, $\forall\, n$ and $\forall\, j = s+1,\ldots,n-s$,

$$\lambda_j(\mathcal{P}_n(f,g))-r\left(\frac{j\pi}{n+1}\right) \leq r\left(\frac{(j+s)\pi}{n+1}\right)-r\left(\frac{j\pi}{n+1}\right) = r'(\bar{\theta})\frac{s\pi}{n+1} \leq ||r'||_\infty s\pi h, \qquad (8.28)$$

with $\bar{\theta}\in(\frac{j\pi}{n+1},\frac{(j+s)\pi}{n+1})$ and

$$\lambda_j(\mathcal{P}_n(f,g)) - r\left(\frac{j\pi}{n+1}\right) \geq r\left(\frac{(j-s)\pi}{n+1}\right) - r\left(\frac{j\pi}{n+1}\right) \geq -||r'||_\infty s\pi h. \qquad (8.29)$$

By setting $C = ||r'||_\infty s\pi$, for $s+1 \leq j \leq n-s$, we obtain

$$\left|\lambda_j(\mathcal{P}_n(f,g)) - r\left(\frac{j\pi}{n+1}\right)\right| \leq Ch. \qquad (8.30)$$

Furthermore, from [136] $\forall\, j = 1,\ldots,n$, we know that

$$m_r \leq \lambda_j(\mathcal{P}_n(f,g)) \leq M_r,$$

where

$$m_r = \min_{\theta\in(0,\pi)} r(\theta); \quad M_R = \max_{\theta\in(0,\pi)} r(\theta),$$

with strict inequalities that is $m_r < \lambda_j(\mathcal{P}_n(f,g)) < M_r$ if $m_r < M_r$, while the case $m_r = M_r$ is in fact trivial. Hence for $n-s < j \le n$

$$\left|r\left(\frac{j\pi}{n+1}\right) - \lambda_j(\mathcal{P}_n(f,g))\right| \le \left|r\left(\frac{j\pi}{n+1}\right) - r\left(\frac{n\pi}{n+1}\right)\right| \le \left|r'\left(\bar{\theta}\right)\right| \left|\frac{(n-j)\pi}{n+1}\right|,$$

where $\bar{\theta} \in (\frac{j\pi}{n+1}, \frac{n\pi}{n+1})$. If $n-s < j \le n$ then $|n-j| < s$, so that

$$\left|r\left(\frac{j\pi}{n+1}\right) - \lambda_j(\mathcal{P}_n(f,g))\right| \le ||r'||_\infty s\pi h = Ch.$$

For $1 \le j < s+1$

$$\left|r\left(\frac{j\pi}{n+1}\right) - \lambda_j(\mathcal{P}_n(f,g))\right| \le \left|r\left(\frac{j\pi}{n+1}\right) - r\left(\frac{\pi}{n+1}\right)\right| \le \left|r'\left(\bar{\theta}\right)\right| \left|\frac{(j-1)\pi}{n+1}\right|,$$

where $\bar{\theta} \in (\frac{\pi}{n+1}, \frac{j\pi}{n+1})$. If $1 \le j < s+1$ then $|j-1| < s$, so

$$\left|r\left(\frac{j\pi}{n+1}\right) - \lambda_j(\mathcal{P}_n(f,g))\right| \le ||r'||_\infty s\pi h = Ch.$$

Hence

$$\left|\lambda_j(\mathcal{P}_n(f,g)) - r\left(\frac{j\pi}{n+1}\right)\right| \le Ch \quad \forall j \forall n.$$

□

Here we present a second proof of the previous theorem.

Proof. We adopt the very same notation used for the first proof. First we notice that the low rank matrices $H_n(f)$ and $H_n(g)$ are also Hermitian matrices because $T_n(f)$, $T_n(g)$, $\tau_n(f)$, and $\tau_n(g)$ are Hermitian matrices. Let $\mathbf{x}_i$ and $\lambda_i\left(\mathcal{P}_n(f,g)\right)$ be a pair eigenvector and eigenvalue of $\mathcal{P}_n(f,g)$. Then we can write

$$\mathcal{P}_n(f,g)\mathbf{x}_i = \lambda_i\left(\mathcal{P}_n(f,g)\right)\mathbf{x}_i.$$

By multiplying the previous equation from the left by the matrix $T_n(g) = \tau_n(g) + H_n(g)$, we obtain

$$\left(\tau_n(f) + H_n(f)\right)\mathbf{x}_i = \lambda_i\left(\mathcal{P}_n(f,g)\right)\left(\tau_n(g) + H_n(g)\right)\mathbf{x}_i,$$

which is equivalent to

$$(\tau_n(f) + H_n(f) - \lambda_i(\mathcal{P}_n(f,g))\, H_n(g))\, \mathbf{x}_i = \lambda_i(\mathcal{P}_n(f,g))\, \tau_n(g)\mathbf{x}_i.$$

Finally, by setting $\mathbf{y}_i = \tau_n^{1/2}(g)\mathbf{x}_i$ and by multiplying from the left by the matrix $\tau_n^{-1/2}(g)$, we have

$$\tau^{-1/2}(g)\,(\tau_n(f) + H_n(f) - \lambda_i(\mathcal{P}_n(f,g))\, H_n(g))\, \tau^{-1/2}(g)\mathbf{y}_i = \lambda_i(\mathcal{P}_n(f,g))\, \mathbf{y}_i.$$

(8.31)

Equation (8.31) tells us that $\lambda_i(\mathcal{P}_n(f,g))$ is also the eigenvalue of

$$\tau_n^{-1/2}(g)\,(\tau_n(f) + H_n(f) - \lambda_i(\mathcal{P}_n(f,g))\, H_n(g))\, \tau_n^{-1/2}(g).$$

We can write

$$\tau_n^{-1/2}(g)\,(\tau_n(f) + H_n(f) - \lambda_i(\mathcal{P}_n(f,g))\, H_n(g))\, \tau_n^{-1/2}(g)$$

as

$$\begin{aligned}\tau_n^{-1/2}(g)\tau_n(f)\tau_n^{-1/2}(g) + \tau_n^{-1/2}(g)\,(H_n(f) - \lambda_i(\mathcal{P}_n(f,g))\, H_n(g))\, \tau_n^{-1/2}(g) &= \\ = \tau_n(f/g) + \tau_n^{-1/2}(g)\,(H_n(f) - \lambda_i(\mathcal{P}_n(f,g))\, H_n(g))\, \tau_n^{-1/2}(g)\,. &\end{aligned} \tag{8.32}$$

Notice that the rank of any linear combination of $H_n(f)$ and $H_n(g)$ is $R_{f,g} = \max\{\mathrm{rank}(H_n(f)), \mathrm{rank}(H_n(g))\}$ and the argument is the special Hankel structure of $H_n(f)$ and $H_n(g)$. As a conclusion, from the expression above, using the MinMax characterization and the interlacing theorem for Hermitian matrices, we write

$$\lambda_{i-R_{f,g}}(\tau_n(f/g)) \leq \lambda_i(\mathcal{P}_n(f,g)) \leq \lambda_{i+R_{f,g}}(\tau_n(f/g)), \tag{8.33}$$

where $i \in \{R_{f,g}+1, \cdots, n-R_{f,g}\}$, which leads again to the proof of Theorem 8.1. □

Remark 8.2. With regard to Theorem 8.1, the case where r is bounded and non-monotone is even easier. If we consider $\hat{r}$, the monotone nondecreasing rearrangement of r on $[0,\pi]$, taking into account that the derivative of r has at most a finite number S of sign changes, we deduce that $\hat{r}$ is Lipschitz continuous and its Lipschitz constant is bounded by $\|r'\|_\infty$ (notice that $\hat{r}$ is not necessarily continuously differentiable, but the derivative of $\hat{r}$ has at most S points of discontinuity). Furthermore, the eigenvalues of $\tau_n(r)$ are exactly given

$$r\left(\frac{j\pi}{n+1}\right)$$

so that, by ordering these values nondecreasingly, we deduce that they coincide with $\hat{r}(x_{j,h})$, with $x_{j,h}$ of the form $\frac{j\pi}{n+1}(1+o(1))$. With these premises, the proof follows exactly the same steps as in Theorem 8.1, using the MinMax characterization and the interlacing theorem for Hermitian matrices.

Chapter 9

A robust numerical method to compute eigenvalues of banded symmetric Toeplitz matrices

A robust numerical methods is constructed to extrapolate and interpolate the eigenvalues of banded symmetric Toeplitz matrices when the matrix size is large. The distribution of eigenvalues of a banded symmetric Toeplitz matrix $(T_n(f))$ in $[0,\pi]$ get closer to the distributed value of $f(\theta)$ for $\theta = \frac{j\pi}{n+1}$, $f(\cdot)$ being the symbol of banded symmetric Toeplitz matrix, as the size n of banded symmetric Toeplitz increases. In general the the difference $\lambda_j(T_n(f)) - f(\theta_j)$ is not zero. In the following we present an algorithm to choose $t_{j,n}$ such that $\lambda_j(T_n(f)) = f(t_{j,n}\theta_j)$. With the help of cubic-spline fitting, we show how it is possible to extrapolate and interpolate the eigenvalues for large matrix sizes. A strategy is also presented in order to choose the best interpolated and extrapolated eigenvalues.

9.1 Introduction

In Chapter 8, The expansion [131–133] of $\lambda_j(T_n(f))$ is given (assume $g(x) = 1$ and $r(x) = f(x)$) i.e.

$$\lambda_j(T_n(f)) = f(\theta_{j,n}) + \sum_{k=1}^{\alpha} c_k(\theta_{j,n})h^k + E_{j,n,\alpha}, \tag{9.1}$$

where:

- the eigenvalues of $T_n(f)$ are arranged in nondecreasing or nonincreasing order, depending on whether r is increasing or decreasing;
- $\{c_k\}_{k=1,2,\ldots}$ is a sequence of functions from $[0,\pi]$ to $\mathbb{R}$ which depends only on f;
- $h = \frac{1}{n+1}$ and $\theta_{j,n} = \frac{j\pi}{n+1} = j\pi h$;
- $E_{j,n,\alpha} = O(h^{\alpha+1})$ is the remainder (the error), which satisfies the inequality $|E_{j,n,\alpha}| \leq C_\alpha h^{\alpha+1}$ for some constant C_α depending only on α and f.

Detailed discussion concerning the implementation of (9.1) can be found in Chapter 8. We are looking for parameter $t_{j,n}$ such that

$$\lambda_j(T_n(f)) = f(t_{j,n}\,\theta_{j,n})\,. \tag{9.2}$$

The cases when the symbols of banded symmetric Toeplitz matrices are either monotonically increasing or decreasing are considered.

Notice that when the symbol $f(\cdot)$ is either increasing or decreasing the inversion of $f(\cdot)$ is possible. If the analytical expression of $f^{-1}(\cdot)$ is available then the computation of $t_{j,n}$ is trivial

$$t_{j,n} = f^{-1}\left(\frac{\lambda_j(T_n(f))}{\theta_j}\right)\,. \tag{9.3}$$

9.2 Algorithm

Suppose $T_n(f)$ is a Toeplitz matrix with monotonically increasing or decreasing symbol $f(\cdot)$. We define $n_i = 2^{i-1}(n_1+1)-1$, $h_i = \frac{1}{n_i+1}$ and $\theta_{j_i,n_i} = \frac{j_i\,\pi}{n_i+1} = j\,\pi\,h_i$.

If we equate

$$\begin{aligned}
&\theta_{j_1,n_1} = \theta_{j_i,n_i}\\
&\frac{j_1\,\pi}{1+n_1} = \frac{j_i\,\pi}{1+n_i} = \frac{j_i\,\pi}{1+2^{i-1}(n_1+1)-1} = \frac{j_i\,\pi}{2^{i-1}(n_1+1)}\\
&j_i = j_1\,2^{i-1}\,.
\end{aligned}$$

In other words we can say

$$\theta_{j_1\,2^{i-1},n_i} = \theta_{j_1,n_1}\,.$$

We can compute the eigenvalues of small size Toeplitz matrix $T_n(f)$ and looking for numerical approximation of the eigenvalues of much larger matrix. Suppose it is given that

$$\lambda_{j_i,n_i}(T_{n_i}(f))\,,\quad \text{for } j_i = 1,2,\cdots,n_i\,,\quad \text{and for } i = 1,2,\cdots,m_1\,.$$

For some $m_2 > m_1$, it is aimed to approximate

$$\lambda_{j_{m_2},n_{m_2}}(T_{n_{m_2}}(f))\,,\quad \text{for } j_{m_2} = 1,2,\cdots,n_{m_2}\,.$$

But first we explain how to compute

$$\lambda_{j_1\,2^{m_2-1},n_{m_2}}(T_{n_{m_2}}(f))\,,\quad \text{for } j_1 = 1,2,\cdots,n_1\,.$$

	θ_1	θ_2	$\cdots$	θ_{n_1}
h_1	λ_{1,n_1}	λ_{2,n_1}	$\cdots$	λ_{n_1,n_1}
h_2	$\lambda_{2(1),n_2}$	$\lambda_{2(2),n_2}$	$\cdots$	$\lambda_{2(n_1),n_2}$
h_3	$\lambda_{4(1),n_3}$	$\lambda_{4(2),n_3}$	$\cdots$	$\lambda_{4(n_1),n_3}$
$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\vdots$
h_{m_1}	$\lambda_{2^{m_1-1}(1),n_{m_1}}$	$\lambda_{2^{m_1-1}(2),n_{m_1}}$	$\cdots$	$\lambda_{2^{m_1-1}(n_1),n_{m_1}}$

Table 9.1: Seclection of eigenvalues according to θ_i.

	θ_1	θ_2	$\cdots$	θ_{n_1}
h_1	t_{1,n_1}	t_{2,n_1}	$\cdots$	t_{n_1,n_1}
h_2	$t_{2(1),n_2}$	$t_{2(2),n_2}$	$\cdots$	$t_{2(n_1),n_2}$
h_3	$t_{4(1),n_3}$	$t_{4(2),n_3}$	$\cdots$	$t_{4(n_1),n_3}$
$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\vdots$
h_{m_1}	$t_{2^{m_1-1}(1),n_{m_1}}$	$t_{2^{m_1-1}(2),n_{m_1}}$	$\cdots$	$t_{2^{m_1-1}(n_1),n_{m_1}}$

TABLE 9.2: Parameters t_{j_i,n_i} according to θ_i.

Arrangements of λ_{j_i,n_i}'s and t_{j_i,n_i}'s are shown in Tables 9.1 and 9.2 respectively. Each column in Tables 9.1 and 9.2 corresponds to a fixed value of θ. The computation of t_{j_i,n_i} can be performed according to (9.3). In case if there is no empirical formula to compute the inverse of bijective symbol $f(\cdot)$ one may us inverse interpolation by tabulating the values of $f(\cdot)$.

To extrapolate the values of parameters t_{j_i,n_i} from Table 9.2 we can fit Lagrange polynomial in each column of Table 9.2 by taking h_i as independent variables. Idea is presented in Table 9.3.

h_1	t_{i,n_1}
h_2	$t_{2\,i,n_2}$
h_3	$t_{4\,i,n_3}$
$\vdots$	$\vdots$
h_{m_1}	$t_{2^{m_1-1}\,i,n_{m_1}}$
	↓
	(Lagrange extrapolation)
	↓
h_{m_2}	$t_{2^{m_2-1}\,i,n_{m_2}}$
	↓
	$\tilde{\lambda}_{2^{m_2-1}\,i,n_{m_2}} = f\left(t_{2^{m_2-1}\,i,n_{m_2}}\theta_i\right)$

TABLE 9.3: For fixed θ_i: $t_{2^{m_2-1}\,i,n_{m_2}}$ is an extrapolated value for h_{m_2} by using Lagrange polynomial.

Once we have extrapolate the value of $t_{2^{m_2-1}i,n_{m_2}}$ for h_{m_2}, it is an easy task to approximate $\lambda_{2^{m_2-1}i,n_{m_2}}$. The approximation is

$$\tilde{\lambda}_{2^{m_2-1}i,n_{m_2}} = f\left(t_{2^{m_2-1}i,n_{m_2}}\theta_i\right).$$

9.2.1 Best approximation of $\lambda_{2^{m_2-1}i,n_{m_2}}$

We proposed an algorithm to get best approximation value of $\lambda_{2^{m_2-1}i,n_{m_2}}$. For each fixed θ_i we have set of paired data of size m_1 as it is shown in Table 9.3. According to Table 9.5 first we use m_1 paired values to approximate $\lambda_{2^{m_2-1}i,n_{m_2}}$ and call the approximated value $\left(\tilde{\lambda}_{2^{m_2-1}i,n_{m_2}}\right)_1$. Next use $m_1 - 1$ paired values to get second approximation $\left(\tilde{\lambda}_{2^{m_2-1}i,n_{m_2}}\right)_2$ and so on. Finally we use only two paired values to get approximation $\left(\tilde{\lambda}_{2^{m_2-1}i,n_{m_2}}\right)_{m_1-1}$. We can compute m_1-2 forward differences from $m_1 - 1$ data values and that are shown in Table 9.5. We find the index of minimum positive forward difference and declare the approximated eigenvalue best that corresponds to said index. Algorithm to find best value is given in Algorithm 1 (Best value finder algorithm).

9.2.2 Extrapolation and interpolation of eigenvalues

As we have discussed for each θ_i for $i = 1, 2, \cdots, n_1$ we can choose best approximation to eigenvalue $\tilde{\lambda}_{2^{m_2-1}i,n_{m_2}}$. From our algorithm we also have the information about $t_{2^{m_2-1}i,n_{m_2}}$ that corresponds to best selected approximated eigenvalues. We fit piecewise cubic spline in paired data $\cdot(\theta_i, t_{2^{m_2-1}i,n_{m_2}})$ which is also shown in Table 9.6. Once we have constructed piecewise cubic spline we extrapolate and interpolate the value of $t_{2^{m_2-1}i,n_{m_2}}$ for $\theta_{j_{m_2},n_{m_2}}$ for $j_{m_2} = 1, 2, \cdots, n_{m_2}$. From above information the computation of $\tilde{\lambda}_{j_{m_2},n_{m_2}}$ for $j_{m_2} = 1, 2, \cdots, n_{m_2}$ is straightforward. Algorithm 2 (extrapolation and interpolation of eigenvalues) describes the detailed implementation of extrapolation and interpolation of eigenvalues.

9.2.3 Best extrapolation and interpolation of eigenvalues

We have m_1 data set of eigenvalues of sizes $n_1, n_2, \cdots, n_{m_1}$ respectively. If we divide successively these data sets of eigenvalues in the way

$$\begin{array}{cl}
(n_1, n_2, \cdots, n_{m_1}) & m_1 \quad \text{data sets} \\
(n_2, n_3, \cdots, n_{m_1}) & m_1 - 1 \text{ data sets} \\
(n_3, n_4, \cdots, n_{m_1}) & m_1 - 2 \text{ data sets} \\
\vdots & \\
(n_{m_1-1}, n_{m_1}) & 2 \quad \text{data sets}\,,
\end{array}$$

we obtain $m_1 - 1$ data sets of eigenvalues. By applying the procedure of Section 9.2.2 we can compute best approximation $\left(\tilde{\lambda}_{j_{m_2}, n_{m_2}}\right)_q$ for $j_{m_2} = 1, 2, \cdots, n_{m_2}$ and $q = 1, 2, 3, \cdots, m_1 - 1$. Here q-index varies over the $m_1 - 1$ data set (see Table 9.7). To select globally best eigenvalues of these $m_1 - 1$ data sets of eigenvalues we again use the strategy of forward difference which is describe in Algorithm 1. Algorithm 3 execute the all the procedures to get globally best eigenvalues.

Paired data sets	Lagrange extrapolation	Approximated eigenvalues values
$(h_1, t_{i,n_1}), \cdots, (h_{m_1}, t_{2^{m_1-1}\,i, n_{m_1}})$	→	$\left(\tilde{\lambda}_{2^{m_2-1}\,i, n_{m_2}}\right)_1$
$(h_2, t_{2\,i,n_1}), \cdots, (h_{m_1}, t_{2^{m_1-1}\,i, n_{m_1}})$	→	$\left(\tilde{\lambda}_{2^{m_2-1}\,i, n_{m_2}}\right)_2$
$(h_3, t_{4\,i,n_1}), \cdots, (h_{m_1}, t_{2^{m_1-1}\,i, n_{m_1}})$	→	$\left(\tilde{\lambda}_{2^{m_2-1}\,i, n_{m_2}}\right)_3$
$\vdots$	$\vdots$	$\vdots$
$(h_{m_1-1}, t_{2^{m_1-2}\,i,n_1}), \cdots, (h_{m_1}, t_{2^{m_1-1}\,i, n_{m_1}})$	→	$\left(\tilde{\lambda}_{2^{m_2-1}\,i, n_{m_2}}\right)_{m_1-1}$

TABLE 9.4: For fixed θ_i: $m_1 - 1$ approximations of $\lambda_{2^{m_2-1}\,i, n_{m_2}}$.

Approximated eigenvalues	Forward differences
$\left(\tilde{\lambda}_{2^{m_2-1}\,i,n_{m_2}}\right)_1$	$\left(\tilde{\lambda}_{2^{m_2-1}\,i,n_{m_2}}\right)_2$ - $\left(\tilde{\lambda}_{2^{m_2-1}\,i,n_{m_2}}\right)_1$
$\left(\tilde{\lambda}_{2^{m_2-1}\,i,n_{m_2}}\right)_2$	$\left(\tilde{\lambda}_{2^{m_2-1}\,i,n_{m_2}}\right)_3$ - $\left(\tilde{\lambda}_{2^{m_2-1}\,i,n_{m_2}}\right)_2$
$\left(\tilde{\lambda}_{2^{m_2-1}\,i,n_{m_2}}\right)_3$	$\left(\tilde{\lambda}_{2^{m_2-1}\,i,n_{m_2}}\right)_4$ - $\left(\tilde{\lambda}_{2^{m_2-1}\,i,n_{m_2}}\right)_3$
$\vdots$	$\vdots$
$\left(\tilde{\lambda}_{2^{m_2-1}\,i,n_{m_2}}\right)_{m_1-2}$	$\left(\tilde{\lambda}_{2^{m_2-1}\,i,n_{m_2}}\right)_{m_1-1}$ - $\left(\tilde{\lambda}_{2^{m_2-1}\,i,n_{m_2}}\right)_{m_1-2}$

Table 9.5: Computations of forward differences.

θ	$t_{2^{m_2-1}\,i,n_{m_2}}$
θ_1	$t_{2^{m_2-1}\,1,n_{m_2}}$
θ_2	$t_{2^{m_2-1}\,2,n_{m_2}}$
θ_3	$t_{2^{m_2-1}\,3,n_{m_2}}$
$\vdots$	$\vdots$
θ_{n_1}	$t_{2^{m_2-1}\,n_1,n_{m_2}}$

Table 9.6: Best selected $t_{2^{m_2-1}\,i,n_{m_2}}$ for θ_i.

Data set	Best approximated eigenvalues
$(n_1, n_2, \cdots, n_{m_1})$	$\left(\tilde{\lambda}_{j_{m_2},n_{m_2}}\right)_1$ for $j_{m_2} = 1, 2, \cdots, n_{m_2}$
$(n_2, n_3, \cdots, n_{m_1})$	$\left(\tilde{\lambda}_{j_{m_2},n_{m_2}}\right)_2$ for $j_{m_2} = 1, 2, \cdots, n_{m_2}$
$(n_3, n_4, \cdots, n_{m_1})$	$\left(\tilde{\lambda}_{j_{m_2},n_{m_2}}\right)_3$ for $j_{m_2} = 1, 2, \cdots, n_{m_2}$
$\vdots$	$\vdots$
(n_{m_1-1}, n_{m_1})	$\left(\tilde{\lambda}_{j_{m_2},n_{m_2}}\right)_{m_1-1}$ for $j_{m_2} = 1, 2, \cdots, n_{m_2}$

Table 9.7: Computation of $\tilde{\lambda}_{j_{m_2},n_{m_2}}$ for different data sets.

9.3 Numerical testing

The central finite difference approximation of d^{2m}/dx^{2m} can be expressed in terms of symmetric banded Toeplitz matrix $T_n(g_m)$ and $g_m(\theta) = (2-2cos(\theta))^m$. We can notice that g_m is a monotonic function for $\theta \in [0, \pi]$ and its inverse is $g_m^{-1}(x) = cos^{-1}\left(1 - x^{1/m}/2\right)$.

Algorithm 1 Main program for extrapolation and interpolation of eigenvalues

```
procedure MAIN_PROGRAM($n_1$,$s$,$q$)
    for $j$ from 1 to $s$ do
        $n_i \leftarrow 2^{(j-1)}(n_1+1)-1$
        $h_j \leftarrow \frac{1}{1+n_j}$
    end for
    for $j$ from 1 to $s$ do
        for $i$ from 1 to $n_j$ do
            $\lambda_{i,j} \leftarrow$ eigenvalue $\left(T_{n_j(f)}\right)$
        end for
    end for
    for $i$ from 1 to $s-1$ do
        $\bar{\lambda} \leftarrow$ EXTRAPOLATION_INTERPOLATION_OF_EIGENVALUES($\lambda, n_{i:s}, s-i+1, q-i+1$)
    end for
    $[\hat{\lambda}, \sim] \leftarrow$ BEST_VALUE_FINDER($\bar{\lambda}$)
    return $\hat{\lambda}$                                ▷ Return eigenvalues of $T_{n_q}(f)$
end procedure
```

Algorithm 2 Best value finder algorithm

```
procedure BEST_VALUE_FINDER($B$) ▷ Find best value in each row of matrix $B$
    $[n,m] \leftarrow$ size($B$)                ▷ Find the dimension of matrix $B$
    $v \leftarrow$ zeros($n$, 1)                ▷ Define vector $v$ of dimension $n \times 1$
    $A \leftarrow$ diff($B'$)$'$ ▷ "diff" is a command in Matlab to compute the difference and $(\cdot)'$ means transpose
    for $i$ from 1 to $n$ do
        $J \leftarrow$ find($A(i,:) < 0$)    ▷ Find the indexes of elements of $i$th row of $A$ which are negative
        $A(i,J) \leftarrow \infty$           ▷ Replace negative values of $A$ with very big value
        $[\sim, I] \leftarrow$ min($A(i,:)$)    ▷ Find the index of element of $i$th row of $A$ which has minimum value
        index($i$) $\leftarrow I$
        $v(i) \leftarrow B(i,$ index($i$))    ▷ Save minimum value element of $i$th row of matrix $B$ in $v(i)$
    end for
    return $v$, index
end procedure
```

Algorithm 3 Extrapolation and interpolation of eigenvalues

```
procedure EXTRAPOLATION_INTERPOLATION_OF_EIGENVALUES($\lambda$,$n$,$s$,$q$)
    for $j$ from 1 to $s$ do          ▷ $\lambda = \{\lambda_{i,n_j} | i = 1, 2, \cdots, n_j \& j = 1, 2, \cdots, s\}$
        $h_j \leftarrow \frac{1}{1+n_j}$          ▷ $n = [n_1, n_2, \cdots, n_s]$
    end for
    $\bar{h} \leftarrow [h_1, h_2, \cdots, h_s]$
    $n_q \leftarrow 2^{(q-1)}(n_1 + 1) - 1$
    $h_q \leftarrow \frac{1}{1+n_q}$
    for $i$ from 1 to $n_1$ do
        $\theta_i \leftarrow i\, h_1\, \pi$
    end for
    for $i$ from 1 to $n_q$ do
        $\hat{\theta}_i \leftarrow i\, h_q\, \pi$
    end for
    for $j$ from 1 to $s$ do
        for $i$ from 1 to $n_1$ do
            $\bar{\lambda}_{i,j} \leftarrow \lambda_{2^{(j-1)}i,n_j}$
            $f(t_{i,j}\theta_i) = \bar{\lambda}_{i,j}$          ▷ Solve $f(t_{i,j}\theta_i) = \bar{\lambda}_{i,j}$ to find the value of $t_{i,j}$
        end for
    end for
    for $j$ from 1 to $s - 1$ do
        for $i$ from 1 to $n_1$ do          ▷ Cubic spline for extrapolation
            $\bar{t}_{i,j} \leftarrow \text{spline}(\bar{h}(j : s), t(i, j : s), h_q)$          ▷ $\bar{h}(j : s) = [h_j, h_{j+1}, \cdots, h_s]$
            $\bar{\lambda}_{i,j} \leftarrow f(\bar{t}_{i,j}\theta_i)$
        end for
    end for
    if $s - 1 > 1$ then
        $B \leftarrow [\bar{\lambda}_{i,j}]_{n_1 \times (s-1)}$
        $[\sim, \text{index}] \leftarrow$ BEST_VALUE_FINDER($B$)          ▷ Call algorithm 2
        for $i$ from 1 to $n_1$ do
            $\tilde{t}(i) \leftarrow \bar{t}(i, index(i))$
        end for
    else
        $\tilde{t} \leftarrow \bar{t}$
    end if
    $\hat{t} \leftarrow \text{spline}(\theta, \tilde{t}, \hat{\theta})$
    $\hat{\lambda} \leftarrow f(\hat{t}\,\hat{\theta})$          ▷ Cubic spline for interpolation and extrapolation
    return $\hat{\lambda}$          ▷ Return eigenvalues of $T_{n_q}(f)$
end procedure
```

The associated matrix with central difference approximation of $\frac{d^4}{dx^4}$ is

$$T(g_2) = \begin{bmatrix} 6 & -4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -4 & 6 & -4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 6- & 4 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 6 & -4 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -4 & 6 & -4 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -4 & 6 & -4 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -4 & 6 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -4 & 6 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -4 & 6 & -4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -4 & 6 \end{bmatrix}_{10\times 10}$$

We can compute

$$t_{j_q,n_q} = \frac{cos^{-1}\left(1 - \frac{\sqrt{\lambda_{j_q,n_q}}}{2}\right)}{\theta_{j_q,n_q}},$$

where $q = 1, 2, \cdots, s$.

In Figures 9.1, 9.2, 9.3 and 9.4 we can see absolute error plots before (left) and after (right) the implementation of Algorithm 1. Full extrapolation and interpolation with best selection of approximated eigenvalues are shown in Figures 9.5 and 9.6 for different values of m_1. Figures 9.1, 9.2, 9.3 and 9.4 clearly tell that the performance of our proposed algorithm is comparability better.

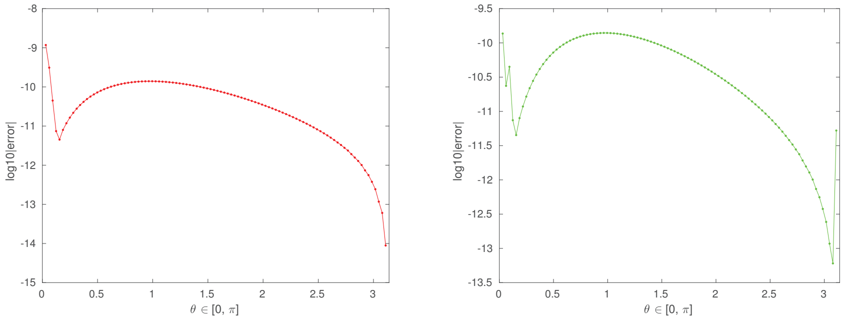


FIGURE 9.1: Absolute error in n_1 eigenvalues out of $n_8 = 12927$: error before (left), error after (right), $m_1 = 4$, $n_8 = 12927$ and $n_1 = 100$.

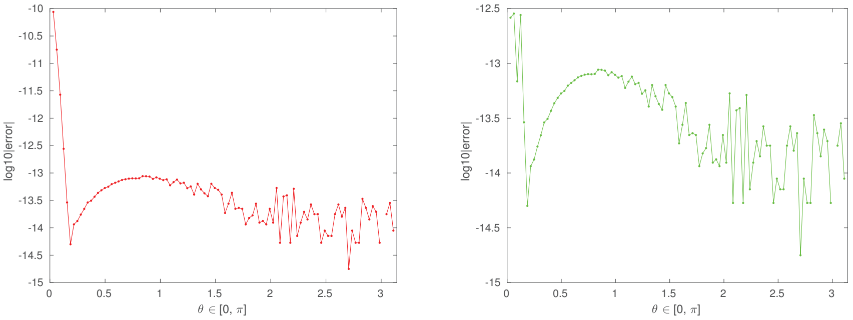


FIGURE 9.2: Absolute error in n_1 eigenvalues out of $n_8 = 12927$: error before (left), error after (right), $m_1 = 5$, $n_8 = 12927$ and $n_1 = 100$.

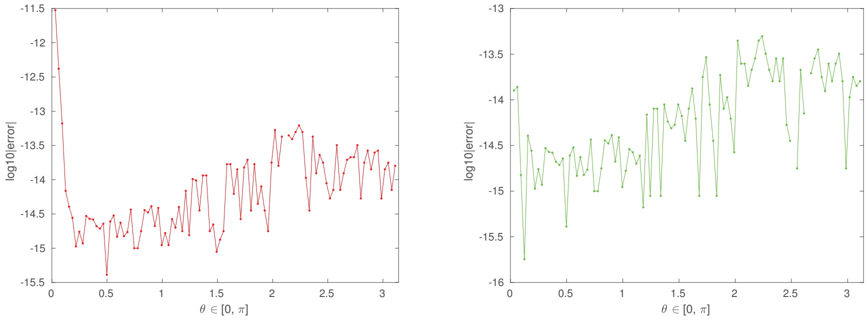


FIGURE 9.3: Absolute error in n_1 eigenvalues out of $n_8 = 12927$: error before (left), error after (right), $m_1 = 6$, $n_8 = 12927$ and $n_1 = 100$.

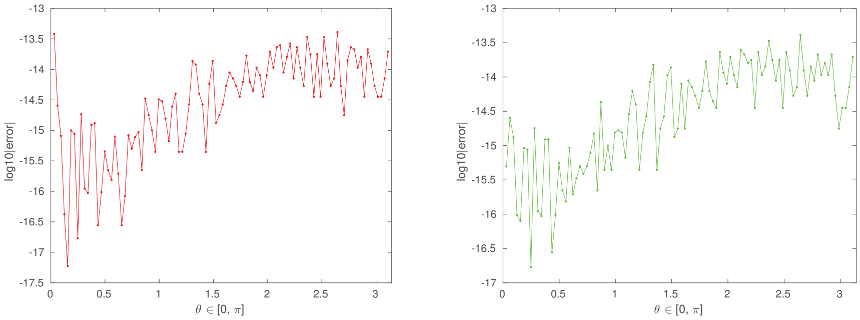


FIGURE 9.4: Absolute error in n_1 eigenvalues out of $n_8 = 12927$: error before (left), error after (right), $m_1 = 7$, $n_8 = 12927$ and $n_1 = 100$.

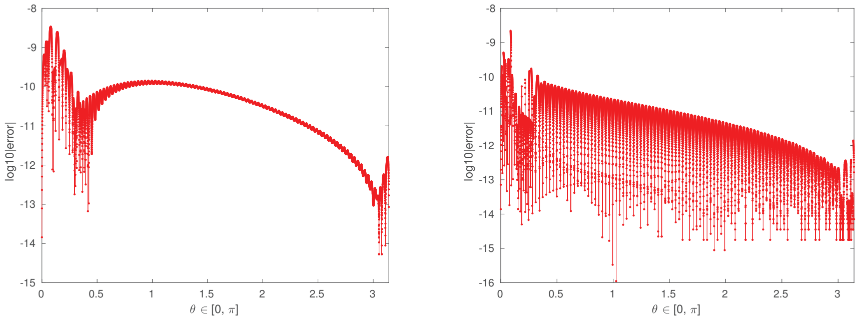


FIGURE 9.5: Extrapolation and interpolation of $n_8 = 12927$ eigenvalues: $n_1 = 100$, $m_1 = 4$ (left) and $m_1 = 5$ (right).

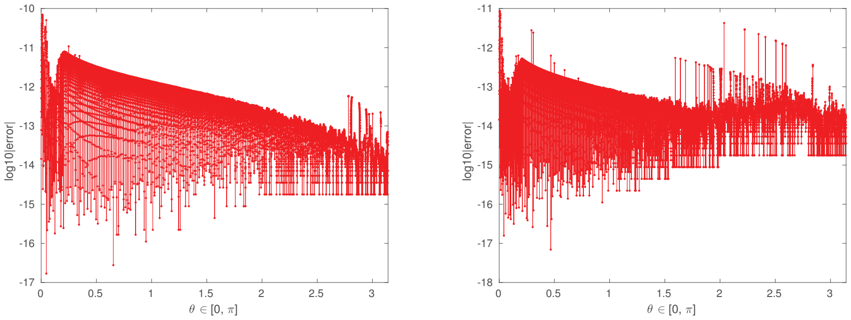


FIGURE 9.6: Extrapolation and interpolation of $n_8 = 12927$ eigenvalues: $n_1 = 100$, $m_1 = 6$ (left) and $m_1 = 7$ (right).

9.4 Summary

A new algorithm is proposed to extrapolate the eigenvalue of symmetric banded Toeplitz matrices. The bijective symbol helps us for inverse interpolation to compute the value of our introduced parameter t_i. By fitting Lagrange polynomial in t_i data we can extrapolate the values of t_i to approximate the eigenvalues of much larger symmetric banded symmetric matrix. To improve the performance of our proposed algorithm we devise a strategy to select local best approximation of eigenvalues. One we have local best approximations of eigenvalues we select global best approximations to eigenvalues.

Chapter 10

Summary and Future work

When we talk about iterative methods without memory to solves system of nonlinear equations there is a number of ways to improve their performance. Newton method is the classical method to solve equations and system of nonlinear equations with the quadratic rate of convergence. To widen the region of convergence of Newton method, we introduced a parameter without altering its quadratic rate of convergence. By changing the value of parameter we can alter the path of convergence of Newton method that may give us a fast convergence. This is not the only benefit that parameter provides us. In the case of divergence of Newton method, we have the possibility to get convergence by changing the value of the parameter in the parametrized Newton method without changing our initial guess. The parametrized Newton method also accepts the vector values for the parameter and perform well. The inclusion of multi-steps in the base method makes the Newton method Newton multi-step method. Multi-steps with frozen Jacobian are the cheaper way to increase the convergence order of iterative method with low computational cost. Low computational cost is due to frozen Jacobian.

Preconditioners for solving system of nonlinear equations could be helpful. In iterative methods preconditioners are introduced without changing the roots of original systems of nonlinear equations. Preconditioners are vector-valued function whom each component does not have any real root. Preconditioners do not increase the theoretical order of convergence of iterative method but can make the method fast by changing the course of the sequence of approximations. Our proposal for the preconditioners to improve the efficiency of Newton multi-step iterative method does not affect its convergence order and computational cost is almost the same. May be preconditioned Newton multi-step method has higher computational cost in the case of dense preconditioners and it opposite in the case of sparse preconditioners.

In realistic numerical simulations most of the time we do not know the analytical expression for the system of nonlinear equations. In such a scenario system of nonlinear equations are the black boxes. If there are no analytical expressions for system of nonlinear equations we cannot compute its Jacobian analytical and hence have to use its numerical approximation. When we use the numerical approximation of a Jacobian in an iterative method we call that method derivative free method. Our proposal for derivative-free Newton multi-step method in the case of preconditioners is also efficient. Preconditioners when are applied to derivative-free Newton multi-step method they again do not change its original order of convergence but can provide fast convergence by changing the dynamics of the iterative method.

The construction of iterative methods for solving systems of nonlinear equations to find simple roots is relative easy compared with that of roots with multiplicities. The application of preconditioners is also considered when the system of nonlinear equations has unknown multiplicities. In this case, preconditioners when used in the Newton method, for solving the system of nonlinear equations with unknown multiplicities, does not affect its quadratic convergence. The inclusion of preconditioners provides us with numerical stability and efficiency.

In Newton multi-step method the convergence order of the base method was two and there is an increment of order one per multi-step. To enhance the convergence rate of Newton multi-step method we constructed a new method whose base method has convergence order two and there is an increment of order three per two multi-steps. The idea behind the construction of higher order frozen Jacobian multi-step method is the prototype of the base method and multi-step part. We develop the prototype of the method in a way that base method help us to get maximum increment in the convergence order per multi-step. The condition of frozen Jacobian makes the iterative method computationally more effective. But in general almost all higher-order methods share a bad trait of the narrow region of convergence or in other words, they are very sensitive in the selection of initial guess in complicated simulations that provides us systems of nonlinear equations.

Iterative methods for solving nonlinear equations with memory have got the attention of a considerable number of researchers. Literature is rich with this kind of methods but there are very few numbers of efficient iterative methods with memory to solve systems of nonlinear equations. The major difference between iterative solvers for nonlinear equations and systems of nonlinear equations is the computational cost of the Jacobian and its indirect inversion. Iterative methods with memory show big convergence rate compare to the iterative method without memory for solving system of nonlinear equations but there is a trade-off between big convergence rate and additional computational cost. We can say iterative methods with memory to solve the system of nonlinear equation are marginally efficient when we compare them to iterative methods without memory. Because iterative methods with memory pay the additional computational cost of Jacobian that is constructed by the reuse of iteration information. Iterative methods with memory for the system of mildly nonlinear equations can beat their competitors with a big difference in performance because the computational cost to construct Jacobian is very low. In future, we are interested to construct iterative methods with memory for solve system of mildly nonlinear equations. Usually, the discretization of nonlinear boundary value problems provides us system of mild nonlinear equations in most of the cases.

Qualitatively and quantitatively the eigenvalues of banded symmetric Toeplitz matrices can be approximated by their generating functions. The eigenvalues of banded symmetric Toeplitz matrices can be have expansion under some conditions. The role of generating symbol is important. The eigenvalue expansion is more effective in the case of bijective generating symbols. The real benefit of such eigenvalue expansions is two extrapolate the eigenvalues of much large size matrices by collecting information from the eigenvalues of small matrices. It is interesting to note that eigenvalues are the roots of characteristic polynomials that we are approximating. So this method can be seen as a numerical method for finding the roots of characteristic polynomials. Numerically eigenvalue expansion method is very efficient because it provides a good approximation of almost all the roots of characteristic polynomials altogether. On contrary other numerical iterative solvers deal a single root at a time. The case of preconditioned banded symmetric Toeplitz matrices is also discussed and eigenvalue expansion is also constructed.

Finally, a robust numerical method to extrapolate and interpolate all the eigenvalues of banded symmetric Toeplitz matrices is developed. In this method, we do not use the expansion of eigenvalues. Our proposal of this robust numerical method is also applicable in the case of preconditioned banded symmetric Toeplitz matrices. We do not include this case in our thesis. Numerical simulations show that our numerical method is stable even when we dealt with the finite difference approximations of higher order derivative operators.

The construction of iterative methods with and without memory for the system of mildly nonlinear equations with a high order of convergence is an interesting topic for the future research topic. Especially when the nonlinearity has a finite number of linearly dependent number of derivatives.

We have observed that the numerical methods for the extrapolation and interpolation perform well in the case where the generating symbol is bijective. But where the part of generating symbol is non-bijective the proposed methods do not give reasonable accuracy. The development of numerical methods to extrapolate eigenvalues for the non-bijective generating symbol is an interesting topic for future research.

Bibliography

[1] J. F. Traub, Iterative Methods for the Solution of Equations, Publishing Company: New York, 1982.

[2] A. M. Ostrowski, Solution of Equations and Systems of Equations, Academic Press, New York, 1966.

[3] J. M. Ortega, W. C. Rheinboldt, Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, New York, 1970.

[4] T. Poinsot, D. Veynante, Theoretical and Numerical Combustion, Second Edition, PA, 2005.

[5] A. D. Polyanin, V. F. Zaitsev, Handbook of Nonlinear Partial Differential Equations, Chapman & Hall/CRC, Boca Raton, 2004.

[6] C. T. Kelley, Solving Nonlinear Equations with Newton's Method, SIAM, Philadelphia, 2003.

[7] R. L. Burden, J.D. Faires, Numerical Analysis. PWS Publishing Company, Boston, 2001.

[8] J. M. McNamee, Numerical Methods for Roots of Polynomials. Part I, Elsevier, Amsterdam, 2007.

[9] A. Böttcher, B. Silbermann, Introduction to Large Truncated Toeplitz Matrices, Springer-Verlag, New York, USA, 1999.

[10] C. Brezinski, M. Redivo Zaglia, Extrapolation Methods: Theory and Practice, Elsevier Science Publishers B.V., North-Holland, 1991.

[11] C. Garoni, S. Serra-Capizzano, Generalized Locally Toeplitz Sequences: Theory and Applications (Volume I), Springer, 2017.

[12] H. N. Abel, Beweis der Unmöglichkeit, algebraische Gleichungen von höheren Graden als dem vierten allgemein aufzulösen, Journal für die reine und angewandte Mathematik, 65(1) (1988) 66–87.

[13] F. I. Chicharro, A. Cordero, J. R. Torregrosa, Dr.awing Dynamical and Parameters Planes of Iterative Families and Methods, The Scientific World Journal, 2013 (2013) 1–11.

[14] A. Douady, J. H. Hubbard, On the dynamics of polynomials-like mappings, Annales Scientifiques de l'École Normale Supérieure, 18(2) (1985) 287–343.

[15] J. Curry, L. Garnet, D. Sullivan, On the iteration of a rational function: computer experiments with Newton's method, Communications in Mathematical Physics, 91(2) (1983) 267–277.

[16] P. Blanchard, The dynamics of Newton's method, Proceeding of Symposia in Applied Mathematics, 49 (1994) 139–154.

[17] N. Fagella, Invariants in dinàmica complexa, Butlletí de la Societat Catalana de Matemàtiques, 23(1) (2008) 29–51.

[18] J. L. Varona, Graphic and numerical comparison between iterative methods, The Mathematical Intelligencer, 24(1) (2002) 37–46.

[19] S. Amat, S. Busquier, S. Plaza, Review of some iterative root-finding methods from a dynamical point of view, Series A: Mathematical Sciences, 10 (2004) 3–35.

[20] J. M. Gutiérrez, M. A. Hernández, N.Romero, Dynamics of a new family of iterative processes for quadratic polynomials, Journal of Computational and Applied Mathematics, 233(10) (2010) 2688–2695.

[21] G. Honorato, S. Plaz, N. Romero, Dynamics of a higher-order family of iterative methods, Journal of Complexity, 27(2) (2011) 221–229.

[22] F. Chicharro, A. Cordero, J. M. Gutiérrez, J. R. Torregrosa, Complex dynamics of derivative-free methods for nonlinear equations, Applied Mathematics and Computation, 219(12) (2013) 7023–7035.

[23] S. Artidiello, F. Chicharro, A. Cordero, J. R. Torregrosa, Local convergence and dynamical analysis of a new family of optimal fourth-order iterative methods, International Journal of Computer Mathematics, 90(10) (2013) 2049–2060.

[24] M. Scott, B. Neta, C. Chun, Basin attractors for various methods, Applied Mathematics and Computation, 218(6) (2011) 2584–2599.

[25] C. Chun, M. Y. Lee, B. Neta, J. Dzunic, On optimal fourth-order iterative methods free from second derivative and their dynamics, Applied Mathematics and Computation, 218(11) (2012) 6427–6438.

[26] B. Neta, M. Scott, C. Chun, Basin attractors for various methods for multiple roots, Applied Mathematics and Computation, 218(9) (2012) 5043–5066.

[27] A. Cordero and J. Garcia-Maimo and J. R. Torregrosa. M. P. Vassileva, P. Vindel, Chaos in King's iterative family, Applied Mathematics Letters, 26(8) (2013) 842–848.

[28] R. L. Devaney, The Mandelbrot set, the Farey tree, and the Fibonacci sequence, The American Mathematical Monthly, 106(4) (1999) 289–302.

[29] Y. I. Kim, A triparametric family of three-step optimal eighth-order methods for solving nonlinear equations, International Journal of Computer Mathematics, 89(8) (2012) 1051–1059.

[30] A. Cordero, J. R. Torregrosa, P. Vindel, Dynamics of a family of Chebyshev-Halley-type method, Applied Mathematics and Computation, 219 (2013) 8568–8583.

[31] P. Jarratt, Some fourth order multi-point iterative methods for solving equations, Mathematics of Computation, 20(95) (1996) 434-437.

[32] F. Soleymani, M. Sharifi, S. Shateyi, F. K. Haghani, A class of Steffensen-type iterative methods for nonlinear systems, Journal of Applied Mathematics, 2014 (2014) 1–9.

[33] M. A. Noor, M. Waseem, Some iterative methods for solving a system of nonlinear equations, Computers & Mathematics with Applications, 57(1) (2009) 101–106.

[34] D. K. R. Babajee, A. Cordero, F. Soleymani, J. R. Torregrosa, On a novel fourth-order algorithm for solving systems of nonlinear equations, Journal of Applied Mathematics, 2012 (2012) 1–12.

[35] F. Awawdeh, On new iterative method for solving systems of nonlinear equations, Numerical Algorithms, 54 (2010) 395–409.

[36] D. K. R. Babajee, M. Z. Dauhooa, M. T. Darvishi, A. Karamib, A. Barati, Analysis of two Chebyshev-like third order methods free from second derivatives for solving systems of nonlinear equations, Journal of Computational and Applied Mathematics, 233(8) (2010) 2002–2012.

[37] F. Ahmad, E. Tohidi, M.Z. Ullah, J. A. Carrasco, Higher order multi-step Jarratt-like method for solving systems of nonlinear equations: Application to PDEs and ODEs, Computers & Mathematics with Applications, 70(4) (2015) 624–636.

[38] D. A. Budzkoa, A. Cordero, J. R. Torregros, New family of iterative methods based on the Ermakov-Kalitkin scheme for solving nonlinear systems of equations, Computational Mathematics and Mathematical Physics, 55(12) (2015) 1947–1959.

[39] A. Cordero, J. R. Torregrosa, Variants of Newton's Method using fifth-order quadrature formulas, Applied Mathematics and Computation, 190(1) (2007) 686–698.

[40] A. Cordero, F. Soleymani, J. R. Torregrosa, Dynamical analysis of iterative methods for nonlinear systems or how to deal with the dimension? Applied Mathematics and Computation, 244(1) (2014) 398–412.

[41] A. Genocchi, Relation entre la différence et la dérivée d'un même ordre quelconque, Archiv Mathematical Physics I, 49 (1869) 342–345.

[42] M. Grau-Sánchez, M. Noguera, S. Amat, On the approximation of derivatives using divided difference operators preserving the local convergence order of iterative methods, Journal of Computational and Applied Mathematics, 237(1) (2013) 363–372.

[43] M. Grau-Sánchez, M. Noguera, J. L. Diaz-Barrero, Note on the efficiency of some iterative methods for solving nonlinear equations, SeMA Journal, 71(1) (2015) 15–22.

[44] A. Griewank, Broyden Updating, the Good and the Bad! Documenta Mathematica, Extra Volume 21st International Symposium on Mathematical Programming, (2012), 301–315.

[45] C. Hermite, Sur la formule d'interpolation de Lagrange, Journal für die reine und angewandte Mathematik, 84 (1878) 70–79.

[46] T. Lotfi, F. Soleymani, M. Ghorbanzadeh, P. Assari, On the construction of some tri-parametric iterative methods with memory, Numerical Algorithms, 70(4) (2015) 835–845.

[47] H. Montazeri, F. Soleyman, S. Shateyi, S. Motsa, On a new method for computing the numerical solution of systems of nonlinear equations, Journal of Applied Mathematics, 2012 (2012) 1–15.

[48] M. S. Petković, J. R. Sharma, On some efficient derivative-free iterative methods with memory for solving systems of nonlinear equations, Numerical Algorithms, 71(2) (2016) 457–474.

[49] F. -A. Potra, A characterisation of the divided differences of an operator which can be represented by Riemann integrals, Revue d'analyse numérique et de théorie de l'approximation, 2 (1980) 251–253.

[50] J. Schmidt, Die regula falsi für operatoren in Banachräumen, Zeitschrift für Angewandte Mathematik und Mechanik, 41 (1961) 61–63.

[51] J. R. Sharma, H. Arora, M. S. Petković, An efficient derivative free family of fourth order methods for solving systems of nonlinear equations, Applied Mathematics and Computation, 235 (2014) 383–393.

[52] A. R. Soheili, F. Soleymani, Iterative methods for nonlinear systems associated with finite difference approach in stochastic differential equations, Numerical Algorithms, 71(1) (2016) 89–102.

[53] F. Soleymani, T. Lotfi, P. Bakhtiari, A multi-step class of iterative methods for nonlinear systems, Optimization Letters, 8(3) (2014) 1001–1015.

[54] P. S. Stanimirović, F. Soleymani, F. K. Haghani, Computing outer inverses by scaled matrix iterations, Journal of Computational and Applied Mathematics, 296 (2016) 89–101.

[55] I. G. Tsoulos, A. Stavrakoudis, On locating all roots of systems of nonlinear equations inside bounded domain using global optimization methods, Nonlinear Analysis: Real World Applications, 11(4) (2010) 2465–2471.

[56] M. Z. Ullah, F. Soleymani, A. S. Al-Fhaid, Numerical solution of nonlinear systems by a general class of iterative methods with application to nonlinear PDEs, Numerical Algorithms, 67(1) (2014) 223–242.

[57] M. Z. Ullah, S. Serra-Capizzano, F. Ahmad, E. S. Al-Aidarous, Higher order multi-step iterative method for computing the numerical solution of systems of nonlinear equations: Application to nonlinear PDEs and ODEs, Applied Mathematics and Computation, 269 (2015) 972–987.

[58] Wolfram Research, Inc., Mathematica, Version 10.0, Champaign, IL (2014).

[59] W. L. Cruz, J. M. Martinez, M. Raydan, Spectral residual method without gradient information for solving large-scale nonlinear systems of equations, Mathematics of Computation, 75(255) (2006) 1429–1448.

[60] H. B. An, Z. Z. Bai, A globally convergent Newton-GMRES method for large sparse systems of nonlinear equations, Applied Numerical Mathematics, 57(3) (2007) 235–252.

[61] M. Kh. Zak, F. Toutounian, Nested splitting conjugate gradient method for matrix equation $AXB = C$ and preconditioning, Computers & Mathematics with Applications, 66(3) (2013) 269–278.

[62] A. Cordero, J. L. Hueso, E. Martinez, J. R. Torregrosa, A modified Newton-Jarratt's composition, Numerical Algorithms, 55(1) (2010) 87–99.

[63] J. R. Sharma, H. Arora, Efficient Jarratt-like methods for solving systems of nonlinear equations, Calcolo, 51(1) (2014) 193–210.

[64] J. R. Sharma, R. K. Guha, R. Sharma, An efficient fourth order weighted-Newton method for systems of nonlinear equations, Numerical Algorithms, 62(2) (2013) 307–323.

[65] F. Soleymani, T. Lotfi, P. Bakhtiari, A multi-step class of iterative methods for nonlinear systems, Optimization Letters, 8(3) (2014) 1001–1015.

[66] A. Cordero, J. L. Hueso, E. Martínez, J. R. Torregrosa, Efficient high-order methods based on golden ratio for nonlinear systems, 217 (9) (2011) 4548–4556.

[67] A. Cordero, J. L. Hueso, E. Martínez, J. R. Torregrosa, Increasing the convergence order of an iterative method for nonlinear systems, 25 (12) (2012) 2369–2374.

[68] L. O. Jay, A note on Q-order of convergence, BIT, 41 (2) (2001) 422–429.

[69] S. Weerakoon, T. G. I. Fernando, A variant of Newton's method with accelerated third-order convergence, Applied Mathematics Letters, 13(8) (2000) 87–93.

[70] M. Z. Ullah, S. Serra-Capizzano, F. Ahmad, An efficient multi-step iterative method for computing the numerical solution of systems of nonlinear equations associated with ODEs, Applied Mathematics and Computation, 250 (2015) 249–259.

[71] D. Budzko, A. Cordero, J. R. Torregrosa, Modifications of Newton's method to extend the convergence domain, SeMA Journal, 66(1) (2014) 43–53.

[72] E. S. Alaidarous, M. Z. Ullah, F. Ahmad, and A. S. Al-Fhaid, An Efficient Higher-Order Quasilinearization Method for Solving Nonlinear BVPs, Journal of Applied Mathematics, 2013 (2013) 1–11.

[73] S. S. Motsa and S. Shateyi, New Analytic Solution to the Lane-Emden Equation of Index 2, Mathematical Problems in Engineering, 2012 (2012) 1–19.

[74] C. A. Ladeia, N. M. L. Romeiro, Numerical Solutions of the 1D Convection-Diffusion-Reaction and the Burgers Equation using Implicit Multi-stage and Finite Element Methods, Integral Methods in Science and Engineering, 2013 (2013) 205–216.

[75] T. S. Jang, An integral equation formalism for solving the nonlinear Klein-Gordon equation, Applied Mathematics and Computation, 243(1) (2014) 322–338.

[76] D. Gurarie, K. W. Chow, Vortex arrays for sinh-Poisson equation of two-dimensional fluids Equilibria and stability, Physics of Fluids, 16(9) (2004) 165–178.

[77] M. Averick Brett, M. Ortega James, fast solution of nonlinear Poisson-type equations, Argonne national laboratory, 9700 South Cass Avenue, August 1991.

[78] A. H. Bhrawy, An efficient Jacobi pseudospectral approximation for nonlinear complex generalized Zakharov system, Applied Mathematics and Computation, 247 (2014) 30–46.

[79] S. Abbasbandy, E. Babolian, M. Ashtiani, Numerical solution of the generalized Zakharov equation by homotopy analysis method, Communications in Nonlinear Science and Numerical Simulation, 14(12) (2009) 4114–4121.

[80] Q. Chang, H. Jiang, A conservative difference scheme for the Zakharov equations, Journal of Computational Physics, 113(2) (1994) 309–319.

[81] Q. Chang, B. Guo, H. Jiang, Finite difference method for generalized Zakharov equations, Mathematics of Computation, 64(210) (1995) 537–553.

[82] M. Javidi, A. Golbabai, Exact and numerical solitary wave solutions of generalized Zakharov equation by the variational iteration method, Chaos, Solitons & Fractals, 36(2) (2008) 309–313.

[83] W. Bao, F. Sun, G. W. Wei, Numerical methods for the generalized Zakharov system, Journal of Computational Physics, 190(1) (2003) 201–228.

[84] W. Bao, F. Sun, Efficient and stable numerical methods for the generalized and vector Zakharov system, SIAM Journal on Scientific Computing, 26 (2005) 1057–1088.

[85] M. Dehghan, F. F. Izadi, The spectral collocation method with three different bases for solving a nonlinear partial differential equation arising in modeling of nonlinear waves, Mathematical and Computer Modelling, 5(9–10) (2011) 1865–1877.

[86] E. H. Doha, A. H. Bhrawy, S. S. Ezz-Eldien, Efficient Chebyshev spectral methods for solving multi-term fractional orders differential equations, Applied Mathematical Modelling, 35(12) (2011) 5662–5672.

[87] E. H. Doha, A. H. Bhrawy, R. M. Hafez, On shifted Jacobi spectral method for high-order multi-point boundary value problems, Communications in Nonlinear Science and Numerical Simulation, 17(10) (2012) 3802–3810.

[88] O. R. N. Samadi, E. Tohidi, The Spectral Method for Solving Systems of Volterra Integral Equations, Journal of Applied Mathematics and Computing, 40(1–2) (2012) 477–497.

[89] E. Tohidi, O. R. N. Samadi, Optimal Control of Nonlinear Volterra Integral Equations via Legendre Polynomials, IMA Journal of Mathematical Control and Information, 30(1) (2013) 67–83.

[90] E. Tohidi, S. Lotfi Noghabi, An efficient Legendre Pseudospectral Method for Solving Nonlinear Quasi Bang-Bang Optimal Control Problems, Journal of Applied Mathematics, Statistics and Informatics, 8(2) (2012) 73–85.

[91] Z. Wang, B.-Y. Guo, Jacobi rational approximation and spectral method for differential equations of degenerate type, Mathematics of Computation, 77(262) (2008) 883–907.

[92] J.M. Gutiérrez, M.A. Hernández, A family of Chebyshev-Halley type methods in Banach spaces, Bulletin of the Australian Mathematical Society, 55(1) (1997) 113–130.

[93] M. Frontini, E. Sormani, Some variant of Newton's method with third-order convergence, Applied Mathematics and Computation, 140 (2003) 419–426.

[94] H. H. Homeier, A modified Newton method with cubic convergence: the multivariable case, Journal of Computational and Applied Mathematics, 169(1) (2004) 161–169.

[95] M. Grau-Sánchez, À. Grau, M. Noguera, Ostrowski type methods for solving systems of nonlinear equations, Applied Mathematics and Computation, 218(6) (2011) 2377–2385.

[96] P. Jarratt, Some fourth order multipoint iterative methods for solving equations, Mathematics of Computation, 20(95) (1966) 434–437.

[97] M. S. Petkovic, On a general class of multipoint root-finding methods of high computational efficiency, SIAM Journal on Numerical Analysis, 47(6) (2010) 4402–4414.

[98] S. S. Motsa, S. Shateyi, New Analytic Solution to the Lane-Emden Equation of Index 2, Mathematical Problems in Engineering, 2012 (2012) 1–19.

[99] M. Grau-Sánchez, À. Grau, M. Noguera, On the computational efficiency index and some iterative methods for solving systems of nonlinear equations, Journal of Computational and Applied Mathematics, 236(6) (2011) 1259–1266.

[100] L. Howarth, On the solution of the laminar boundary layer equations, Proceedings of Royal Society A, 164 (1938) 547–579.

[101] K. W. Chow, S. C. Tsang, C. C. Mak, Another exact solution for two-dimensional, inviscid sinh Poisson vortex arrays, Physics of Fluids, 15 (2003) 264–281.

[102] H. Montazeri, F. Soleymani, S. Shateyi, S. S. Motsa, On a New Method for Computing the Numerical Solution of Systems of Nonlinear Equations, Journal of Applied Mathematics, 2012 (2012) 1–15.

[103] A. Cordero, M. Kansal, V. Kanwar, A stable class of improved second-derivative free Chebyshev-Halley type methods with optimal eighth order convergence, Numerical Algorithms, 72(4) (2016) 937.

[104] V. Arroyo, A. Cordero, J. R. Torregrosa, Approximation of artificial satellites' preliminary orbits: The efficiency challenge, Mathematical and Computer Modelling, 54(7–8) (2011), 1802-1807.

[105] S. Qasim, Z. Ali, F. Ahmad, S. Serra-Capizzano, M. Z. Ullah, A. Mahmood, Solving systems of nonlinear equations when the nonlinearity is expensive, Computers & Mathematics with Applications, 71(7) (2016) 1464–1478.

[106] U. Qasim, Z. Ali, F. Ahmad, S. Serra-Capizzano, M. Z. Ullah, M. Asma, Constructing Frozen Jacobian Iterative Methods for Solving Systems of Nonlinear Equations, Associated with ODEs and PDEs Using the Homotopy Method, Algorithms, 9(1) (2016).

[107] F. Ahmad, E. Tohidi, J. A. Carrasco, A parameterized multi-step Newton method for solving systems of nonlinear equations, Numerical Algorithms, 71(3) (2016) 631–533.

[108] F. Ahmad, E. Tohidi, M. Z. Ullah, J. A. Carrasco, Higher order multi-step Jarratt-like method for solving systems of nonlinear equations: Application to PDEs and ODEs , Computers & Mathematics with Applications, 70(4) (2015) 624–636.

[109] X. Wu, Note on the improvement of Newton's method for systems of nonlinear equations, Applied Mathematics and Computation, 189(2) (2007) 1476–479.

[110] J. L. Hueso, E. Martínez, J. R. Torregrosa, Modified Newton's method for systems of nonlinear equations with singular Jacobian, Journal of Computational and Applied Mathematics, 224(1) (2009) 77–83.

[111] M. A. Noor and F. A. Shah, A Family of Iterative Schemes for Finding Zeros of Nonlinear Equations having Unknown Multiplicity, Applied Mathematics & Information Sciences, 8(5) (2014) 2367–2373.

[112] M.A. Noor, M. Waseem, K.I. Noor, E. Al-Said, Variational iteration technique for solving a system of nonlinear equations, Optimization Letters, 7(5) (2013) 991–1007.

[113] S. Amat, S. Busquier, A. Grau, M. Grau-Snchez, Maximum efficiency for a family of Newton-like methods with frozen derivatives and some applications. Applied Mathematics and Computation, 219(15) (2013) 7954–7963.

[114] J. R. Sharma, H. Arora, A novel derivative free algorithm with seventh order convergence for solving systems of nonlinear equations, Numerical Algorithms, 67(4) (2014) 917–933.

[115] J. R. Sharma, H. Arora, An efficient derivative free iterative method for solving systems of nonlinear equations, Applicable Analysis and Discrete Mathematics, 7(2) (2013) 390–403.

[116] M. Grau-Snchez, A. Grau, M. Noguera, Frozen divided difference scheme for solving systems of nonlinear equations, Journal of Computational and Applied Mathematics, 235(6) (2011) 1739–1743.

[117] M. Grau-Sánchez, M. Noguera, A technique to choose the most efficient method between secant method and some variants, Applied Mathematics and Computation, 218(11) (2012) 6415–6426.

[118] M. Grau-Sánchez, M. Noguera, S. Amat, On the approximation of derivatives using divided difference operators preserving the local convergence order of iterative methods, Journal of Computational and Applied Mathematics, 237(1) (2013) 363–372.

[119] C. Chun, A method for obtaining iterative formulas of order three, Applied Mathematics Letters, 20(11) (2007), 1103–1109.

[120] C. Chun, On the construction of iterative methods with at least cubic convergence, Applied Mathematics and Computation, 189(2) (2007) 1384–1392.

[121] C. Chun, Some variant of Chebshev-Halley method free from second derivative, Applied Mathematics and Computation, 191(1) (2007) 1384–1392.

[122] N. Osada, Improving the order of convergence of iterative functions, Journal of Computational and Applied Mathematics, 98(2) (1998), 311–315.

[123] M. A. Noor, F. A. Shah, Variational iteration technique for solving nonlinear equations, Journal of Applied Mathematics and Computing, 31(1–2) (2009) 247–254.

[124] M. A. Noor, F. A. Shah, K. I. Noor and E. Al-said, Variational iteration technique for finding multiple roots of nonlinear equations, Scientific Research and Essays, 6(6) (2011) 1344–1350.

[125] M. A. Noor and F. A. Shah, A family of iterative schemes for finding zeros of nonlinear equations having unknown multiplicity, Applied Mathematics & Information Sciences, 8(5) (2014) 1–7.

[126] F. A. Shah, M. A. Noor and M. Batool, Derivative-free iterative methods for solving nonlinear equations, Applied Mathematics & Information Sciences, 8(5) (2014) 1–5.

[127] F. Ahmad, S. Serra-Capizzano, M. Z. Ullah, A. S. Al-Fhaid, A Family of Iterative Methods for Solving Systems of Nonlinear Equations Having Unknown Multiplicity, Algorithms, 9(1)5 (2016).

[128] M. Barrera, S. M. Grudsky, Asymptotics of eigenvalues for pentadiagonal symmetric Toeplitz matrices, Operator Theory: Advances and Applications, 259 (2017) 51–77.

[129] R. Bhatia, Matrix Analysis, Graduate Texts in Mathematics, 169. Springer-Verlag, New York (1997).

[130] D. Bini, M. Capovani, Spectral and computational properties of band symmetric Toeplitz matrices, Linear Algebra and its Applications, 52(53) (1983) 99–126.

[131] J. M. Bogoya, A. Böttcher, S. M. Grudsky, E. A. Maximenko, Eigenvalues of Hermitian Toeplitz matrices with smooth simple-loop symbols, Journal of Mathematical Analysis and Applications, 422(2) (2015) 1308–1334.

[132] J. M. Bogoya, S. M. Grudsky, E. A. Maximenko, Eigenvalues of Hermitian Toeplitz matrices generated by simple-loop symbols with relaxed smoothness, Operator Theory: Advances and Applications, 259 (2017) 179–212.

[133] A. Böttcher, S. M. Grudsky, E. A. Maximenko, Inside the eigenvalues of certain Hermitian Toeplitz band matrices, Journal of Computational and Applied Mathematics, 233(9) (2010) 2245–2264.

[134] R. H. Chan, M. Ng, Conjugate gradient methods for Toeplitz systems, SIAM Review, 38(3) (1996) 427–482.

[135] R. H. Chan, P. Tang, Fast band-Toeplitz preconditioners for Hermitian Toeplitz systems, SIAM Journal on Scientific Computing, 15(1) (1994) 164–171.

[136] F. Di Benedetto, G. Fiorentino, S. Serra-Capizzano, C.G. Preconditioning for Toeplitz Matrices, Computers & Mathematics with Applications, 25(6) (1993) 33–45.

[137] S.-E. Ekström, S. Serra-Capizzano, Eigenvalues and Eigenvectors of Banded Toeplitz Matrices and the Related Symbols, Numerical Linear Algebra with Applications, in press (2018).

[138] S.-E. Ekström, C. Garoni, S. Serra-Capizzano, Are the eigenvalues of banded symmetric Toeplitz matrices known in almost closed form? Exp. Math., in press (2017).

[139] T. Huckle, S. Serra-Capizzano, C. Tablino-Possio, Preconditioning strategies for non-Hermitian Toeplitz linear systems, Numerical Linear Algebra with Applications, 12(2/3) (2005) 211–220.

[140] T. Huckle, S. Serra-Capizzano, C. Tablino-Possio, Preconditioning strategies for Hermitian indefinite Toeplitz linear systems, SIAM Journal on Scientific Computing, 25(5) (2004) 1633–1654.

[141] S. Serra-Capizzano, New PCG based algorithms for the solution of Hermitian Toeplitz systems, Calcolo 32 (1995) 53–176.

[142] S. Serra-Capizzano, Optimal, quasi-optimal and superlinear band-Toeplitz preconditioners for asymptotically ill-conditioned positive definite Toeplitz systems, Mathematics of Computation, 66(218) (1997) 651–665.

[143] S. Serra-Capizzano, An ergodic theorem for classes of preconditioned matrices, Linear Algebra and its Applications, 282(1–3) (1998) 161–183.

[144] J. Stoer, R. Bulirsch, Introduction to Numerical Analysis, Third Edition, Springer (2002).