UNIVERSITI TEKNOLOGI MARA

MODIFIED PICARD ITERATIVE METHOD FOR EPIDEMIOLOGICAL MODELING AND NONLINEAR DYNAMICS

AHMAD BAZLI KHAIRUDDIN

MSc

July 2026

UNIVERSITI TEKNOLOGI MARA

MODIFIED PICARD ITERATIVE METHOD FOR EPIDEMIOLOGICAL MODELING AND NONLINEAR DYNAMICS

AHMAD BAZLI KHAIRUDDIN

Thesis submitted in fulfilment
of the requirements for the degree of
Master of Science
(Mathematics)

Faculty of Computer and Mathematical Sciences

July 2026

CONFIRMATION BY PANEL OF EXAMINERS

I certify that a Panel of Examiners has met on July 9, 2026 to conduct the final examination of Ahmad Bazli Khairuddin on his Master of Science (Mathematics) thesis entitled "Modified Picard Iterative Method for Epidemiological Modeling and Nonlinear Dynamics" in accordance with Universiti Teknologi MARA Act 1976 (Akta 173). The Panel of Examiners recommends that the student be awarded the relevant degree. The Panel of Examiners was as follows:

Norizan Mat Diah, PhD
Associate Professor
Faculty of Computer and Mathematical Sciences
Universiti Teknologi MARA
(Chairman)

Iskandar Shah Mohd Zawawi, PhD
Associate Professor
Faculty of Computer and Mathematical Sciences
Universiti Teknologi MARA
(Internal Examiner)

Khatim Hassan, PhD
Associate Professor
Faculty of Information Science and Technology
Universiti Kebangsaan Malaysia
(External Examiner)

PROF DR HJH ZURAEDA IBRAHIM
Dean
Institute of Postgraduate Studies
Universiti Teknologi MARA
Date: 16 July 2026

AUTHOR'S DECLARATION

I declare that the work in this thesis was carried out in accordance with the regulations of Universiti Teknologi MARA. It is original and is the results of my own work, unless otherwise indicated or acknowledged as referenced work. This thesis has not been submitted to any other academic institution or non-academic institution for any degree or qualification.

I, hereby, acknowledge that I have been supplied with the Academic Rules and Regulations for Postgraduate, Universiti Teknologi MARA, regulating the conduct of my study and research.

Name of Student	:	Ahmad Bazli Khairuddin
Student I.D. No.	:	2024618356
Programme	:	Master of Science Mathematics - CDCS752
Faculty	:	Faculty of Computer and Mathematical Sciences
Thesis Title	:	Modified Picard Iterative Method for Epidemiological Modeling and Nonlinear Dynamics
Signature of Student	:	....
Date	:	July 2026

ABSTRACT

Mathematical models involving ordinary differential equations (ODEs) and partial differential equations (PDEs) are widely used to describe real-life dynamic systems such as disease transmission, biochemical reactions, population interactions, nonlinear growth, and reaction–diffusion processes. This thesis develops enhanced numerical schemes based on the classical Picard Iterative Method (PIM), namely the Multistage Picard Iterative Method (MPIM), the Picard Iterative Method for *n*th-order differential equations (PIM*n*), and the Multistage Picard Iterative Method for *n*th-order differential equations (MPIM*n*). The proposed methods are applied to selected nonlinear ODEs, systems of ODEs, higher-order differential equations, and PDEs. The study was motivated by two limitations of the classical PIM. First, although the classical PIM provides accurate approximations over short time intervals, it may exhibit increasing errors and reduced numerical stability over extended intervals when applied to nonlinear models. Second, the standard formulation of the classical PIM is developed for first-order initial value problems and requires transformation or reformulation when applied to higher-order differential equations. To address these limitations, the MPIM introduces a partitioned time-interval strategy, where Picard corrections are applied within each subinterval before the initial conditions are updated for the next subinterval, whereas PIM*n* and MPIM*n* are formulated to solve higher-order differential equations directly. All computations were implemented in Maple. For the system of ODE applications, the numerical performance of the PIM and MPIM was evaluated through absolute error analysis and comparison with the Adomian Decomposition Method (ADM), Multistage Adomian Decomposition Method (MADM), and the fourth-order Runge–Kutta (RK4) method. For the Monkeypox transmission model, the eighth-order Runge–Kutta (RK8) method was additionally included as a higher-order numerical reference. The results indicate that MPIM generally provides closer agreement with the reference numerical solutions than the classical PIM for the first-order nonlinear models considered, particularly over extended time intervals where PIM may accumulate larger errors and exhibit less stable numerical behaviour. The results also indicate that PIM*n* and MPIM*n* can solve higher-order differential equations directly without transforming them into first-order systems, with MPIM*n* generally producing more stable approximations and closer numerical agreement than PIM*n*. However, the improved numerical behaviour of MPIM and MPIM*n* requires additional computational effort because the multistage formulation divides the time domain into many subintervals, such as 12,000 subintervals in the long-term simulation. Therefore, the contribution of the proposed methods is not claimed in terms of computational efficiency, but rather in terms of improved long-term numerical accuracy and stability compared with the classical PIM and PIM*n*.

ACKNOWLEDGEMENT

Praise be to Allah who gives me the strength, inspiration and prudence to bring this thesis a close. Peace be upon His Messenger Muhammad and his honorable family.

Throughout this academic quest, the following have generously granted their precious guidance and never-ending assistance for which I'm eternally grateful.

First of all, I would like to express my deepest gratitude to my supervisor, Dr. Mat Salim Selamat and my co-supervisor Assoc. Prof. Dr. Nor Azizah Yacob, for their immeasurable wisdom, endless patience and ever constructive comments.

I am also indebted to my beloved parents, sisters, and brother. Words cannot express my gratitude for their love, support, patience, constant encouragement and understanding that acts as a giant support for my success.

My thanks also go to all lecturers at the School of Mathematical Sciences in the Universiti Teknologi MARA cawangan Negeri Sembilan kampus Seremban for their cooperation, and the friendly atmosphere around the school.

Last but not least in recollection of many kindness, continuous encouragement for perseverance and patience, my grateful thanks to all my friends and colleagues, wherever they may be, thank you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Abbreviations

ADM	Adomian Decomposition Method
DJM	Daftardar-Geiji and Jafari Method
HPM	Homotopy-Perturbation Method
IVPs	Initial Value Problems
MHPM	Multistage Homotopy-Perturbation Method
MPIM	Multistage picard iterative method
MPIM*n*	Multistage picard iterative method for *n*th order
ODEs	Ordinary differential equations
PDEs	Partial differential equations
PIM	Picard iterative method
PIM*n*	Picard iterative method for *n*th order
RK4	Fourth-Order Runge-Kutta
VIM	Variational iterative method

CHAPTER 1
INTRODUCTION

1.1 Research Background

Monkeypox is a zoonotic disease caused by a virus belonging to the Orthopoxvirus family, sharing several clinical similarities with smallpox but generally being less severe (Letafati & Sakhavarz, 2023). The disease can be transmitted from animals to humans, particularly through direct contact with infected animals, while human-to-human transmission has also been documented (Di Giulio & Eckburg, 2004). Its ability to spread across species makes it more complex to control compared to other diseases, especially in today's highly interconnected world (Krishna et al., 2024). Therefore, a deeper scientific understanding of the infection patterns and transmission dynamics of monkeypox is essential to enable more effective prevention, control, and public health responses.

In addressing these global health challenges, mathematical models play a vital role in understanding the dynamics of disease transmission, offering a structured and quantitative approach to analyze how infections spread within and between populations. These models are particularly valuable because they provide the ability to predict the potential course of an outbreak, simulate various intervention strategies, and evaluate the likely effectiveness of different control measures before they are implemented in real-world settings (Ngungu et al., 2023). In epidemiology, one of the most widely adopted approaches is the use of systems of Ordinary Differential Equations (ODEs), which divide populations into compartments such as susceptible, exposed, infected, and recovered individuals (Cascante-Vega et al., 2022). By mathematically describing the rates of transition between these compartments, ODE-based models can capture the complex interplay between biological, social, and environmental factors that influence disease dynamics (Onifade et al., 2025). Furthermore, such models allow the estimation of key epidemiological parameters, such as the basic reproduction number (R_0), transmission rate, and recovery rate, which are critical in guiding public health decision-making (Liao et al., 2023). In this

way, mathematical modeling serves as an indispensable tool that bridges theoretical understanding with practical applications in controlling infectious diseases.

Although ODEs provide a powerful framework for modeling the spread of infectious diseases, the resulting systems are often nonlinear and highly complex (Hoang & Ehrhardt, 2025). These nonlinearities may arise from factors such as multiple modes of transmission, population heterogeneity, or feedback mechanisms within the host and environment. As a consequence, obtaining exact analytical solutions to such models is usually extremely difficult, and in many cases, mathematically impossible (Piovella, 2020). This limitation significantly reduces the practical applicability of purely theoretical approaches, especially when rapid and accurate insights are required for public health decision-making. To address this issue, numerical methods are employed to generate approximate solutions that can closely capture the dynamics of disease transmission (Adom-Konadu et al., 2023). Numerical techniques not only provide a practical alternative to analytical solutions but also allow researchers to simulate scenarios under different parameter values, explore long-term behavior of epidemics, and assess the effectiveness of intervention strategies (Lin et al., 2024; Onifade et al., 2025; Qian et al., 2025). Thus, the use of numerical methods has become indispensable in modern epidemiological modeling, bridging the gap between theoretical formulations and real-world applications.

The study of infectious disease dynamics is often carried out through compartmental models, which categorize individuals in a population based on their disease status. Classical models such as the SIR (Susceptible–Infected–Recovered) and its extension, the SEIR (Susceptible–Exposed–Infected–Recovered), have been widely adopted due to their simplicity and adaptability. The SIR model, for example, assumes individuals move directly from being susceptible to be infected and then recovered, whereas the SEIR model accounts for an additional exposed stage, representing the latent period before symptoms emerge (Gao et al., 2023; Tamakloe et al., 2025). The SEIR model is particularly relevant for the disease with distinct incubation periods, such as monkeypox. Variants of these models can also incorporate additional compartments, such as quarantine, hospitalization, or vaccination, to more accurately reflect real-world interventions (Madubueze et al., 2022; Musafir et al., 2024). By capturing the dynamic transitions between compartments, these models

provide a valuable lens through which researchers can study how an outbreak evolves under different scenarios. However, increasing complexity of such models inevitably leads to more challenging mathematical formulations, particularly when nonlinear terms are involved (Hoang & Ehrhardt, 2025). This complexity necessitates the use of advanced numerical methods to obtain reliable solutions.

Over the years, various numerical methods have been developed to approximate solutions of nonlinear ODEs arising in epidemiological models. Among the most widely applied are classical schemes such as the Runge–Kutta methods, as well as semi-analytical approaches like the Variational Iteration Method (VIM), the Adomian Decomposition Method (ADM), and the Daftardar–Gejji and Jafari Method (DJM). While these methods have proven effective in many contexts, they are not without limitations. Runge–Kutta methods, for instance, often require small step sizes to maintain stability, leading to increased computational cost. Likewise, decomposition-based methods such as ADM and DJM may suffer from slow convergence when applied to highly nonlinear problems or systems of higher order. Abed and AL-Jawary (2021) stated that when the DJM is applied to the SIR model, the convergence of the resulting iterative series is contingent upon satisfying specific conditions related to the model parameters and the selection of terms in the approximation series. Failure to meet these conditions may result in slow error reduction or necessitate a large number of iterations to achieve acceptable accuracy. In practice, these approaches can become computationally intensive and less efficient, particularly when dealing with the complex, nonlinear dynamics typical of infectious disease models like monkeypox. These challenges highlight the need for alternative or modified techniques that can offer both accuracy and efficiency in solving such systems.

In the present study, the ADM and the Multistage ADM (MADM) are included as supplementary semi-analytical comparison methods to provide a broader evaluation of the proposed Picard-based approaches. The ADM and MADM results are compared with those obtained using the PIM and MPIM for selected nonlinear systems of ODEs, namely the Monkeypox transmission model, COVID-19 model, biochemical reaction model, and multispecies Lotka-Volterra equations. In addition, the fourth-order Runge-Kutta method (RK4) is used as a numerical reference method,

while the eighth-order Runge-Kutta method (RK8) is additionally employed as a higher-order numerical benchmark for the Monkeypox transmission model due to its extended simulation interval and large number of time steps. These comparisons provide a more comprehensive assessment of the accuracy and long-time numerical behaviour of the proposed methods.

Among the broad spectrum of numerical approaches, iterative methods hold particular promise due to their step-by-step refinement process and capacity to handle nonlinear systems effectively. Unlike direct solution techniques, iterative methods progressively improve approximations until the desired level of accuracy is achieved. Although these methods have been widely studied in numerical analysis and applied across various fields of science and engineering, their application in epidemiological modeling remains relatively limited. This presents an opportunity to introduce new perspectives into disease modeling by leveraging iterative schemes. In particular, the Picard Iterative Method (PIM) provides a systematic procedure for solving Initial Value Problems (IVPs). However, its accuracy tends to deteriorate over extended time intervals, making it unreliable for long-term predictions. To overcome this limitation, this study proposes the Multistage Picard Iterative Method (MPIM), which enhances the numerical stability and convergence of the classical PIM for long-term simulations. Furthermore, as the conventional PIM is mainly developed for first-order problems, this study proposes a generalized framework in the form of an nth order extension of PIM and MPIM, thereby extending their applicability to higher-order differential equations. By positioning the Picard-based iterative family within the broader context of numerical epidemiology, this work sets the foundation for a deeper exploration of how methodological advancements such as the MPIM, Picard Iterative Method for nth order (PIMn) and Multistage Picard Iterative Method for nth order (MPIMn) can contribute to accurate and stable analyses of monkeypox transmission dynamics.

1.2 Motivation for This Work

The re-emergence of monkeypox as a global health threat has emphasized the need for accurate and reliable mathematical models to describe its transmission dynamics, as these models, typically expressed as nonlinear ODEs, rarely yield exact analytical solutions and therefore require robust numerical approximations to provide

meaningful predictions (Agbata et al., 2025; Liu et al., 2023b; Sudsutad et al., 2024). While several numerical techniques such as RK4, VIM, ADM, and the DJM have been extensively applied, they often suffer from drawbacks including computational complexity, slow convergence, and reduced accuracy in highly nonlinear systems (Sweilam et al., 2024).

Similarly, the classical PIM has been widely recognised as an important analytical and numerical approach for solving nonlinear differential equations, particularly due to its theoretical significance in establishing existence and uniqueness of solutions. Previous studies, including the work by Youssef et al. (2024), have demonstrated that the classical PIM can produce acceptable approximate solutions under certain problem settings. However, when compared with the numerical results obtained in the present study (Khairuddin et al., 2025), the classical PIM shows good performance mainly over short computational intervals, while its accuracy decreases over longer time domains due to the accumulation of error. This limitation becomes more critical in complex epidemiological models, where long-term numerical prediction is required to describe disease transmission behaviour.

The motivation for developing the MPIM is also supported by previous work on multistage numerical frameworks. For instance, Han et al. (2024) proposed a multistage-based numerical approach to improve approximation performance by dividing the computational domain into smaller intervals. Inspired by this idea, the present study extends the classical PIM by incorporating a multistage time-marching structure, where the solution interval is partitioned into smaller subintervals and the approximation is updated iteratively at each stage. This modification is introduced to reduce error accumulation, enhance convergence behaviour, and improve numerical accuracy over extended computational intervals.

Therefore, this study proposes the MPIM as an improved PIM numerical framework for solving nonlinear differential equation models. By applying MPIM to the monkeypox transmission model and validating its performance against other well-known nonlinear equations such as Riccati, Duffing, and Newell Whitehead Segel, this research is driven not only by the need to improve predictive tools for monkeypox but also by the broader objective of advancing PIM for nonlinear ODEs and PDEs.

1.3 Problem Statement

Nonlinear mathematical models such as epidemiological transmission models have been studied to predict the spread and dynamic behaviour of infectious diseases such as monkeypox, COVID-19, dengue fever and other communicable diseases. These models are important because they can describe the interaction between different population groups, estimate disease transmission patterns, and support future outbreak prediction. However, due to their nonlinear structure, exact analytical solutions are often difficult to obtain, especially when the model involves several interacting compartments or higher order nonlinear differential equations. Therefore, there is a need to introduce a modified method to reduce the complexity of the previous method and obtain a stable and accurate approximation result.

The classical PIM is a semi-analytical method that has been applied to solve linear and nonlinear differential equations because of its simple iterative structure and direct formulation. However, its application to nonlinear problems is limited in two important aspects. First, the classical PIM generally produces accurate approximations only over short computational intervals, while its accuracy and numerical stability may deteriorate over longer time domains due to error accumulation (Khairuddin et al., 2025). This limitation is particularly critical in epidemiological modelling, such as monkeypox and COVID-19 models, where reliable long-term predictions are required to estimate peak infection levels, outbreak duration, and the effectiveness of intervention strategies. Second, the classical PIM is fundamentally formulated for first-order differential equations (Picard, 1893), which restricts its direct applicability to higher-order ODEs and partial differential equations (PDEs). Higher-order problems are therefore commonly transformed into equivalent first-order systems before applying conventional iterative or numerical schemes. However, this transformation increases the number of dependent variables, computational complexity, and the potential for accumulated numerical errors, making the computational procedure less direct. Therefore, there is a need to improve and extend the classical PIM to enhance its long-term performance and enable the direct treatment of higher-order nonlinear differential equations.

To overcome the limitation of the classical PIM over extended computational

intervals, Mungkasi (2020) introduced a multistage-analytical framework based on the successive approximation method. The method divides a large time domain into smaller subintervals and applies the approximation procedure successively from one stage to the next. To evaluate the effectiveness of the proposed multistage approach, the method was applied to a SEIR infectious disease model with a vaccination strategy. The results showed that the multistage analytical method produced more accurate approximations over a larger time domain than the standard successive approximation and variational iteration methods. In particular, the incorporation of the multistage structure improved the approximation accuracy over an extended time domain, indicating that the successive treatment of smaller subintervals can reduce cumulative errors and enhance long-term numerical performance. However, the method was developed for a first-order SEIR system and was not formulated to solve higher-order nonlinear differential equations directly without order reduction, nor has it been applied to the monkeypox transmission model. For monkeypox transmission modelling, Peter et al. (2022a) employed the fourth-order Runge-Kutta (RK4) method to obtain numerical solutions. Although RK4 provides a numerical benchmark for the model, it does not modify or extend the classical PIM to address its limitations in long-time accuracy, error accumulation, and direct applicability to higher-order differential equations. Likewise, existing classical and fractional-order monkeypox studies have not considered the application of a MPIM for solving the monkeypox transmission model. Therefore, there is a need to extend the classical PIM through the development of PIM*n* and a multistage formulation, namely MPIM*n*, to enable the direct solution of *n*th-order nonlinear differential equations without reducing them into first-order systems.

This thesis addresses three interconnected mathematical challenges: the short-time accuracy limitation of classical PIM in epidemiological models, the absence of a multistage Picard framework for enhanced long-term stability, and the lack of a direct *n*th order Picard formulation. The proposed MPIM improves longterm predictive reliability for nonlinear epidemic models by controlling cumulative errors across successive stages, while the PIM*n* introduces a direct formulation for higher-order differential equations without order reduction. The MPIM*n* further integrates both stability enhancement and higher-order applicability within a

unified framework. By simultaneously addressing stability and higher-order direct solvability, this study contributes a comprehensive extension of the classical PIM for broader mathematical and epidemiological applications.

1.4 Research Questions

In order to address the problem statements and accomplish the goals of this research, three questions have been identified for this study as follows:

a) How does the implementation of the MPIM enhance the classical PIM in solving first order ODEs and PDEs?

b) How does the PIM and MPIM be extended and formulated to directly handle *n*th order differential equations efficiently?

c) How does the proposed Picard-based framework, consisting of MPIM, PIM*n*, and MPIM*n*, perform with respect to accuracy, convergence behaviour, and numerical stability in solving selected nonlinear ODE and PDE models?

1.5 Research Objectives

To address the problem statements and accomplish the research goals, three main objectives have been set for this study as follows:

a) To modify the classical PIM by implementing a MPIM for the first order ODEs and PDEs.

b) To modify the PIM and MPIM to solve direct *n*th order problem.

c) To verify the accuracy, convergence behaviour, and numerical stability of MPIM, PIM*n* and MPIM*n* by applying them to selected nonlinear ODE and PDE models.

1.6 Significance of Study

The significance of this study lies in its contribution to the development of improved PIM for solving nonlinear differential equations, particularly those arising in

epidemiological modelling such as the monkeypox transmission model. Although the classical PIM is well known for its simple formulation and analytical basis, its direct implementation may become less practical when dealing with higher-order problems and long-term approximation intervals. Therefore, this study introduces modified formulations, namely PIM*n*, MPIM*n*, and the multistage implementation of PIM, to support and enhance the existing PIM framework.

The proposed methods are significant as they provide alternative computational approaches that can complement the established numerical methods such as RK4. Instead of replacing existing methods, these formulations aim to improve the flexibility of PIM by extending its application to higher-order differential equations and by using a multistage structure to reduce accumulated approximation errors over longer intervals. Through the application to selected nonlinear models, including the monkeypox transmission model, this study demonstrates how the modified PIM can be used as supportive numerical tools for analysing disease dynamics.

Furthermore, this study contributes to numerical research in applied mathematics by presenting a structured formulation, implementation procedure, and comparative analysis of the proposed methods. The findings may provide a useful reference for future studies involving iterative methods, nonlinear differential equations, and epidemiological models.

1.7 Limitations

This study is limited to the development, implementation, and numerical validation of the PIM, the MPIM, and their higher-order extensions, namely PIM*n* and MPIM*n*, for selected deterministic nonlinear differential equation models with prescribed initial conditions. The study focuses on benchmark equations, higher-order equations, and small- to medium-scale biological and epidemiological systems in order to systematically evaluate the accuracy, convergence behaviour, and numerical stability of the proposed methods within a controlled mathematical setting.

The classical PIM and MPIM are applied to first-order nonlinear models, including the Monkeypox transmission model, the COVID-19 epidemiological model, the biochemical reaction model, the multispecies Lotka-Volterra system, the Riccati equation, and the Newell-Whitehead–Segel equation. These models are

selected because the classical PIM and MPIM are formulated for first-order IVPs. For higher-order models, namely the nonlinear Duffing equation and the Pohlhausen flow equation, the study develops and applies PIMn and MPIMn directly without transforming the original equations into equivalent systems of first-order equations. For consistency in the numerical implementation, a fixed step size of $h = 0.01$ is used throughout the computational simulations. Therefore, this study does not investigate the influence of different step sizes or adaptive step-size strategies on the performance of the proposed methods.

The numerical solutions obtained are evaluated through comparison with selected semi-analytical and numerical methods, including ADM, MADM, and RK4 for the selected ODE systems. For the Monkeypox transmission model, RK8 is additionally used as a higher-order numerical benchmark. Exact solutions are also used where available for the selected benchmark problems. However, CPU execution time was not included in this study. Therefore, the numerical comparison focuses on solution accuracy, convergence behaviour, and numerical stability rather than computational time.

The study does not extend beyond the selected deterministic models to stochastic differential equations, fractional-order models, delay differential equations, or large-scale high-dimensional systems. Such problems involve additional features, including random effects, memory-dependent operators, delayed states, or substantially greater computational complexity, which may require further modification or hybridisation of the proposed methods with other numerical techniques (Han et al., 2024; Sudsutad et al., 2024).

Furthermore, the epidemiological applications in this study are theoretical and computational in nature, where the analysis is based on numerical results generated using Maple. The purpose is to evaluate the accuracy, convergence behaviour, and numerical stability of the proposed PIM when applied to the selected mathematical models. Hence, the numerical results do not fully represent the physical behaviour of actual disease transmission, as the models are analysed using fixed parameter values and initial conditions and do not incorporate several real-world factors, such as demographic variation, spatial movement, behavioural changes, intervention strategies, reporting uncertainty, and time-varying transmission

conditions. Therefore, the findings should be interpreted within the theoretical formulation and numerical setting considered in this study.

1.8 Scope of Study

This study focuses on the development, formulation, and validation of the Picard-based numerical frameworks, namely the PIM, the MPIM, and their higher order extensions PIM*n* and MPIM*n*. The scope includes the application of PIM and MPIM to selected first order nonlinear ODEs and PDES models, specifically the Monkeypox transmission model, the COVID-19 epidemiological model, the biochemical reaction (Michaelis–Menten) model, the multispecies Lotka–Volterra system, the Riccati equation, and the Newell Whitehead Segel equation. For higher order differential equations, the study extends the framework through PIM*n* and MPIM*n*, which are applied to benchmark problems such as the nonlinear Duffing equation and the Pohlhausen flow equation. The methodology is implemented through an iterative computational procedure in Maple, where the Picard-based approximations are generated symbolically and evaluated numerically over specified time intervals. For MPIM and MPIM*n*, the computation is carried out using a multistage time-marching approach, in which the solution interval is divided into smaller subintervals and the initial condition is updated at each stage. This implementation is used to improve the long-time numerical behaviour of the classical PIM, particularly for nonlinear models. The numerical performance of the proposed methods is evaluated in terms of accuracy, stability, convergence behaviour, and absolute error comparison using appropriate numerical and semi-analytical reference methods. For the COVID-19 model, biochemical reaction model, and multispecies Lotka–Volterra equations, the PIM and MPIM results are compared with ADM, MADM, and RK4. For the Monkeypox transmission model, the comparison additionally includes RK8 as a higher-order numerical benchmark because the model is simulated over an extended time interval with a large number of time steps. The ADM and MADM are used as supplementary comparison methods and do not form part of the proposed Picard-based framework. The number of iterations and step size are restricted to certain values to ensure computational feasibility, symbolic manageability, and stable numerical output in Maple. A very large number of

iterations may produce lengthy symbolic expressions, increase computational cost, and lead to memory or simplification difficulties. Similarly, the selected step size must be sufficiently small to improve accuracy, but not too small to avoid excessive computational time. Therefore, the iteration number and step size are chosen based on a balance between accuracy, convergence behaviour, and computational efficiency. All implementations are carried out using Maple for symbolic and numerical computation. The study is confined to deterministic nonlinear models with given initial conditions and does not cover stochastic, fractional-order, delay, or large-scale high-dimensional systems. The outcomes of the study are also influenced by the selected step size, number of iterations, time interval, model parameters, and the capability of Maple to handle symbolic expressions generated by the iterative schemes.

1.9 Thesis Outline

This thesis is organized into seven chapters. In the first chapter, a general introduction, motivation for this work, problem statement, research question, research objective, significance of the study, limitation and thesis scope are given.

In Chapter 2, a structured and comprehensive review of the relevant literature is presented to establish the theoretical and methodological foundation of this study. The chapter begins with the background of Monkeypox disease and the mathematical modelling of infectious diseases, followed by a discussion of the Monkeypox transmission model proposed by Peter et al. (2022a). It then reviews the fundamental principles and formulation of the PIM for solving first-order differential equations, together with its implementation in selected nonlinear differential equation systems. The evolution of the Modified PIM, including the multistage concept and related extensions, is also reviewed. Through critical analysis of previous studies, the strengths and limitations of existing iterative approaches are identified, thereby highlighting the research gap that motivates the development of the proposed modified PIM in this thesis.

In chapter 3, the mathematical development of the proposed numerical methods is presented. This chapter introduces the formulation of the MPIM, PIM*n* and MPIM*n*, beginning with the fundamental principles of the classical PIM. The

limitations of the standard PIM are discussed, followed by the derivation of the MPIM framework is designed to improve accuracy, stability, and global convergence by applying the Picard iteration across successive subintervals. The chapter also develops the PIM*n* and MPIM*n*, which enables direct solution of higher order differential equations without reducing them to first order systems. Convergence analysis, iterative structures, and implementation procedures for MPIM, MPIM*n* and MPIM*n* are thoroughly detailed.

In Chapter 4, the PIM*n* is first introduced and derived in detail to demonstrate how the general formulation for higher order IVPs is constructed from the integral representation of differential equations. The method is then verified using two initial value problem examples to assess its accuracy, convergence behaviour, and computational reliability. Next, the PIM and MPIM are applied to the nonlinear first order Riccati equation to evaluate their performance in solving classical nonlinear ODEs. The comparison with the exact solution and RK4 benchmark shows that MPIM provides better accuracy and stability than the standard PIM. Furthermore, the PIM*n* and MPIM*n* are applied to nonlinear higher order problems, namely the Duffing equation and the Pohlhausen flow equation. The numerical results show that MPIM*n* maintains small absolute errors and strong agreement with RK4, while PIM*n* is more reliable over short time intervals. Overall, this chapter fulfils the first, second and third research objective and further validates the applicability of the proposed Picard-based methods for solving nonlinear ODEs.

In Chapter 5, the applicability of the PIM and MPIM is extended to nonlinear PDEs. This chapter focuses on the Newell Whitehead Segel equation, a nonlinear reaction diffusion model, to assess whether Picard-based methods can accurately solve nonlinear PDEs. The numerical results are compared with the exact solution and the RK4 benchmark computed using Maple. The comparison shows that MPIM provides better accuracy and stability than the standard PIM, particularly over longer time intervals. Overall, this chapter further validates the applicability of the proposed Picard-based methods for nonlinear PDEs and contributes to fulfilling the first research objective.

In Chapter 6, the PIM and MPIM are applied to various nonlinear systems of ODEs, including the Monkeypox transmission model, the COVID-19 model, the

biochemical reaction (Michaelis-Menten) system, and the multispecies Lotka-Volterra equations. To provide a broader comparative assessment, the ADM and MADM are also included as supplementary semi-analytical methods for all four applications. The numerical approximations obtained using PIM, MPIM, ADM, and MADM are compared with RK4 results generated using Maple. For the Monkeypox transmission model, RK8 is additionally included as a higher-order numerical benchmark because the simulation involves a long computational interval and a large number of time steps. The comparative results demonstrate the improved long-time numerical performance of MPIM relative to the classical PIM and contribute to fulfilling the first and third research objectives of the study.

In the final chapter, the main findings of the thesis are summarised, highlighting the contributions and significance of the proposed methods. This chapter also outlines several potential directions for future research, offering recommendations for further extensions and applications of the developed numerical techniques.

CHAPTER 2
LITERATURE REVIEW

2.1 Introduction

Mathematical modelling plays a significant role in understanding the transmission dynamics and control strategies of infectious diseases such as Monkeypox. Epidemiological models are typically formulated as nonlinear systems of differential equations, where exact analytical solutions are often difficult to obtain. Consequently, numerical approximation methods are required to analyse the behaviour of these systems over time. This chapter begins by reviewing the background of Monkeypox disease, the mathematical modelling of infectious diseases, and the Monkeypox transmission model proposed by Peter et al. (2022a), which serves as the foundational model in this study.

The chapter then focuses on the development of the Picard Iterative Method (PIM), including its historical background, formulation for first order differential equations, and applications in existing literature. Modifications and extensions of PIM, particularly the multistage concept, are also reviewed to provide the theoretical basis for the development of the Modified PIM proposed in this thesis. The chapter concludes by identifying the existing research gaps that motivate the proposed method.

2.2 Background of Monkeypox Disease

Monkeypox is an emerging zoonotic viral infection caused by the monkeypox virus, belonging to the Orthopoxvirus genus within the Poxviridae family (Tajudeen et al., 2023). It's closely related to the variola virus that causes smallpox and has become a major global health concern due to its potential to cause human outbreaks (Chen et al., 2025). Although first identified in laboratory monkeys in 1958, human cases were officially recorded in the Democratic Republic of Congo in 1970, marking the beginning of the disease's documented epidemiological history (Di Giulio & Eckburg, 2004).

Traditionally, monkeypox has been endemic to remote regions of Central and West Africa, where transmission is facilitated by close contact with infected wildlife reservoirs such as rodents and other small mammals (Reynolds et al., 2019). The virus spreads to humans primarily through zoonotic exposure via bites, scratches, or contact with infected bodily fluids or lesions but human to human transmission can also occur (Harris, 2022). Human transmission typically happens through respiratory droplets during prolonged face to face interactions, direct contact with skin lesions, or contact with contaminated materials such as bedding or clothing. The clinical progression commonly begins with fever, chills, headache, muscle pain, and lymphadenopathy, followed by the development of a characteristic vesiculo-pustular rash that spreads across the body (Yadav et al., 2025).

The global landscape of monkeypox changed significantly in 2022, when multiple non-endemic countries reported widespread community transmission, including regions in Europe, North America, and Asia (Wu et al., 2025). These outbreaks highlighted the importance of understanding monkeypox dynamics beyond endemic settings, especially given the increased human mobility, urbanization, and potential changes in reservoir–host interactions. The disease's capacity to spread both zoonotically and through human networks makes it essential to develop robust models that account for complex interactions and multiple transmission pathways (Yadav et al., 2025).

Mathematical modelling has therefore become a critical tool for assessing monkeypox transmission potential, predicting outbreak size, and designing effective control strategies such as vaccination, isolation, surveillance, and targeted public health interventions (Agbata et al., 2025). Because monkeypox involves multiple epidemiological stages including exposure, latent infection, symptomatic infection, and recovery compartmental models using ODEs are widely employed by researchers. These models help describe the temporal evolution of the disease within human and animal populations while accounting for parameters such as infection rates, contact patterns, recovery rates, and mortality.

Given the recurrent outbreaks and the increasing complexity of transmission channels, the need for accurate, efficient, and reliable numerical methods to solve monkeypox models has become more important than ever. Improvements in

numerical approximation techniques, particularly for nonlinear and multi-compartment ODE systems, play a significant role in enhancing the predictive power and reliability of these mathematical models.

2.3 Mathematical Modeling of Infectious Diseases

Mathematical modeling is one of the most widely used approaches for understanding infectious disease dynamics, as it allows researchers to quantify how diseases spread within populations and to assess the likely effectiveness of interventions. The earliest and most influential models are the compartmental frameworks, such as the SIR and SEIR models, which classify individuals according to their disease status. These simple systems of nonlinear differential equations have provided valuable insight into epidemic trajectories and control thresholds, such as the basic reproduction number, yet they also rely on assumptions of homogeneous mixing and exponentially distributed transition times that may oversimplify real-world conditions (Hunter & Kelleher, 2022; Tolles & Luong, 2020).

In recent years, advances in mathematical epidemiology have expanded these models to incorporate additional biological and social mechanisms. Researchers have developed extensions that account for vaccination, waning immunity, contact heterogeneity, and spatial effects, as well as fractional-order and fractal-fractional operators that capture memory and nonlocal transmission dynamics (Alaje & Olayiwola, 2023; Alalhareth et al., 2024). These modifications aim to address the limitations of classical models by producing outcomes that align more closely with observed epidemic data. However, such complexity often results in highly nonlinear systems that cannot be solved analytically, creating a strong demand for accurate and efficient numerical methods to approximate solutions.

The re-emergence of monkeypox since 2022 has spurred renewed interest in epidemiological modeling. Several studies have formulated nonlinear differential equation models to represent human-to-human and zoonotic transmission routes, while others have introduced fractional-order frameworks to capture time-dependent memory effects and evaluate vaccination strategies (Gunasekar et al., 2025; Peter et al., 2022a). These models provide important qualitative insights through stability and sensitivity analyses but, in practice, exact closed-form solutions are rarely

attainable. Numerical simulation therefore becomes essential to generate predictions that can guide public health decision-making. This challenge underscores the importance of developing novel numerical methods capable of handling nonlinear epidemiological models such as monkeypox, thereby forming the foundation and motivation for the present study.

2.4 Monkeypox Transmission Model Proposed by Peter et al. (2022a)

Several mathematical models have been developed to understand the transmission dynamics of the Monkeypox virus. Among these, the deterministic compartmental model proposed by Peter et al. (2022a) provides a comprehensive framework that captures both human-to-human and rodent-to-human transmission pathways. The model incorporates isolation measures and performs detailed analytical investigations, including equilibrium analysis and bifurcation behaviour. Due to its well-structured formulation and rigorous mathematical analysis, this model serves as a suitable foundation for further numerical investigation.

2.4.1 Model Formulation

Peter et al. (2022a) proposed a deterministic compartmental model to describe the transmission dynamics of the Monkeypox virus involving interactions between human and rodent populations. The model consists of two interacting populations, namely humans and rodents, and is formulated as a system of nonlinear ordinary differential equations.

The human population is subdivided into five mutually exclusive compartments: susceptible humans $S_h(t)$, exposed humans $E_h(t)$, infected humans $I_h(t)$, isolated humans $Q_h(t)$, and recovered humans $R_h(t)$. The rodent population is subdivided into three compartments: susceptible rodents $S_r(t)$, exposed rodents $E_r(t)$, and infected rodents $I_r(t)$. The inclusion of the isolated compartment $Q_h(t)$ allows the model to capture intervention strategies such as isolation of infected individuals.

The transmission dynamics are governed by nonlinear incidence terms representing:

- Rodent-to-human transmission,

- Human-to-human transmission,
- Rodent-to-rodent transmission.

The full model is described by a system of eight nonlinear ordinary differential equations, incorporating recruitment, progression, recovery, natural death, and disease-induced mortality rates.

Peter et al. (2022a) further conducted a rigorous mathematical analysis of the model. The biological feasibility region was established and shown to be positively invariant. The monkeypox-free equilibrium and endemic equilibrium were derived explicitly. The basic reproduction number R_0 was computed using the next-generation matrix approach. Local and global stability analyses of the disease-free equilibrium were performed, and conditions for the existence of backward bifurcation were established using centre manifold theory. Additionally, sensitivity analysis and numerical simulations were carried out to illustrate the impact of key epidemiological parameters on disease transmission dynamics.

Their findings highlighted the significant role of human-to-human transmission and demonstrated that isolation of infected individuals reduces disease spread.

2.4.2 Method Used in Previous Studies

In the original study by Peter et al. (2022a), the monkeypox transmission model was formulated as a deterministic system of nonlinear ordinary differential equations involving human and rodent populations. The authors established the positivity and boundedness of solutions and derived both the monkeypox-free equilibrium and endemic equilibrium states. The basic reproduction number R_0 was computed using the next-generation matrix method. Local and global stability analyses were performed using linearization techniques and Lyapunov-based approaches, while the occurrence of backward bifurcation was investigated via centre manifold theory. Numerical simulations were carried out using classical numerical integration schemes to illustrate the theoretical findings. Their results highlighted the significant role of human-to-human transmission and demonstrated that isolation of infected individuals reduces disease spread.

Following this work, several researchers extended the monkeypox transmission framework using alternative mathematical techniques. For instance, Peter et al. (2022b) further developed a fractional order version of the model in their study on fractional mathematical modeling of monkeypox transmission dynamics, where Caputo fractional derivatives were introduced to capture memory effects in disease spread. In particular, Alharbi et al. (2022) developed a fractional-order version of the monkeypox model using Caputo fractional derivatives to incorporate memory effects and investigated stability properties under non-integer dynamics. Similarly, Liu et al. (2023a) and Khan et al. (2024) employed fractional epidemiological modeling approaches to investigate vaccination impacts and real-data fitting under fractional derivatives. Okyere and Ackora-Prah (2023) analyzed the monkeypox model using Atangana Baleanu fractional derivatives and examined stability properties and numerical behavior under fractional operators. Majee et al. (2023) incorporated treatment and vaccination controls within a fractional order framework and analyzed transcritical bifurcation phenomena. In addition to fractional approaches, Akkilic et al. (2024) utilized a radial basis deep neural network with Bayesian regularization optimization to approximate solutions of the monkeypox transmission model. Furthermore, Yapışkan et al. (2023) proposed optimal preventive strategies using Pontryagin's Maximum Principle combined with Runge Kutta numerical schemes. These studies demonstrate the adaptability of the original monkeypox transmission model and reveal a growing interest in employing advanced analytical, fractional, and computational techniques to explore its dynamical behavior.

Following that, Movaheedi and Rahmani (2026) conducted a comparative numerical study of the monkeypox transmission model using the fourth-order Runge-Kutta (RK4) and Backward Euler methods, concluding that RK4 provides higher short-term accuracy while the Backward Euler method demonstrates superior stability for stiff epidemiological systems and long-term simulations. Manivel et al. (2025) developed a fractional-order co-infection model for monkeypox and HIV using the Atangana–Baleanu fractional operator, where stability was examined via fixed-point theory and Picard's successive approximations, and numerical simulations were implemented using a Newton polynomial-based method to analyze the impact

of fractional parameters on co-infection dynamics. Sweilam et al. (2025) proposed a crossover fractional mathematical model of monkeypox transmission based on the Ψ-Caputo derivative and employed Ψ-nonstandard finite difference schemes alongside a modified Euler Maruyama method to numerically investigate fractional, variable-order, and stochastic dynamics, validating the approach through comparative analysis and real data simulations.Yeshwanth and Kumbinarasaiah (2025) analyzed the monkeypox transmission model using a Modified Hermite Wavelet Collocation Method, where fractional differential equations were transformed into an algebraic system via an operational matrix of integration and solved using the Newton–Raphson method, with numerical results compared against RK4 and NDSolver to demonstrate the accuracy and efficiency of the wavelet-based approach. Suantai et al. (2023) investigated the fractional-order monkeypox transmission model using Caputo derivatives and employed an artificial intelligence–based scaled conjugate gradient neural network to obtain numerical solutions, validating the stochastic approximations against reference Adams method results through regression, error analysis, and state-transition performance measures. Bansal et al. (2025) investigated the monkeypox transmission dynamics under vaccination effects using an Atangana Baleanu Caputo fractional derivative framework, where existence and uniqueness were established via Banach's fixed point theorem, the basic reproduction number was derived using the next-generation matrix approach, and numerical simulations were performed using the fractional Adams Bashforth method to analyze the impact of model parameters on disease dynamics.

Despite the extensive analytical and computational developments surrounding the monkeypox transmission model, the majority of existing studies have focused on classical stability analysis, fractional-order extensions, data-driven parameter estimation, artificial intelligence-based approximations, or optimal control frameworks. While these approaches provide valuable insights into the qualitative and numerical behavior of the system, the application of iterative analytical–numerical schemes such as the PIM and its multistage extension (MPIM) has not been reported for the monkeypox transmission model.

In particular, no existing study has investigated the convergence characteristics, long-term stability performance, or computational efficiency of PIM when applied

directly to the nonlinear monkeypox transmission system. This methodological gap motivates the present study, which aims to examine the applicability and robustness of the PIM and the MPIM in solving the monkeypox transmission model. By evaluating their accuracy and stability performance against classical numerical benchmarks, this study seeks to provide an alternative and reliable numerical framework for epidemiological modeling of monkeypox dynamics.

2.5 Picard Iterative Method (PIM)

This section provides a detailed overview of the Picard Iterative Method (PIM), including its historical development, mathematical formulation for first order ordinary differential equations (ODEs), and its applications in existing literature. The discussion establishes the theoretical foundation required for the subsequent development of the proposed modifications in this study.

2.5.1 Historical Development of PIM

The PIM originated in the late 19th century through the foundational work of Émile Picard. His pioneering contributions, particularly in the context of differential equations, laid the groundwork for iterative approaches to solving nonlinear problems. Picard's initial formulation was centered around the iteration of fixed points, which became formalized through the analysis of integral equations (Chen et al., 2022). This methodology was a significant advancement as it allowed for the systematic approximation of solutions and relates to the Banach Fixed Point Theorem. This theorem asserts the existence and uniqueness of fixed points for contraction mappings in complete metric spaces, encapsulating the essence of PIM (Ahmad et al., 2021). The application of this theory to the Picard method illustrates its robustness, ensuring convergence under specific conditions related to contraction.

As the field of numerical analysis has evolved, the application of PIM has expanded significantly, particularly from 2020 to 2023, marking a period rife with innovative methodologies and applications pertinent to nonlinear ODEs. Researchers have harnessed PIM not only for traditional applications in physics and engineering but also for complex systems within biology and epidemiology. For instance,

contemporary studies demonstrate PIM's utility in simulating chemical kinetics, where the method assists in resolving the challenges posed by stiff ODEs encountered in reactive-flow simulations (Ji et al., 2021). In this context, sophisticated numerical solvers have emerged that leverage PIM principles alongside new computational techniques like Physics-Informed Neural Networks (PINNs) to enhance efficiency and accuracy in solving ODEs (Huang & Seinfeld, 2022; Wu et al., 2023). This synthesis of traditional iterative methods with modern computational techniques underscores the method's continued relevance and adaptability in tackling complex differential equations in various scientific domains.

Despite its successes, classical PIM is not without limitations. In scenarios involving stiff or higher order ODEs, the classical approach can struggle, often leading to slow convergence rates. Therefore, researchers have explored modifications to the original Picard framework. Innovations such as multistage PIM and various acceleration techniques have been proposed to enhance the convergence speed and stability of the method (Azevedo et al., 2020). For example, techniques such as the accelerated spectral deferred correction method have shown promise by integrating sophisticated linear algebra strategies to improve convergence in iterative processes solving fractional differential equations (Chen et al., 2022). These advancements address fundamental constraints of the conventional PIM, allowing it to manage diverse applications ranging from chemical kinetics to dynamical systems modeling, where conventional approaches may fall short.

In summary, the historical development of the Picard Iterative Method has transitioned from its inception in Picard's seminal work to a highly relevant technique in modern numerical analysis. The convergence guarantees provided by the Banach Fixed Point Theorem have been instrumental in establishing the method's credibility. Recent innovations reflecting the need for more efficient variants have emerged to address classical limitations, particularly in the context of stiff and higher order ODEs. Collectively, these facets underscore the method's evolution and highlight ongoing efforts to optimize its application across various scientific disciplines, affirming Picard's legacy in the landscape of functional analysis and numerical methods.

2.5.2 PIM for First order ODEs

The integral form of a first order initial value problem (IVP) is foundational to the analysis and solution of such problems in mathematical analysis and numerical methods. The general integral representation can be defined as

$$y(t) = y_0 + \int_{t_0}^{t} f(s, y(s))ds, \tag{2.1}$$

where y_0 represents the initial condition and $f(s, y(s))$ is a function dependent on both the independent variable s and the unknown function $y(s)$. This formulation serves as the basis for the (PIM), which is a powerful tool for solving IVPs. It establishes a framework where the solution can be approximated through a series of iterative refinements starting from the initial condition y_0 (Tafakkori-Bafghi et al., 2020). The iterative nature of this method allows for the successive approximations of the solution to converge under suitable conditions, particularly the Lipschitz condition, which ensures that the function f responds linearly to changes in y, hence guaranteeing the uniqueness of the solutions produced through the method (Lyons et al., 2017).

The classical Picard iteration method constructs successive approximations via the fixed-point iteration scheme. This approach begins with an initial guess, typically taken as the initial condition, and iteratively substitutes the current approximation into the integral form to yield a new approximation. This process can be mathematically defined as

$$y_{n+1}(t) = y_0 + \int_{t_0}^{t} f(s, y_n(s))ds, \tag{2.2}$$

where y_n denotes the nth approximation (Pollock et al., 2019). Notably, each iteration seeks to improve the accuracy of the previous estimate by using it in the integral formulation, thereby refining the approximation until a satisfactory degree of accuracy is achieved. A critical aspect of the convergence of the Picard iterations is captured by the Banach fixed-point theorem, which asserts that, under appropriate circumstances (namely, Lipschitz continuity), the sequence of approximations produced will converge to a unique fixed point, which corresponds to the solution of the IVP (Lyons et al., 2017; Zeng & Zhang, 2020).

In ensuring the convergence of the Picard iteration method, mathematical

properties such as Lipschitz continuity and contraction mapping play crucial roles. A function $f(s,y)$ satisfies the Lipschitz condition if there exists a constant L such that $|f(s,y_1) - f(s,y_2)| \leq L|y_1 - y_2|$ for all y_1 and y_2. This property guarantees that the iterations draw sufficiently close together promoting convergence: as the iterations progress, deviations between subsequent approximations reduce, ensuring convergence to the unique solution (Liu et al., 2025). Moreover, computational challenges, such as the potential increase in complexity of the fixed-point problem as the dimensionality increases, lead to modifications of the standard Picard iteration method, including the development of accelerated variants. These include approaches such as Anderson acceleration and Jacobi-Picard methods, which aim to enhance convergence rates and improve computational efficiency for practical applications in various fields (Almeida et al., 2018; Pollock et al., 2019).

The standard Picard iteration method has found modern applications across diverse domains, reflecting its continued relevance and adaptability to contemporary challenges in solving nonlinear ODEs. Recent studies highlight its utilization in engineering, biological modeling, and epidemiology, where Picard iterations contribute to the accurate modeling of dynamic systems and the spreading of diseases (Filobello-Nino et al., 2024; Youssef et al., 2024). For instance, the application of a modified Picard iteration method has provided reliable solutions in the context of nonlinear dynamics in biological systems, demonstrating not only accuracy but also the ability to navigate the inherent complexities of modeling real-world phenomena (Tafakkori-Bafghi et al., 2020). However, the method is not without limitations, including potential convergence issues and computational burdens, particularly for high-dimensional ODEs. To address these challenges, researchers have proposed several advanced strategies that leverage the foundation laid by the Picard approach while optimizing computational performance, thereby ensuring the continued applicability of this classical method in addressing contemporary mathematical modeling challenges (Nwaigwe & Mishra, 2024).

2.5.3 Application PIM in Literature

The PIM, an iterative technique for solving differential equations, has garnered considerable attention for its versatility in addressing various forms of

ODEs. It has been effectively applied to linear initial value problems (IVPs), where its capacity to generate solutions aligns well with the linear dynamics' simplicity. Recent advancements include the work of Hamid et al. (2022), who introduced a hybrid Picard-Chelyshkov polynomial method directed at nonlinear ODE models, demonstrating its utility in addressing complex nonlinear dynamics. In chemical kinetics, the method has been utilized to model reaction rates accurately; for instance, it has effectively solved nonlinear equations arising from reaction-diffusion phenomena (Abdel-Gawad et al., 2017). Moreover, in ecological models such as predator-prey systems, the Picard method has enhanced the understanding of dynamic interspecies interactions, with applications noted in modeling population dynamics (Hamid et al., 2022). Thus, the method's applicability spans various fields, showcasing its value in both linear cases and intricate nonlinear scenarios.

The simplicity and conceptual clarity of the Picard method are among its greatest strengths. Notably, the method relies on the Banach fixed-point theorem, which underpins its mathematical foundation established through contraction mappings (Nguyen et al., 2018). This characteristic provides a systematic approach to generating analytic-type iterative approximations, enhancing its appeal for solving ODEs. In practical applications, this method often yields remarkably precise results, largely due to its iterative nature that approximates solutions to a high degree of accuracy. Furthermore, the ability of the Picard method to tackle integral equations through fixed-point iterations showcases its versatility beyond traditional linear contexts (Azevedo et al., 2020). Recent advancements also highlight the application of Picard iterations in modern computational techniques (Nguyen et al., 2018).

However, key limitations of the standard Picard method present challenges. The method exhibits slow convergence when applied to stiff or strongly nonlinear systems, significantly hampering its efficiency in dynamic scenarios characterized by rapid changes (Yao et al., 2021). Moreover, its sensitivity to step size or interval length implies that small perturbations can disproportionately affect accuracy, restricting its practical utility in sensitive applications. The classical Picard approach is predominantly tailored for first order ODEs; thus, higher order equations require transformation into systems of first order equations, complicating its usage (Abbass et al., 2025). In addition, the accuracy of solutions is intricately tied to the complexity

of the integral form being evaluated, often resulting in challenges when higher precision is required in numerical contexts (Yao et al., 2021).

To address the limitations of the standard Picard method, recent literature has seen an emergence of innovative approaches (2020-2025). Various modified Picard iterative methods have been proposed, such as accelerated PIM, and hybrid decomposition-Picard methods, aimed at enhancing convergence rates and broader applicability (Nwaigwe & Mishra, 2024). These adaptations seek to mitigate slow convergence through relaxation techniques and optimized iterations while retaining the foundational benefits of the original Picard approach (Yao et al., 2021). For instance, the application of the Anderson acceleration method in conjunction with modified Picard iterations has shown effectiveness in enhancing numerical performance, particularly in scenarios demanding rapid convergence (Zhao et al., 2025). Such advancements signify a crucial evolution of the Picard framework, leading to increased flexibility and efficiency in addressing modern computational challenges across various fields, from chemical kinetics to ecological modeling.

2.6 Modified Picard Iterative Method

This section presents the development of the Modified PIM as an enhancement of the classical PIM. It first reviews the evolution of modified iterative methods in the literature, followed by an introduction to the multistage concept designed to improve numerical stability and convergence over extended time intervals. The section concludes with a discussion on the key modifications and extensions of the Picard Iterative Method that underpin the proposed methodology in this study.

2.6.1 Evolution of Modified Iterative Method

The evolution of modified iterative methods for solving nonlinear ODEs has seen significant advancements, particularly in methodologies such as the Variational Iteration Method (VIM), Homotopy Perturbation Method (HPM), and the Adomian Decomposition Method (ADM). These techniques have been enhanced to improve their applicability in complex and stiff systems. Recent studies highlight the effectiveness of VIM and its modified variants in addressing stiff ODEs,

demonstrating simpler implementations compared to classical methods, thereby allowing for more efficient tackling of various linear and nonlinear stiff ODEs (Atay & Kilic, 2013). Furthermore, the HPM has evolved to mitigate slow convergence issues traditionally associated with classical methods by integrating perturbation techniques that enable faster and more stable solutions, especially for higher order ODEs (Azhari et al., 2025). The ADM has also been further developed to provide analytic solutions in the form of convergence series, leading to improved accuracy and efficiency relative to traditional approaches (Hunaiber et al., 2023; Ziada, 2022).

The necessity for these modifications arises primarily from the inherent challenges posed by classical iterative methods, including slow convergence rates, stability issues, and limitations in handling stiff systems. Classical methods often encounter difficulties with nonlinear terms due to their complexity, leading to oscillations and divergence from accurate solutions. Stiffness in systems—particularly in multi-compartment models and higher order ODEs—presents significant computational challenges, often requiring adaptive strategies or hybrid methods to ensure convergence and stability (Priyadarsini et al., 2025; Sahani & Sah, 2024). Recent explorations reveal the inadequacies of classic methods, justifying the evolution of these iterative techniques to accommodate the complexities often encountered in fields like pharmacokinetics and biological modeling (Hasegawa & Duffull, 2018).

Recent advancements (2020–2025) in iterative methods have aimed at enhancing convergence, stability, and accuracy while effectively managing highly nonlinear or biological models. Researchers have introduced various hybrid and multistage versions of existing methods to address the shortcomings of classical approaches. For instance, a novel hybrid framework combining neural networks with higher order ODE solvers has demonstrated substantial improvements in accuracy and computational speed when applied to stiff and nonlinear differential equations (Murugesh et al., 2025). Additionally, new algorithms like the Spectral Deferred Correction method have shown potential for accelerating convergence through mechanisms such as the Generalized Minimal Residual method, enhancing the reliability of numerical solutions for fractional differential equations (Chen et al., 2022). These developments represent a significant shift towards more adaptive

computational techniques tailored to the behaviors of complex and nonlinear systems.

This ongoing evolution is closely linked with the motivation to develop the modified PIM. As the limitations of classical Picard methods such as slow convergence and sensitivity to initial conditions become increasingly apparent, similar issues that drive enhancements in VIM, HPM, and ADM provide a compelling basis for the development of MPIM. Modifying the Picard method to incorporate rapid convergence techniques and refined stability assessments aligns with contemporary trends in numerical analysis that prioritize efficiency and accuracy in simulating nonlinear dynamic systems. By addressing the iterative refinement needs that have sparked innovations in existing methodologies, the MPIM aspires to establish its relevance and utility in effectively solving modern complex differential equations.

2.6.2 Introduction to Multistage Concept

The multistage concept has become increasingly significant for enhancing the robustness and accuracy of numerical methods for ODEs, particularly in the context of multistage Runge-Kutta methods, multistage Variational Iteration Methods (VIM), and multistage Adomian Decomposition Methods (ADM). Recent contributions to the field illustrate improved convergence and stability properties derived from multistage formulations. For instance, Prabowo and Mungkasi (2021) introduced a multistage successive approximation method for solving Riccati differential equations, addressing stability and error propagation through strategic refinements at each iteration subinterval. These advancements collectively underscore the efficacy of multistage approaches compared to traditional single-interval methods, resulting in improved local accuracy and reduced error accumulation issues.

The mathematical principles underpinning multistage strategies primarily aim to enhance local accuracy and mitigate global error propagation in ODE problems. By subdividing the computational domain into multiple smaller intervals, these methods effectively manage the issues associated with stiff equations. For example, in the context of the multistage ADM framework, subinterval iterations effectively control rounding and truncation errors, thereby preserving stability throughout the numerical process. Kaur et al. (2024) illustrated that adopting multistage principles leads to

improved accuracy in fine-grained computations, allowing for a deeper understanding of complex nonlinear behaviors. This dimensionality reduction stabilizes numerical methods over expansive domains and reinforces the theoretical foundations required for tackling highly nonlinear or stiff systems in computational applications.

Connecting multistage frameworks to iterative improvements in solving nonlinear ODEs reveals significant advantages in managing truncation errors and achieving coherent approximations. The higher order numerical stability provided by multistage methods is critical when approximating integral or derivative terms across different discretization points. The work of Pollock et al. (2019) emphasizes how Anderson acceleration enhances the Newton iteration process for nonlinear solvers, significantly improving convergence rates. By implementing error correction stages systematically, these frameworks yield stronger stability and superior approximation capabilities compared to traditional single-interval strategies. Such insights enrich the algebraic structure of solutions derived from multistage iterative approaches, leading to more resilient numerical techniques.

In contemplating the construction of a Multistage Picard Iterative Method (MPIM), the core motivation arises from consolidating the advantages of multistage frameworks with the established principles of the Picard iterative method. This integration of multistage techniques is anticipated to enhance convergence properties while alleviating sensitivities related to interval constraints, thus addressing critical limitations exhibited by classical Picard methods. By embracing this multifaceted approach, MPIM is positioned to offer substantial improvements in solving intricate nonlinear systems, particularly those relevant in fields such as epidemiology and control theory. As research evolves towards sophisticated iterative frameworks, the anticipated benefits of greater numerical stability and efficiency position MPIM as a cornerstone for advancing computational methodologies in mathematical sciences.

2.6.3 Modifications and Extensions of Picard Iterative Method

Over the years, researchers have recognized the limitations of the classical PIM when applied to highly nonlinear or large-scale problems. As a result, several modifications and extensions have been developed to improve its accuracy, convergence rate, and applicability. One of the notable approaches is the

Chebyshev–Picard iteration, which combines Picard's successive approximation with Chebyshev polynomials to achieve faster convergence for delay differential equations and complex dynamical systems (Ma & Liu, 2024). More recently, deep learning frameworks have integrated Picard iterations into neural network architectures, enabling the method to handle high-dimensional nonlinear PDEs (Boussange et al., 2023). These innovations highlight how Picard's classical framework has inspired modern numerical techniques that can adapt to increasingly complex mathematical models.

Another direction of modification is the introduction of multistage and partitioned iterative methods, which divide the solution interval into subintervals and apply PIM step by step. This strategy improves stability and accuracy by reducing error accumulation across long time spans, making it particularly suitable for nonlinear ODEs that describe real-world phenomena. In applied mathematics, such approaches have been shown to outperform the classical PIM, especially in problems where traditional techniques converge too slowly or fail to capture nonlinear dynamics effectively (Ansari & Akrami, 2024). These developments illustrate the potential of multistage Picard-based schemes to serve as efficient alternatives to conventional numerical methods such as Runge–Kutta or ADM.

Despite these advances, there remain relatively few studies that systematically apply modified PIM to epidemiological models. The majority of existing applications have focused on physics or engineering problems, while the use of iterative schemes in mathematical epidemiology has largely been limited to qualitative analysis rather than numerical approximation. This gap provides the motivation for the present study to develop a MPIM specifically tailored to nonlinear disease transmission models such as monkeypox, and to validate its performance on benchmark nonlinear equations including the COVID-19 model, the biochemical reaction model, the multispecies Lotka-Volterra equation, Riccati, and Newell–Whitehead–Segel. By bridging the theoretical structure of PIM with the practical demands of epidemiological modeling, MPIM is expected to extend the scope of iterative numerical techniques in both mathematics and applied health sciences.

2.7 Applications in Epidemiological Models

Mathematical models in epidemiology have been widely supported by numerical methods, particularly when exact analytical solutions cannot be obtained for nonlinear systems. Classical compartmental models such as SIR and SEIR often lead to sets of nonlinear differential equations, which become increasingly complex when additional factors such as vaccination, waning immunity, or zoonotic transmission are incorporated (Ghosh & Ghosh, 2023). In such cases, numerical schemes provide approximate solutions that enable researchers to simulate disease dynamics, predict outbreak sizes, and evaluate intervention strategies. Over time, numerical techniques including Runge–Kutta, ADM, VIM), and DJM have been applied to a variety of epidemiological models, each with distinct advantages and drawbacks (Abed & AL-Jawary, 2021; Biazar, 2006; Rafei et al., 2007).

Recent studies have focused on applying advanced numerical methods to emerging diseases such as monkeypox, which re-emerged as a significant global health issue in 2022. Several researchers have formulated fractional-order and fractal-fractional models of monkeypox to capture memory effects and nonlocal interactions, demonstrating that fractional operators can provide a closer fit to real-world transmission dynamics compared to classical integer-order models (Agbata et al., 2025; Gunasekar et al., 2025). These models rely heavily on numerical simulations for validation, as closed-form solutions are generally unattainable. Such reliance on numerical approximation highlights both the importance and the challenge of selecting methods that balance accuracy, efficiency, and stability.

Despite this progress, the application of iterative schemes such as the PIM remains underexplored in epidemiological contexts. While Picard-based methods have been studied in physics, engineering, and high-dimensional PDEs (Han et al., 2018, 2024; Zhou et al., 2024), their potential in modeling infectious disease dynamics has not been fully realized. This gap suggests that modified PIM, particularly multistage variants, could serve as effective tools for solving nonlinear epidemiological systems. Applying a MPIM to the monkeypox transmission model offers an opportunity to enhance predictive reliability and extend the use of iterative methods within applied epidemiology, contributing to both methodological

innovation and practical disease modeling.

2.8 Adomian Decomposition Method and Multistage ADM

This section reviews the application of the Adomian Decomposition Method (ADM) and its multistage extension in solving nonlinear differential equations. The discussion focuses on previous applications, reported strengths and limitations, and the role of ADM and MADM as comparative semi-analytical methods in the present study.

2.8.1 Applications of ADM in Nonlinear Systems

The ADM is a semi-analytical technique that has been widely used to obtain approximate solutions for nonlinear differential equations. The method represents the solution as a series of components and treats nonlinear terms through Adomian polynomials. This allows nonlinear problems to be handled without directly linearising the governing equations (Adomian, 1994; Li & Pang, 2020).

Previous studies have shown that ADM can be applied to a wide range of nonlinear problems, including ordinary differential equations, partial differential equations, fractional differential equations, and systems of coupled nonlinear equations. For example, Li and Pang (2020) investigated the iterative scheme and convergence behaviour of ADM for nonlinear systems and demonstrated its application to nonlinear algebraic and fractional differential equations. Other studies have also applied ADM and its modified variants to biological, engineering, and dynamical systems due to their ability to produce analytical-type series approximations (Hunaiber et al., 2023; Ziada, 2022).

Despite these advantages, the performance of ADM may be affected when it is applied to strongly nonlinear systems or over large computational intervals. As more decomposition terms are required, the associated Adomian polynomials can become increasingly complicated, which may increase computational effort. In addition, the truncated series solution may gradually lose accuracy as the simulation interval becomes longer. These limitations have motivated the development of modified and multistage ADM approaches to improve the numerical behaviour of the classical

method.

In the present study, ADM is included as a comparative semi-analytical method for the selected system of ODE applications. Its inclusion enables the performance of PIM and MPIM to be assessed against another iterative decomposition-based approach, particularly with respect to approximation accuracy and numerical behaviour over the selected computational interval.

2.8.2 Multistage ADM in Previous Studies

The MADM was developed to address the limited global validity that may arise when the classical ADM series is applied over a long time interval. Instead of constructing a single ADM approximation over the entire domain, the multistage strategy divides the computational interval into several smaller subintervals. The approximate solution obtained at the end of one subinterval is then used as the initial condition for the next subinterval.

This procedure helps to control the accumulation of approximation errors and reduces the need to use a large number of decomposition terms over an extended interval. Ann and Samarji (2009) applied MADM to the autonomous Van der Pol system and compared its results with classical ADM and a Runge-Kutta method. Their findings indicated that the multistage formulation provided closer agreement with the Runge-Kutta solution than the classical ADM when the same number of decomposition terms was used. More recently, Dong and Xie (2025) examined the application of the conventional MADM to fractional ordinary differential equations. Using a fractional FitzHugh-Nagumo neuron model, they found that the traditional MADM produced physically inconsistent results because it neglected the nonlocal memory effect. They then proposed a revised MADM incorporating a numerical memory term, which produced stable, convergent, and more accurate approximations. Jalili (2025) developed a hybrid framework that combines the Double Sumudu Transform with a parameter-optimized multistage ADM for solving nonlinear time-fractional PDEs with mixed boundary conditions. The method was tested on several benchmark problems, including the time-fractional heat equation, fractional Klein-Gordon equation, and nonlinear reaction-diffusion models. The results showed improved convergence, lower root-mean-square errors, and

competitive computational cost compared with classical ADM, transform-based methods, and numerical techniques.

Duan (2023) discussed the application of ADM for simulating bulk power system electromechanical dynamics. Since the semi-analytical ADM solution is effective only over a limited time window, a multistage ADM was introduced to extend its applicability over longer simulation intervals. The proposed multistage approach was considered for both deterministic and stochastic power system models, including uncertainties in distributed energy resources and loads. Agom et al. (2017) applied the MADM to solve nonlinear fourth-order multipoint boundary value problems. The method produced a series-form approximate solution using a finite number of Adomian polynomial terms. Comparison with the continuous closed-form solution showed very small absolute errors, indicating the potential of the method for nonlinear differential equations. Razali et al. (2013) applied the MADM to solve the chaotic Lü system. The standard ADM was reformulated into a hybrid numerical-analytical multistage ADM to improve its performance for the nonlinear system. Numerical comparisons with the classical ADM and RK4 method showed that MADM provided more accurate approximations. Chowdhury et al. (2015) presented the multistage Adomian decomposition method for solving both linear and nonlinear stiff systems of ordinary differential equations. The study used the multistage strategy to overcome the limited applicability of the classical ADM over extended solution intervals. The results demonstrated that MADM can provide an effective semi-analytical approach for stiff ODE systems.

In this thesis, MADM is not introduced as a new proposed method. Instead, it is employed as a comparative extension of ADM for the system of ODE applications. The comparison among PIM, MPIM, ADM, and MADM provides a broader evaluation of whether the multistage strategy improves the numerical performance of iterative semi-analytical methods over the selected simulation intervals.

2.9 Runge-Kutta Methods as Numerical Benchmarks

Runge-Kutta methods are widely used numerical techniques for approximating the solutions of ordinary differential equations. Among these methods, the fourth-order Runge-Kutta method (RK4) is commonly applied because it offers a

practical balance between computational efficiency and numerical accuracy for many non-stiff initial value problems (Atkinson, 2008; Forsythe et al., 1977; Hairer et al., 1993). Consequently, RK4 has frequently been used as a numerical reference in studies involving nonlinear differential equations, including epidemiological and biological models.

However, the suitability of explicit Runge-Kutta methods depends on the characteristics of the problem being considered. In particular, explicit methods may require sufficiently small time steps when applied to systems that exhibit stiff behaviour, whereas implicit methods are generally more suitable for strongly stiff problems due to their wider stability properties (Butcher, 2016; Wanner & Hairer, 1996). For example, Movaheedi and Rahmani (2026) compared RK4 and the Backward Euler method for a monkeypox transmission model and reported that Backward Euler showed better numerical stability for the stiff simulation considered in their study.

2.9.1 Justification of RK4 and RK8 Benchmarks

Stiffness is an important consideration in the numerical solution of epidemic models because explicit numerical methods may require sufficiently small time steps to maintain numerical stability. In general, explicit Runge-Kutta methods are commonly applied to non-stiff initial value problems, whereas implicit methods are generally preferred for stiff systems because of their more favourable stability properties (Butcher, 2016; Hairer et al., 1993; Wanner & Hairer, 1996). In particular, Movaheedi and Rahmani (2026) reported that the Backward Euler method provided better numerical stability than RK4 for the stiff monkeypox simulation considered in their study.

Nevertheless, the numerical behaviour of a compartmental epidemic model depends on the selected parameter values, initial conditions, simulation interval, and numerical step size. Therefore, the result reported by Movaheedi and Rahmani (2026) does not necessarily imply that the monkeypox model considered in the present study exhibits the same degree of stiffness. In this study, RK4 is implemented using a small time step of $\Delta t = 0.01$ over the interval $0 \leq t \leq 120$. Its role is to provide a well-established numerical reference for evaluating the approximations obtained by

PIM, MPIM, ADM, and MADM under the selected numerical configuration.

To provide an additional level of verification for the monkeypox model, the eighth-order Runge-Kutta method (RK8) is also employed as a higher-order numerical benchmark. The monkeypox model is selected for this additional comparison because it consists of eight coupled nonlinear compartments and is simulated over a relatively long time interval. The RK8 solution is therefore used to assess the consistency of the RK4 results and to provide a more rigorous reference for the long-term comparison of the semi-analytical methods. The close agreement between the RK4 and RK8 solutions for the selected parameter setting supports the use of RK4 with $\Delta t = 0.01$ as a practical benchmark in this application.

RK8 is not applied to all examples in this study. For the remaining ODE and PDE applications, RK4 is retained as the numerical benchmark because these examples are primarily used to examine the formulation, convergence behaviour, and comparative accuracy of the proposed methods under their respective selected settings. Moreover, where an exact solution is available, the exact solution remains the primary reference for assessing numerical accuracy. Hence, the use of RK8 is restricted to the monkeypox model, which represents the most computationally demanding epidemiological application considered in this study.

2.10 Summary of Literature Gaps

A summary of the literature gaps identified in this chapter is presented in Table 2.1. This table highlights the current state of research, the missing components in existing studies, and the specific contributions proposed in this thesis.

Table 2.1
Summary of Literature Gaps and Contributions of This Thesis

Topic	**What Literature Has Done**	**Gap**	**Contribution of This Thesis**
Monkeypox ODE Model	Used fractional Adams Bashforth schemes (Peter et al., 2022b), Atangana Baleanu Caputo fractional derivatives (Manivel et al., 2025), and other fractional order numerical techniques for solving nonlinear multi compartment monkeypox transmission models (Sweilam et al., 2025).	No Picard-based method has been applied to the monkeypox transmission model.	First application of PIM and MPIM to simulate the monkeypox model.
Picard Method	Classical Picard method mostly used for first order IVPs with slow convergence for nonlinear systems (Pollock et al., 2019; Yao et al., 2021).	Cannot solve higher order ODEs directly; requires conversion to first order systems.	Proposed PIM and MPIM capable of solving *n*th-order ODEs directly.
Modified Methods	Multistage improvements exist in RK methods, VIM, HAM, and ADM (Prabowo & Mungkasi, 2021).	No multistage Picard Iterative Method reported in the literature.	Development of a MPIM to enhance accuracy.

The literature review demonstrates that although numerous numerical techniques have been successfully applied to nonlinear epidemiological models, there remains a clear methodological gap in the use of Picard-based iterative schemes, particularly for the monkeypox transmission model. The absence of multistage frameworks within the Picard family and the inability of classical Picard iteration to handle higher order differential equations directly further highlight the need for methodological advancements. This thesis addresses these gaps by introducing two key innovations: (i) a MPIM that enhances local accuracy and improves convergence behaviour for nonlinear ODE systems, and (ii) a direct PIM and MPIM capable of solving *n*th order differential equations without converting them into systems of first order equations. These contributions represent novel extensions to the Picard framework and provide a more efficient numerical approach for analysing complex infectious disease models such as monkeypox.

CHAPTER 3
RESEARCH METHODOLOGY

3.1 Introduction

This chapter presents the methodological framework for the development and implementation of Picard-based methods in solving nonlinear differential equations. Four approaches are considered, namely the classical Picard Iterative Method (PIM), the Multistage Picard Iterative Method (MPIM), the Picard Iterative Method for *n*th-order differential equations (PIM*n*), and the Multistage Picard Iterative Method for *n*th-order differential equations (MPIM*n*). The PIM and MPIM are developed for the solution of first-order differential equations, systems of ordinary differential equations, and partial differential equations, whereas PIM*n* and MPIM*n* extend the approaches for the direct solution of *n*th-order ordinary differential equations. This chapter explains the theoretical formulation, algorithmic structure, and computational procedures of these four Picard-based approaches.

The chapter is organized as follows. Section 3.2 introduces the mathematical preliminaries and fundamental equations considered in this research. Section 3.3 discusses the theoretical formulation and basic principles of the PIM, while Section 3.4 presents the formulation and algorithmic implementation of PIM*n*. Section 3.5 describes the formulation and algorithmic implementation of the MPIM, followed by the formulation of MPIM*n* in Section 3.6. Section 3.7 demonstrates the application of PIM*n* and MPIM*n* to ordinary differential equations. Section 3.8 presents the application of PIM and MPIM to systems of ordinary differential equations, while Section 3.9 presents their application to partial differential equations. Finally, Section 3.10 introduces the absolute error formula used to evaluate the numerical results in this study.

3.2 Mathematical Preliminaries

To establish a clear foundation for the proposed methods discussed in this chapter, this section reviews several mathematical preliminaries relevant to the study

of differential equations. The problems addressed in this research are expressed in terms of differential equations, which describe rates of change with respect to an independent variable such as time or space. These equations are broadly categorized into ODEs and PDEs. The forms and notations introduced here serve as the groundwork for the development of the PIM and the MPIM in the subsequent sections.

A general first order ordinary differential equation can be written as

$$\frac{dy}{dt} = f(t,y), \qquad y(t_0) = y_0, \tag{3.1}$$

where $y(t)$ denotes the unknown function, $f(t,y)$ represents a continuous function describing the rate of change, and y_0 is the initial value of y at $t = t_0$. Equation (3.1) expresses how the dependent variable y evolves with respect to the independent variable t. In many real-world applications, direct closed-form solutions of Equation (3.1) are not attainable due to the nonlinearity of $f(t,y)$ (Basor & Morrison, 2024; Branga, 2022). Therefore, approximate or iterative methods are often employed to determine its behavior.

Equation (3.1) represents a first-order initial value problem; however, many real-world phenomena are more appropriately described by higher-order differential equations. Such systems involve derivatives of order greater than one and require multiple initial conditions to ensure a unique solution. In general, an nth-order ordinary differential equation can be expressed in the following form:

$$\frac{d^n y}{dt^n} = f\left(t, y, y', \ldots, y^{(n-1)}\right), \quad y^{(k)}(t_0) = y_k,\ k = 0, \ldots, n-1 \tag{3.2}$$

The formulation can be extended to a system of first-order ODEs when several dependent variables interact, as represented by

$$\frac{d\mathbf{Y}}{dt} = \mathbf{F}(t, \mathbf{Y}), \qquad \mathbf{Y}(t_0) = \mathbf{Y}_0. \tag{3.3}$$

Here, $\mathbf{Y} = [y_1, y_2, \ldots, y_n]^\top$ is a vector of unknown functions, while $\mathbf{F}(t,\mathbf{y}) = [f_1, \ldots, f_n]^\top$ denotes a vector-valued function describing the corresponding rates of

change. Equation (3.3) commonly arise in chemical kinetics, biological population dynamics, and engineering systems, where multiple quantities evolve simultaneously and may be nonlinearly coupled.

For problems that depend on both time and spatial variables, the mathematical description is expressed as a partial differential equation. A general evolution-type PDE considered in this study can be written as

$$\frac{\partial u}{\partial t} = F\big(x, t, u, u_x, u_{xx}\big), \tag{3.4}$$

in this expression, $u(x,t)$ is the dependent variable varying with respect to space x and time t, while u_x and u_{xx} denote the first and second spatial derivatives, respectively. The function F encodes the governing dynamics of the system. Equations (3.4) arise frequently in heat conduction, diffusion, reaction–diffusion processes, and fluid-related phenomena (Higham, 2008; Wong et al., 2023; Yasui, 2022).

Equations (3.1)-(3.4) summarize the fundamental mathematical forms that define the problems investigated in this research. Due to the presence of nonlinear terms, these equations often cannot be solved via direct analytical means. Consequently, semi-analytical approaches notably iterative constructs developed later in this chapter are introduced to obtain accurate and efficient approximations. The detailed formulations and algorithms of the PIM and its multistage enhancement (MPIM) will be presented in Sections 3.3 and 3.5, respectively.

3.3 Picard Iterative Method (PIM)

The PIM, first introduced by the French mathematician Picard (1893), represents one of the earliest semi-analytical frameworks for obtaining approximate solutions to differential equations in the late nineteenth century (circa 1890). The method was subsequently formalized and extended by Lindelöf (1894), resulting in the well-known Picard–Lindelöf (Cauchy–Lipschitz) theorem, which rigorously establishes the existence and uniqueness of solutions for first order ODEs under suitable continuity and Lipschitz conditions. In essence, the PIM reformulates a given differential equation into its corresponding integral form and constructs a sequence of successive approximations that converges toward the exact analytical solution.

Owing to its conceptual simplicity and strong theoretical underpinnings, the method remains foundational in modern semi-analytical and numerical studies of nonlinear ODEs (Butcher, 2016; Hermann & Saravi, 2016).

To motivate the construction, consider the general first order ODE introduced in Equation (3.1). By integrating both sides of Equation (3.1) from t_0 to an arbitrary time t, the problem can be rewritten in an equivalent integral form,

$$y(t) = y_0 + \int_{t_0}^{t} f\big(s, y(s)\big)\, ds. \tag{3.5}$$

Since the exact function $y(s)$ inside the integral is unknown, an iterative construction is employed by replacing the unknown solution with its current approximation.

The PIM is then defined by

$$y_{n+1}(t) = y_0 + \int_{t_0}^{t} f\big(s, y_n(s)\big)\, ds, \tag{3.6}$$

where $y_n(t)$ denotes the nth approximation and $y_{n+1}(t)$ the $(n+1)$th. The iteration is initiated with an admissible initial guess $y_0(t)$, typically taken as the constant function $y_0(t) = y_0$ consistent with the initial condition. Successive approximations are computed by substituting y_n into the right-hand side of (3.6).

In practice, the iterative process is terminated once the difference between two consecutive approximations becomes sufficiently small according to a prescribed tolerance, ε.

$$\big| y_{n+1}(t) - y_n(t) \big| < \varepsilon. \tag{3.7}$$

Theoretical convergence of the sequence $\{y_n\}_{n\geq 0}$ towards the unique solution of (3.1) is guaranteed when the right-hand side f satisfies a Lipschitz condition with respect to y,

$$\big| f(t, y_1) - f(t, y_2) \big| \leq L \big| y_1 - y_2 \big|, \qquad \text{for all } y_1, y_2 \text{ in the domain}, \tag{3.8}$$

where $L > 0$ is a Lipschitz constant. Under such a condition, (3.6) defines a contraction mapping on a suitable function space, and the fixed point coincides with the unique solution of (3.1).

From a computational standpoint, the PIM proceeds through the following steps:

a) **Initialization:** Specify the ODE $y'(t) = f(t,y)$ and the initial value $y(t_0) = y_0$.

b) **Initial approximation:** Choose $y_0(t) \equiv y_0$ (or another admissible simple function consistent with the initial data).

c) **Picard update:** For $n = 0, 1, 2, \ldots$, compute y_{n+1} via (3.6).

d) **Convergence check:** Stop the iteration once the criterion (3.7) holds for all t in the interval of interest (or in a discrete mesh thereof).

Although the PIM offers a conceptually simple and analytically transparent framework, its direct application to strongly nonlinear problems or over extended time intervals may lead to slow convergence and the accumulation of truncation errors. These limitations motivate the enhanced multistage strategy presented later in Section 3.5, in which the solution interval is partitioned and Picard updates are performed locally to improve stability and convergence.

3.3.1 Complete Convergence Analysis

The following convergence result is adapted from the Banach fixed point framework for Picard iteration (Teschl, 2012).

Theorem 3.1 (Banach Fixed Point Theorem for PIM).

Let $f : [t_0, T] \times \mathbb{R} \to \mathbb{R}$ *satisfy:*

a) f *is continuous on* $[t_0, T] \times \mathbb{R}$.

b) f *satisfies the Lipschitz condition in* y*:*

$$|f(t,y_1) - f(t,y_2)| \leq L|y_1 - y_2|, \quad \forall t \in [t_0, T],\ y_1, y_2 \in \mathbb{R}. \tag{3.9}$$

Then, for any T *satisfying* $L(T - t_0) < 1$*, the Picard operator* $\mathscr{P}$ *is a contraction on* $C([t_0, T])$ *equipped with the supremum norm* $\|\cdot\|_\infty$*, and the sequence* $\{y_n\}$ *defined by* (3.6) *converges uniformly to the unique solution of* (3.1).

Proof. Define $X = C([t_0, T])$ with norm $\|y\|_\infty = \max_{t\in[t_0,T]} |y(t)|$. For any $\phi, \psi \in X$:

$$\|\mathscr{P}\phi - \mathscr{P}\psi\|_\infty = \max_{t\in[t_0,T]} \left| \int_{t_0}^{t} [f(s,\phi(s)) - f(s,\psi(s))]\, ds \right| \tag{3.10}$$

$$\leq \max_{t\in[t_0,T]} \int_{t_0}^{t} |f(s,\phi(s)) - f(s,\psi(s))|\, ds \tag{3.11}$$

$$\leq \max_{t\in[t_0,T]} \int_{t_0}^{t} L|\phi(s) - \psi(s)|\, ds \tag{3.12}$$

$$\leq L(T - t_0)\|\phi - \psi\|_\infty \tag{3.13}$$

Since $L(T - t_0) < 1$, $\mathscr{P}$ is a contraction. By Banach Fixed Point Theorem, $\mathscr{P}$ has a unique fixed point $y^* \in X$, and for any initial $y_0 \in X$, $y_n = \mathscr{P}^n y_0 \to y^*$. □

Theorem 3.2 (Error Estimate for PIM).

Under the conditions of Theorem 3.1, the error after n iterations satisfies:

$$\|y_n - y^*\|_\infty \leq \frac{[L(T - t_0)]^n}{1 - L(T - t_0)} \|y_1 - y_0\|_\infty \tag{3.14}$$

Proof. From contraction property:

$$\|y_{n+1} - y_n\|_\infty \leq [L(T - t_0)]^n \|y_1 - y_0\|_\infty \tag{3.15}$$

$$\|y^* - y_n\|_\infty \leq \sum_{k=n}^{\infty} \|y_{k+1} - y_k\|_\infty \tag{3.16}$$

$$\leq \sum_{k=n}^{\infty} [L(T - t_0)]^k \|y_1 - y_0\|_\infty \tag{3.17}$$

$$= \frac{[L(T - t_0)]^n}{1 - L(T - t_0)} \|y_1 - y_0\|_\infty \tag{3.18}$$

□

3.3.2 Complete PIM Algorithm

Based on the iterative formulation in Equation (3.6), the stopping criterion in Equation (3.7) and the complete computational procedure of the classical PIM are summarised in Algorithm 1.

Algorithm 1 Classical Picard Iterative Method (PIM)

Require: $f(t,y)$, y_0, $[t_0,T]$, ε, $N_{\max}$
Ensure: Approximate solution $y(t)$ on $[t_0,T]$
Initialize: $y_0(t) \leftarrow y_0$ (constant function)
$n \leftarrow 0$, converged $\leftarrow$ false
while $n < N_{\max}$ and not converged **do**
    Compute $y_{n+1}(t) = y_0 + \int_{t_0}^{t} f(s, y_n(s))\, ds$
    Compute error: $\Delta_n = \|y_{n+1} - y_n\|_\infty$
    if $\Delta_n < \varepsilon$ **then**
        converged $\leftarrow$ true
    else
        $n \leftarrow n+1$
    end if
end while
if not converged **then**
    Warning: Maximum iterations reached without convergence
end if
return $y_{n+1}(t)$

3.4 Picard Iterative Method for *n*th order (PIM*n*)

In this section, the general formulation of the PIM*n* for an *n*th-order IVPs is presented. Consider the following differential equation of order n:

$$\frac{d^n y}{dt^n} = f\big(t, y(t), y'(t), y''(t), \ldots, y^{(n-1)}(t)\big), \qquad t \in [t_0, T], \tag{3.19}$$

subject to the initial conditions

$$y^{(k)}(t_0) = y_k, \qquad k = 0, 1, 2, \ldots, n-1. \tag{3.20}$$

The integral form of Equation (3.19) can be expressed as

$$y(t) = \sum_{r=0}^{n-1} \frac{y_r}{r!}(t-t_0)^r + \int_{t_0}^{t} \frac{(t-s)^{n-1}}{(n-1)!} f\big(s, y(s), y'(s), \ldots, y^{(n-1)}(s)\big)\, ds. \tag{3.21}$$

According to the Picard iterative scheme, successive approximations $\{y_n(t)\}$ are generated recursively as follows:

$$y_{k+1}(t) = \sum_{r=0}^{n-1} \frac{y_r}{r!}(t-t_0)^r + \int_{t_0}^{t} \frac{(t-s)^{n-1}}{(n-1)!} f\big(s, y_k(s), y_k'(s), \ldots, y_k^{(n-1)}(s)\big)\, ds,$$
$$k = 0, 1, 2, \ldots \tag{3.22}$$

Equation (3.22) defines the general iterative formulation of the PIM*n* for any *n*th-order nonlinear differential equation. The detailed derivation and theoretical justification of the PIM*n* are provided in Subsection 4.2.

3.4.1 Convergence Analysis for PIM*n*

Theorem 3.3 (Convergence of PIM*n*).

Let $f : [t_0, T] \times \mathbb{R}^n \to \mathbb{R}$ *satisfies:*

a) f *is continuous*

b) f *satisfies the generalized Lipschitz condition:*

$$|f(t, \mathbf{z}_1) - f(t, \mathbf{z}_2)| \leq L\|\mathbf{z}_1 - \mathbf{z}_2\|, \quad \mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^n \tag{3.23}$$

where $\|\mathbf{z}\| = \sum_{i=1}^{n} |z_i|$

Then, for T *satisfying* $\frac{L(T-t_0)^n}{n!} < 1$*, the sequence* $\{y_m\}$ *converges uniformly to the unique solution of* (3.2)*.*

Proof. Let $X = C^n([t_0, T])$ with norm $\|y\|_{C^n} = \sum_{k=0}^{n} \|y^{(k)}\|_\infty$. For $\phi, \psi \in X$:

$$|(\mathscr{P}_n\phi)(t) - (\mathscr{P}_n\psi)(t)| \leq \frac{1}{(n-1)!} \int_{t_0}^{t} (t-s)^{n-1} |f(s, \phi, \ldots, \phi^{(n-1)})$$
$$- f(s, \psi, \ldots, \psi^{(n-1)})|\, ds \tag{3.24}$$
$$\leq \frac{L}{(n-1)!} \int_{t_0}^{t} (t-s)^{n-1} \sum_{k=0}^{n-1} |\phi^{(k)}(s) - \psi^{(k)}(s)|\, ds \tag{3.25}$$
$$\leq \frac{L(T-t_0)^n}{n!} \|\phi - \psi\|_{C^n} \tag{3.26}$$

Thus, $\mathscr{P}_n$ is a contraction when $\frac{L(T-t_0)^n}{n!} < 1$. □

3.5 Multistage Picard Iterative Method (MPIM)

The MPIM is developed in this study as a modification of the classical Picard Iterative Method (PIM) for solving nonlinear initial value problems. Unlike the classical PIM, which is applied over the entire solution interval, the MPIM divides the interval $[t_0, T]$ into several smaller subintervals. The Picard iteration is then performed locally within each subinterval. The final approximate solution obtained at the end of a stage is used as the initial value for the subsequent stage. Through this staged time-marching procedure, a piecewise-continuous approximation is constructed over the whole computational domain.

For a first order initial value problem

$$y'(t) = f(t, y(t)), \qquad y(t_0) = y_0, \tag{3.27}$$

the equivalent integral equation is given by

$$y(t) = y_0 + \int_{t_0}^{t} f(s, y(s))\, ds. \tag{3.28}$$

By substituting the previous approximation $y_n(s)$ into the nonlinear function $f(s, y(s))$ in (3.28), the classical PIM is expressed as

$$y_{n+1}(t) = y_0 + \int_{t_0}^{t} f(s, y_n(s))\, ds. \tag{3.29}$$

The MPIM is then obtained by rewriting the integral formulation locally on each subinterval. For the k^{th} stage, the initial point t_0 is replaced by the left endpoint of the current subinterval, namely t_{k-1}, while the initial value y_0 is replaced by the updated value $y(t_{k-1})$. This produces the local integral equation

$$y(t) = y(t_{k-1}) + \int_{t_{k-1}}^{t} f(s, y(s))\, ds, \qquad t \in [t_{k-1}, t_k]. \tag{3.30}$$

Applying Picard iteration to (3.30) gives the local MPIM iterative formula

$$y_{n+1,k}(t) = y(t_{k-1}) + \int_{t_{k-1}}^{t} f\big(s, y_{n,k}(s)\big)\, ds, \qquad t \in [t_{k-1}, t_k]. \tag{3.31}$$

Hence, let the computational domain be partitioned into M subintervals as

$$[t_0, T] = [t_0, t_1] \cup [t_1, t_2] \cup \cdots \cup [t_{M-1}, t_M], \tag{3.32}$$

where $t_0 < t_1 < \cdots < t_M = T$. Based on the local formulation in (3.31), $y_{n,k}(t)$ denotes the n^{th} approximation within the k^{th} stage, and $y(t_{k-1})$ acts as the updated initial condition for that stage. Once the solution for stage k is obtained, the final value $y(t_k)$ becomes the starting point for the next stage $k+1$, such that

$$y(t_k) = \lim_{n \to N} y_{n,k}(t_k), \tag{3.33}$$

This recursive update continues sequentially until the final time T is reached, yielding the global approximation $y(t)$ over the entire domain.

The multistage formulation can thus be viewed as a hybrid semi-analytical-numerical strategy, where each subinterval acts as a correction zone that reduces local error propagation. By restricting the integration in (3.31) to a smaller region, the approximation accuracy improves without requiring the classical PIM to approximate the solution over the whole interval at once. Consequently, the MPIM preserves the analytical structure of PIM while improving its computational behaviour, especially for nonlinear systems where the standard PIM series may lose accuracy over extended intervals.

Convergence analysis of the MPIM follows similar reasoning to that of the standard PIM. If $f(t, y)$ satisfies the Lipschitz condition described in (3.9), then each local iteration within a stage converges to the unique solution of that subproblem. Since the approximation is updated at the end of every stage through (3.33), the propagation of error is reduced across the entire time domain. Therefore, the proposed MPIM provides a more stable stage-by-stage approximation framework compared with applying the standard PIM directly over a long interval.

In summary, the MPIM provides an efficient mechanism for extending the applicability of the PIM framework to wider intervals and nonlinear systems. The next section will detail the algorithmic steps used to implement the MPIM in computational form.

3.5.1 Global Convergence of MPIM

Although the local Picard iteration converges within each subinterval, the endpoint approximation obtained from one stage is used as the initial value for the subsequent stage. Hence, the global convergence of the MPIM must also account for the inherited error propagated across successive stages.

For the purpose of the following analysis, let Y_k denote the approximate endpoint value obtained at the end of the k^{th} stage, where

$$Y_0 = y_0, \qquad Y_k = y_{N_k,k}(t_k), \qquad k = 1,2,\ldots,M. \tag{3.34}$$

Let $h_k = t_k - t_{k-1}$ denote the length of the k^{th} subinterval, and define the endpoint error by

$$E_k = \|Y_k - y(t_k)\|. \tag{3.35}$$

Also, define the local iteration difference at the k^{th} stage as

$$\delta_k = \left\|y_{N_k,k} - y_{N_k-1,k}\right\|_{\infty,[t_{k-1},t_k]}. \tag{3.36}$$

Theorem 3.4 (Global convergence and inherited error bound of MPIM).

Let $f : [t_0,T] \times \mathbb{R}^d \to \mathbb{R}^d$ be continuous and satisfy the Lipschitz condition

$$\|f(t,u) - f(t,v)\| \le L\|u - v\|, \qquad \forall t \in [t_0,T], \tag{3.37}$$

where $L > 0$. Suppose that the interval $[t_0,T]$ is partitioned into M subintervals such that

$$q_k = Lh_k < 1, \qquad k = 1,2,\ldots,M. \tag{3.38}$$

Then, the endpoint error of the MPIM approximation satisfies

$$E_k \le e^{Lh_k}E_{k-1} + \frac{q_k}{1-q_k}\delta_k, \qquad k = 1,2,\ldots,M. \tag{3.39}$$

Since $E_0 = 0$, the accumulated error after the m^{th} stage is bounded by

$$E_m \le \sum_{j=1}^{m} e^{L(t_m - t_j)}\frac{q_j}{1-q_j}\delta_j, \qquad m = 1,2,\ldots,M. \tag{3.40}$$

Consequently, if the local stopping criterion satisfies

$$\delta_k \leq \varepsilon_{\text{local},k}, \qquad k = 1, 2, \ldots, M, \tag{3.41}$$

then the global MPIM error on $[t_0, T]$ is bounded by

$$\|\widetilde{y} - y\|_{\infty,[t_0,T]} \leq \sum_{j=1}^{M} e^{L(T-t_j)} \frac{q_j}{1-q_j} \varepsilon_{\text{local},j}. \tag{3.42}$$

Therefore, for a fixed partition satisfying $q_k < 1$, the MPIM approximation converges uniformly to the exact solution as the local stopping tolerances tend to zero.

Proof. For the k^{th} stage, consider the local Picard operator

$$\mathscr{P}_k[v](t) = Y_{k-1} + \int_{t_{k-1}}^{t} f(s, v(s))\, ds. \tag{3.43}$$

For any two functions $u, v \in C([t_{k-1}, t_k], \mathbb{R}^d)$,

$$\begin{aligned} \|\mathscr{P}_k[u] - \mathscr{P}_k[v]\|_{\infty,[t_{k-1},t_k]} &\leq L h_k \|u - v\|_{\infty,[t_{k-1},t_k]} \\ &= q_k \|u - v\|_{\infty,[t_{k-1},t_k]}. \end{aligned} \tag{3.44}$$

Since $q_k < 1$, the local Picard operator is contractive. Let $z_k(t)$ denote its fixed point on $[t_{k-1}, t_k]$. Then, by the a posteriori estimate for a contraction mapping,

$$\left\|y_{N_k,k} - z_k\right\|_{\infty,[t_{k-1},t_k]} \leq \frac{q_k}{1-q_k} \delta_k. \tag{3.45}$$

Next, the difference between the local exact solution $z_k(t)$ and the global exact solution $y(t)$ satisfies

$$\|z_k(t) - y(t)\| \leq E_{k-1} + L \int_{t_{k-1}}^{t} \|z_k(s) - y(s)\|\, ds. \tag{3.46}$$

By Grönwall's inequality Gronwall (1919),

$$\|z_k(t) - y(t)\| \leq e^{L(t-t_{k-1})} E_{k-1}. \tag{3.47}$$

At $t = t_k$, the triangle inequality together with (3.45) and (3.47) gives

$$E_k \leq e^{Lh_k} E_{k-1} + \frac{q_k}{1-q_k}\delta_k, \tag{3.48}$$

which proves (3.39). Repeated application of this recurrence yields (3.40), while (3.42) follows from the local tolerance condition. □

Remark 3.1.

Theorem 3.4 shows that the global MPIM error consists of two components. The first component,

$$e^{Lh_k} E_{k-1}, \tag{3.49}$$

represents the inherited error from the preceding stage, whereas the second component,

$$\frac{q_k}{1-q_k}\delta_k, \tag{3.50}$$

represents the remaining local Picard iteration error. Hence, the MPIM does not eliminate error propagation completely; instead, it provides a controlled stage-by-stage error bound when the subinterval lengths and local stopping tolerances are suitably selected.

3.5.2 Algorithmic Steps of MPIM

The implementation of the MPIM can be summarized in the following algorithmic steps. This procedure illustrates the staged computation process, where each subinterval contributes to the construction of the overall global solution.

a) Define the problem.

   Specify the differential equation in the form $y'(t) = f(t,y)$ together with its initial condition $y(t_0) = y_0$. Select the total simulation interval $[t_0, T]$.

b) Partition the domain.

   Divide the interval $[t_0, T]$ into M uniform or nonuniform subintervals $[t_{k-1}, t_k]$, where $t_k = t_0 + kh$ and $h = (T - t_0)/M$ is the step length.

c) Initialize the first stage.

   Set $k = 1$ and use $y(t_0) = y_0$ as the starting value for the first subinterval.

d) Perform local Picard iterations.

Within each subinterval $[t_{k-1}, t_k]$, compute successive approximations using

$$y_{n+1,k}(t) = y(t_{k-1}) + \int_{t_{k-1}}^{t} f\big(s, y_{n,k}(s)\big)\, ds, \tag{3.51}$$

for $n = 0, 1, 2, \ldots, N$, until the convergence condition

$$\big| y_{n+1,k}(t) - y_{n,k}(t) \big| < \varepsilon \tag{3.52}$$

is satisfied.

e) Update stage condition.

Once convergence is achieved, set $y(t_k) = y_{N,k}(t_k)$ as the initial value for the next subinterval.

f) Proceed to the next stage.

Increment $k \leftarrow k + 1$ and repeat Steps 4–5 until $k = M$.

g) Construct the global solution.

Combine all stagewise approximations $\{y^{(1)}, y^{(2)}, \ldots, y^{(M)}\}$ to obtain a continuous global representation of $y(t)$ over $[t_0, T]$.

3.5.3 Complete MPIM Algorithm

To facilitate numerical implementation, Algorithm 2 summarizes the above steps in pseudocode form.

Algorithm 2 Multistage Picard Iterative Method (MPIM)

Require: $f(t,y)$, y_0, $[t_0,T]$, M, $\varepsilon_{\text{local}}$, $\varepsilon_{\text{global}}$, $N_{\max}^{\text{local}}$
Ensure: Global solution $y(t)$ on $[t_0,T]$

Partition $[t_0,T]$ into M subintervals: $t_k = t_0 + k \cdot \frac{T-t_0}{M}$
Initialize: $y_{\text{global}}(t_0) \leftarrow y_0$
for $k = 1$ to M **do**
    Stage k: Interval $[t_{k-1},t_k]$
    $y^0_{\text{stage}}(t) \leftarrow y_{\text{global}}(t_{k-1})$ (constant initial guess)
    $n \leftarrow 0$, $\text{converged}_{\text{local}} \leftarrow$ false
    while $n < N_{\max}^{\text{local}}$ and not $\text{converged}_{\text{local}}$ **do**
        Compute for $t \in [t_{k-1},t_k]$:
        $y_{(n+1),\text{stage}}(t) = y_{\text{global}}(t_{k-1}) + \int_{t_{k-1}}^{t} f(s, y_{(n),\text{stage}}(s))\,ds$
        $\Delta_{\text{local}} = \max_{t\in[t_{k-1},t_k]} |y_{(n+1),\text{stage}}(t) - y_{(n),\text{stage}}(t)|$
        if $\Delta_{\text{local}} < \varepsilon_{\text{local}}$ **then**
            $\text{converged}_{\text{local}} \leftarrow$ true
        else
            $n \leftarrow n+1$
        end if
    end while
    if not $\text{converged}_{\text{local}}$ **then**
        Warning: Stage k did not converge within $N_{\max}^{\text{local}}$ iterations
        Consider decreasing stage length h or increasing $N_{\max}^{\text{local}}$
    end if
    Set $y_{\text{global}}(t_k) \leftarrow y_{(n+1),\text{stage}}(t_k)$
    Store stage solution: $y^{(k)}(t) = y_{(n+1),\text{stage}}(t)$ for $t \in [t_{k-1},t_k]$
end for
Assemble global solution: $y(t) = y^{(k)}(t)$ for $t \in [t_{k-1},t_k]$, $k = 1,\ldots,M$
return $y(t)$

This algorithm highlights the sequential nature of the MPIM, where each stage operates as a self-contained iterative process before propagating its final output to the next interval. Such a structure ensures both computational efficiency and numerical stability, particularly when applied to nonlinear or stiff ODE systems.

In conclusion, the algorithmic design of the MPIM allows the method to maintain analytical accuracy while systematically managing local errors within each subinterval. This staged correction mechanism results in faster convergence compared with the classical PIM, thereby extending the applicability of the iterative approach to a wider range of nonlinear problems.

3.6 Multistage Picard Iterative Method for *n*th-Order (MPIM*n*)

In this section, the multistage formulation of the PIM*n* for an *n*th-order IVPs is presented. Consider the following *n*th-order nonlinear differential equation:

$$\frac{d^n y}{dt^n} = f\big(t, y(t), y'(t), y''(t), \ldots, y^{(n-1)}(t)\big), \quad t \in [t_0, T], \tag{3.53}$$

subject to the initial conditions

$$y^{(k)}(t_0) = y_k, \qquad k = 0, 1, 2, \ldots, n-1. \tag{3.54}$$

To improve the reliability of long-term prediction, the interval $[t_0, T]$ is partitioned into M subintervals:

$$t_0 < t_1 < \cdots < t_M = T, \qquad t_{m+1} = t_m + h, \tag{3.55}$$

where h denotes the step size and $m = 0, 1, \ldots, M-1$. On each subinterval $[t_m, t_{m+1}]$, the Picard iteration is applied locally and the initial information is updated stage-by-stage.

For the mth stage, the integral representation on $[t_m, t_{m+1}]$ is

$$y_m(t) = \sum_{r=0}^{n-1} \frac{y_r^{(m)}}{r!}(t - t_m)^r + \int_{t_m}^{t} \frac{(t-s)^{n-1}}{(n-1)!} f\big(s, y_m(s), y_m'(s), \ldots, y_m^{(n-1)}(s)\big)\, ds, \quad t \in [t_m, t_{m+1}], \tag{3.56}$$

where $y_r^{(m)}$ denotes the stage initial derivatives:

$$y_r^{(m)} \approx y^{(r)}(t_m), \qquad r = 0, 1, \ldots, n-1. \tag{3.57}$$

According to the multistage Picard iterative scheme, successive approximations $\{y_{m,k}(t)\}$ on the mth stage are generated recursively as follows:

$$y_{m,k+1}(t) = \sum_{r=0}^{n-1} \frac{y_r^{(m)}}{r!}(t-t_m)^r + \int_{t_m}^{t} \frac{(t-s)^{n-1}}{(n-1)!} f\big(s, y_{m,k}(s), y'_{m,k}(s), \ldots, y_{m,k}^{(n-1)}(s)\big)\, ds, \tag{3.58}$$

for $k = 0, 1, 2, \ldots$, with an initial guess typically chosen as the stage Taylor polynomial

$$y_{m,0}(t) = \sum_{r=0}^{n-1} \frac{y_r^{(m)}}{r!}(t-t_m)^r, \qquad t \in [t_m, t_{m+1}]. \tag{3.59}$$

After performing K iterations on the mth subinterval, the stage-end values are used to update the initial derivatives for the next stage. Specifically, the propagation (stage update) is defined by

$$y_r^{(m+1)} = y_{m,K}^{(r)}(t_{m+1}), \qquad r = 0, 1, \ldots, n-1, \tag{3.60}$$

so that the next stage starts from t_{m+1} using refreshed initial information. This multistage mechanism limits the accumulation of numerical errors over long time horizons because the Picard iteration is restarted on a shorter interval at each stage, thus improving stability for long-term prediction.

Equation (3.58) defines the general multistage iterative formulation for any nth-order nonlinear differential equation.

3.6.1 Complete MPIM*n* Algorithm

Based on the multistage formulation developed above, the computational procedure for the MPIMn is summarised in Algorithm 3. The algorithm divides the solution domain into smaller stages, constructs a local Picard approximation for the nth-order problem within each stage, and updates the required endpoint derivatives before proceeding to the subsequent stage.

Algorithm 3 MPIM for nth-Order Equations (MPIMn)

Require: $f(t, y, y', \ldots, y^{(n-1)})$, initial conditions $\{y_k\}_{k=0}^{n-1}$, $[t_0, T]$, M, ε, $N_{\max}$
Ensure: Global solution $y(t)$

Partition $[t_0, T]$ into M stages: $t_k = t_0 + k \cdot \frac{T - t_0}{M}$
Initialize derivatives at t_0: $y^{(j)}(t_0) = y_j$, $j = 0, \ldots, n-1$
for $k = 1$ to M **do**
    Stage k**: Interval** $[t_{k-1}, t_k]$
    Initial approximation: $y_0^{(k)}(t) = \sum_{j=0}^{n-1} \frac{y^{(j)}(t_{k-1})}{j!}(t - t_{k-1})^j$
    $m \leftarrow 0$, converged $\leftarrow$ false
    while not converged and $m < N_{\max}$ **do**
        Compute derivatives numerically: $(y_m^{(k)})^{(j)}(t)$ for $j = 1, \ldots, n-1$
        Compute next iteration: $y_{m+1}^{(k)}(t)$ using (3.58)
        Compute error: $\Delta_m = \max_{j=0}^{n-1} \max_{t \in [t_{k-1}, t_k]} |(y_{m+1}^{(k)})^{(j)}(t) - (y_m^{(k)})^{(j)}(t)|$
        if $\Delta_m < \varepsilon$ **then**
            converged $\leftarrow$ true
        else
            $m \leftarrow m + 1$
        end if
    end while
    Update endpoint values: $y^{(j)}(t_k) = (y_m^{(k)})^{(j)}(t_k)$ for $j = 0, \ldots, n-1$
    Store stage solution: $y^{(k)}(t) = y_m^{(k)}(t)$ for $t \in [t_{k-1}, t_k]$
end for
Assemble global solution: $y(t) = y^{(k)}(t)$ for $t \in [t_{k-1}, t_k]$, $k = 1, \ldots, M$
return $y(t)$

3.7 Application of PIM and MPIM to ODEs

The applicability of the PIM and its multistage extension (MPIM) can be demonstrated through their implementation on ODEs. Such equations frequently arise in modelling dynamic processes across physical, chemical, and biological systems, where exact analytical solutions are often unattainable due to nonlinear terms. In these contexts, semi–analytical iterative methods such as PIM and MPIM provide practical and accurate means of approximating the true solution (Butcher, 2016).

Consider the general nonlinear first–order ODEs previously defined in (3.1). For many nonlinear functions $f(t, y)$, a closed–form analytical solution cannot be expressed in finite terms, and iterative techniques are therefore employed to obtain successive approximations to the exact solution.

The Picard iteration for (3.1) is defined by (3.6), which can be restated as

$$y_{n+1}(t) = y_0 + \int_{t_0}^{t} f(s, y_n(s))\, ds, \tag{3.61}$$

where $y_n(t)$ represents the n^{th} approximation and y_0 is the initial condition. The sequence $\{y_n(t)\}$ converges to the unique solution $y(t)$ under the Lipschitz condition stated in (3.9). Each iteration updates the approximation by integrating the nonlinear term $f(t,y)$ with respect to time over the specified interval.

Although the expression in (3.61) yields reliable results for small intervals, the method's accuracy diminishes for extended domains due to truncation error and slow convergence. To overcome this limitation, the MPIM described in Section 3.5 is employed. In this enhanced formulation, the overall interval $[t_0, T]$ is partitioned into M subintervals, and the local Picard iteration is executed within each subinterval according to (3.31).

For a specific subinterval $[t_{k-1}, t_k]$, the local iterative process is given by

$$y_{n+1,(k)}(t) = y(t_{k-1}) + \int_{t_{k-1}}^{t} f\big(s, y_{n,(k)}(s)\big)\, ds, \qquad t \in [t_{k-1}, t_k], \tag{3.62}$$

where $y_{n,(k)}(t)$ denotes the n^{th} local approximation for the k^{th} stage. Once convergence is achieved in each stage, the terminal value $y(t_k)$ becomes the initial value for the next stage. Repeating this procedure sequentially for $k = 1, 2, \ldots, M$ produces a continuous global approximation $y(t)$ across the entire domain.

The primary advantage of this multistage configuration lies in its control of local truncation errors. Because the integration in each subinterval of (3.62) is confined to a smaller range, convergence is accelerated and numerical stability is improved without increasing computational complexity. The MPIM thus preserves the analytical framework of the PIM approach while extending its capability to handle larger time intervals and strongly nonlinear systems.

The general implementation of PIM and MPIM for ODE problems can be summarised as follows:

a) **Model formulation:**

Represent the mathematical model as an ODE of the form (3.1) with initial

condition $y(t_0) = y_0$.

b) **Application of PIM:**

Generate successive approximations $y_{n+1}(t)$ using (3.61) until the convergence criterion in (3.7) is satisfied.

c) **Application of MPIM:**

Divide the solution interval $[t_0, T]$ into M subintervals and apply the local iterative procedure (3.62) within each subinterval, updating the initial condition at the end of each stage.

d) **Result synthesis:** Combine the stagewise solutions $\{y^{(1)}, y^{(2)}, \ldots, y^{(M)}\}$ to form a continuous global approximation $y(t)$ over $[t_0, T]$.

Finally, the results obtained from the PIM and MPIM can be compared with those from standard numerical methods, such as the classical RK4 method, to evaluate the accuracy, convergence rate, and computational efficiency of the proposed multistage framework.

3.8 Application of PIM and MPIM to System of ODEs

Many real–world phenomena are governed by coupled nonlinear differential equations, which can be represented as systems of ODEs. Examples include chemical kinetics, population dynamics, biochemical reaction networks, and compartmental epidemiological models. Analytical solutions to such systems are rarely attainable, particularly when nonlinear coupling terms are present. Consequently, iterative semi–analytical approaches such as the PIM and the MPIM serve as effective tools for obtaining approximate analytical solutions.

A general system of m coupled first–order ODEs can be expressed in vector form in (3.3). Applying the Picard integral formulation to each component of (3.3) yields the vector iteration form

$$\mathbf{Y}_{n+1}(t) = \mathbf{Y}_0 + \int_{t_0}^{t} \mathbf{F}\big(s, \mathbf{Y}_n(s)\big)\, ds, \tag{3.63}$$

as given in (3.63). Here, $\mathbf{Y}_n(t)$ denotes the n^{th} approximation of the vector solution, and $\mathbf{Y}_{n+1}(t)$ is its subsequent refinement. Each iteration requires the evaluation of the nonlinear functions $f_i(t,\mathbf{Y}_n)$ for $i = 1,2,\ldots,m$, followed by component-wise integration over the time domain.

For improved convergence and numerical stability across extended time intervals, the MPIM can be applied to (3.3). By partitioning the total interval $[t_0,T]$ into M smaller subintervals, the local iteration for the k^{th} stage is expressed as

$$\mathbf{Y}_{n+1,(k)}(t) = \mathbf{Y}(t_{k-1}) + \int_{t_{k-1}}^{t} \mathbf{F}\big(s,\mathbf{Y}_{n,(k)}(s)\big)\,ds, \qquad t \in [t_{k-1},t_k], \tag{3.64}$$

where $\mathbf{Y}_m(t_{k-1})$ is the initial vector for the k^{th} subinterval and $\mathbf{Y}_{n,(k)}(t)$ represents the n^{th} local approximation. Once convergence is achieved within each stage, the endpoint $\mathbf{Y}(t_k)$ serves as the initial condition for the subsequent subinterval. Repeating this process for $k = 1,2,\ldots,M$ yields a globally continuous solution $\mathbf{Y}(t)$ on $[t_0,T]$.

To clarify the procedure, consider the two–variable case ($m = 2$) that commonly arises in biochemical kinetics:

$$\begin{aligned} \frac{dy_1}{dt} &= f_1(t,y_1,y_2), \qquad y_1(t_0) = y_{1,0}, \\ \frac{dy_2}{dt} &= f_2(t,y_1,y_2), \qquad y_2(t_0) = y_{2,0}. \end{aligned} \tag{3.65}$$

The Picard iteration corresponding to (3.65) is

$$\begin{aligned} y_{1,n+1}(t) &= y_{1,0} + \int_{t_0}^{t} f_1\big(s,y_{1,n}(s),y_{2,n}(s)\big)\,ds, \\ y_{2,n+1}(t) &= y_{2,0} + \int_{t_0}^{t} f_2\big(s,y_{1,n}(s),y_{2,n}(s)\big)\,ds, \end{aligned} \tag{3.66}$$

and the MPIM formulation for the k^{th} subinterval is written as

$$\begin{aligned} y_{1,n+1,(k)}(t) &= y_1(t_{k-1}) + \int_{t_{k-1}}^{t} f_1\big(s,y_{1,n,(k)}(s),y_{2,n,(k)}(s)\big)\,ds, \\ y_{2,n+1,(k)}(t) &= y_2(t_{k-1}) + \int_{t_{k-1}}^{t} f_2\big(s,y_{1,n,(k)}(s),y_{2,n,(k)}(s)\big)\,ds, \end{aligned} \tag{3.67}$$

for $t \in [t_{k-1}, t_k]$. The iteration continues until the convergence criteria

$$|y_{1,n+1,(k)}(t) - y_{1,n,(k)}(t)| < \varepsilon, \qquad |y_{2,n+1,(k)}(t) - y_{2,n,(k)}(t)| < \varepsilon \tag{3.68}$$

are simultaneously satisfied within each stage.

The principal advantage of the MPIM formulation (3.67) lies in its stagewise correction mechanism, which minimizes local errors while maintaining computational simplicity. This structure allows the method to capture nonlinear coupling between equations efficiently without sacrificing accuracy.

To summarise, the implementation procedure for a general system of ODEs using PIM and MPIM may be outlined as follows:

a) **Model specification:**

Express the problem as a coupled system of first–order ODEs in the form of (3.3).

b) **Initialization:**

Assign the initial vector $\mathbf{Y}(t_0) = \mathbf{Y}_0$ and define the nonlinear function vector $\mathbf{F}(t, \mathbf{Y})$.

c) **Picard iteration:**

Compute successive approximations using (3.63) until convergence according to the tolerance in (3.7).

d) **Multistage iteration:**

Divide the domain into M subintervals and apply the local MPIM updates as in (3.64).

e) **Result assembly:**

Combine all stagewise approximations $\mathbf{Y}^{(k)}(t)$, for $k = 1, 2, \ldots, M$, to construct a continuous global solution over $[t_0, T]$.

The combined application of PIM and MPIM to systems of ODEs provides a powerful semi–analytical framework for solving nonlinear coupled models. This framework preserves the interpretability of analytical methods while achieving

convergence behaviour comparable to high–order numerical integrators such as the RK4 family.

3.9 Application of PIM and MPIM to PDEs

The PIM and its multistage extension (MPIM) can also be effectively applied to the solution of PDEs, which describe a wide range of physical phenomena involving multiple independent variables. Examples include heat conduction, diffusion processes, fluid flow, and reaction–diffusion systems. In such problems, analytical closed–form solutions are often difficult to obtain, especially for nonlinear PDEs or those defined on complex domains. The use of iterative semi–analytical schemes such as PIM and MPIM provides a flexible alternative for constructing approximate analytical solutions with controllable accuracy.

The general idea remains consistent with the formulation of the ordinary differential equation presented earlier in (3.1), but is now extended to functions that depend on both spatial and temporal variables. Let the dependent variable be represented by $u(x,t)$, where x denotes the spatial coordinate and t the temporal variable. A typical form of a nonlinear first–order PDE can be expressed as

$$\frac{\partial u}{\partial t} = F\left(x,t,u,\frac{\partial u}{\partial x},\frac{\partial^2 u}{\partial x^2},\dots\right), \tag{3.69}$$

where F represents a nonlinear differential operator that may involve spatial derivatives of $u(x,t)$. The same iterative principle of the PIM can be applied by transforming the PDE into an equivalent integral formulation with respect to time, while treating the spatial variables as parameters at each iteration.

In this study, no explicit spatial discretization is introduced. The MPIM is applied exclusively in the temporal direction through the time-integral formulation, while spatial derivatives are retained in analytical form throughout the iteration process. Boundary conditions are incorporated symbolically within the initial approximation or imposed directly on the analytical expression of the solution. Accordingly, the proposed framework focuses on enhancing temporal stability and controlling cumulative error propagation, rather than constructing a fully discretized spatial numerical scheme. This iterative construction follows the classical

Picard framework and generates a sequence of approximations $u_n(x,t)$, for which convergence may be ensured under appropriate smoothness and Lipschitz-type conditions on the operator F.

However, similar to the case of ODEs, the single–stage Picard iteration may exhibit slow convergence or instability when applied to nonlinear PDEs over large time intervals. To improve convergence behaviour, the MPIM described in Section 3.5 can be employed. In the MPIM framework, the temporal domain $[t_0, T]$ is partitioned into M smaller subintervals, and the Picard iteration is performed locally within each subinterval for all spatial points. The final solution at the end of one stage, $u(x,t_k)$, then serves as the initial condition for the next stage, $[t_k, t_{k+1}]$, ensuring a time–marching evolution of the solution.

This staged structure improves numerical stability and enables accurate tracking of transient behaviours in diffusion– or wave–type equations. Moreover, the MPIM formulation allows each local iteration to capture nonlinear spatial interactions more effectively, since the correction mechanism confines errors within smaller time windows. The resulting global approximation across the entire domain is therefore smoother and more stable than that produced by a single–stage Picard iteration.

To summarise the implementation of PIM and MPIM for PDEs, the general steps can be outlined as follows:

a) **Model representation:**

Express the physical problem as a PDE in the general form $\frac{\partial u}{\partial t} = F(x,t,u,u_x,u_{xx})$ with the corresponding initial and boundary conditions.

b) **Transformation:**

Convert the PDE into its equivalent integral form with respect to t, keeping spatial derivatives as parameters.

c) **Picard iteration:**

Apply the iterative update based on the integral formulation to generate successive approximations $u_{n+1}(x,t)$ until the convergence condition similar to (3.7) is satisfied.

d) **Multistage application:**

Partition the time domain into M subintervals and execute the iteration locally in each stage, using $u(x,t_k)$ as the initial condition for the subsequent interval.

e) **Solution reconstruction:** Combine all stagewise results to form the global solution $u(x,t)$ defined over the entire space–time domain.

In essence, the application of the MPIM to PDEs provides a semi–analytical framework that combines the interpretability of analytical methods with the numerical efficiency of staged iteration schemes. This makes it particularly suitable for solving time–dependent nonlinear PDEs where high accuracy and computational stability are essential.

3.10 Absolute Error Measure

To evaluate the numerical accuracy of an approximate solution, the absolute error is defined as (Faires & Burden, 2015)

$$E(t_i) = \left| u_{\text{true}}(t_i) - u_{\text{approx}}(t_i) \right|, \tag{3.70}$$

where $u_{\text{true}}(t_i)$ represents the reference or true value of the solution at time t_i, and $u_{\text{approx}}(t_i)$ denotes the corresponding approximate value obtained from the numerical method.

For systems of equations, the absolute error is computed component-wise for each dependent variable. The maximum absolute error over the simulation interval $[t_0, T]$ is also considered to assess overall error behaviour:

$$E_{\max} = \max_{t_i \in [t_0,T]} E(t_i). \tag{3.71}$$

This measure provides a direct indication of the deviation between the approximate and reference solutions and is used to assess numerical accuracy and error propagation over time.

In addition to absolute error evaluation, the effectiveness of the numerical method is also assessed through convergence behaviour and computational efficiency. Convergence is examined by observing the reduction of error across successive iterations and the stability of the solution over extended time intervals.

Computational efficiency is evaluated based on the number of iterations required and the computational time recorded under identical simulation settings. These criteria ensure a systematic comparison of numerical performance.

CHAPTER 4
APPLICATION OF PIM AND MPIM TO ODES

4.1 Introduction

In this chapter, a new extension of the Picard Iterative Method (PIM) is proposed to address one of its fundamental limitations in solving higher-order differential equations. Traditionally, the PIM is restricted to first-order ODEs, requiring higher-order problems to be reduced into equivalent systems of first-order equations. This indirect transformation often complicates the computation, increases the number of dependent variables, and reduces the overall efficiency of the method. To overcome this issue, a Picard Iterative Method for nth-order (PIM*n*) formulation is developed and introduced in this chapter as a direct approach for solving higher-order initial value problems (IVPs) without reducing them to first-order systems. The chapter begins with a detailed derivation of the PIM*n*, including the mathematical formulation, iterative structure, and the construction of successive approximations for arbitrary-order nonlinear IVPs. This section establishes the theoretical foundation of the proposed method and demonstrates how the nth-order form is generated systematically from the integral representation of higher-order differential equations. Several illustrative examples are presented to assess the accuracy, convergence behaviour, and computational simplicity of the new formulation.

Following this, the chapter evaluates the performance of the classical PIM and its multistage variant, the Multistage Picard Iterative Method (MPIM), by applying them to a first-order nonlinear Riccati equation. This section compares the solution behaviours of PIM and MPIM against the exact solution and the fourth-order Runge-Kutta (RK4) method, where the RK4 values are computed using Maple's built-in numerical solver. The comparison highlights the improvement in accuracy and numerical stability gained through the multistage modification. Next, the proposed PIM*n* and Multistage Picard Iterative Method for *n*th order (MPIM*n*) are applied to a nonlinear second-order Duffing equation, demonstrating their capability in solving higher-order nonlinear oscillatory systems directly. The results are

benchmarked against the exact solution and RK4 to evaluate the reliability, accuracy, and convergence characteristics of the methods.

In addition, this chapter further extends the application of PIM*n* and MPIM*n* to the Pohlhausen flow equation, which represents a higher-order nonlinear differential equation commonly associated with boundary layer flow problems. The inclusion of this equation provides a broader assessment of the proposed formulation beyond second-order oscillatory models and demonstrates its applicability to more complex higher-order nonlinear problems. The comparison of numerical results and absolute errors is carried out to examine the performance of PIM*n*, MPIM*n*, and RK4 in terms of accuracy and stability. Through these applications, the chapter establishes the effectiveness of the PIM*n* and MPIM*n* formulations as unified and efficient tools for handling both first-order and higher-order nonlinear initial value problems.

4.2 *n*th order Initial Value Problems

In this section, consider a class of *n*th order IVPs of the form, Yahaya et al. (2007),

$$y^{(n)}(t) = f(t, y(t), y'(t), \ldots, y^{(n-1)}(t)), \quad 0 \leq t \leq T, \tag{4.1}$$

subject to the initial conditions

$$y(t_0) = y_0, \quad y'(t_0) = y'_0, \ldots, y^{(n-1)}(t_0) = y_0^{(n-1)}, \tag{4.2}$$

where f represents a continuous, real linear or nonlinear function and $y_0, y'_0, \ldots, y_0^{(n-1)}$ are prescribed. It is possible to integrate a special *n*th order differential equation of the form (4.1) by reducing it to a first order system and applying one of the established methods available for such system. However, it seems more natural to provide direct numerical methods for solving (4.1)-(4.2).

The aim of this section is to extend for the first time the applicability of the PIM*n* to the direct solution of *n*th order IVPs. Comparison between PIM*n* solutions and exact solutions shall be made.

4.2.1 Solution approach by PIM*n*

In this section, the PIM*n* to handle equations (4.1)-(4.2) shall be demonstrated. To do so, first write equation (4.1) in the operator form

$$L_n y(t) := \frac{d^n y}{dt^n}(t) = f\big(t, y(t), y'(t), \ldots, y^{(n-1)}(t)\big), \qquad t \in [t_0, T], \tag{4.3}$$

subject to

$$y^{(k)}(t_0) = y_0^{(k)}, \qquad k = 0, 1, \ldots, n-1. \tag{4.4}$$

Inverse Operator L_n^{-1}.

For any integrable g,

$$\big(L_n^{-1} g\big)(t) := \int_{t_0}^{t} \frac{(t-x)^{n-1}}{(n-1)!}\, g(x)\, dx. \tag{4.5}$$

Example $n = 2$.

With $y''(t) = g(t)$,

$$y'(t) = y'(t_0) + \int_{t_0}^{t} g(x)\, dx, \qquad y(t) = y(t_0) + y'(t_0)(t-t_0) + \int_{t_0}^{t} (t-x)\, g(x)\, dx. \tag{4.6}$$

Thus $\big(L_2^{-1} g\big)(t) = \int_{t_0}^{t} (t-x)\, g(x)\, dx.$

Example $n = 3$.

With $y^{(3)}(t) = g(t)$, integrate thrice to get

$$y(t) = y(t_0) + y'(t_0)(t-t_0) + \frac{y''(t_0)}{2!}(t-t_0)^2 + \int_{t_0}^{t}\int_{t_0}^{r}\int_{t_0}^{s} g(x)\, dx\, ds\, dr \tag{4.7}$$

$$= y(t_0) + y'(t_0)(t-t_0) + \frac{y''(t_0)}{2!}(t-t_0)^2 + \int_{t_0}^{t} \frac{(t-x)^2}{2!}\, g(x)\, dx. \tag{4.8}$$

Thus $\big(L_3^{-1} g\big)(t) = \int_{t_0}^{t} \frac{(t-x)^2}{2!}\, g(x)\, dx.$

Kernel Pattern for General n.

Based on the cases $n = 2$ and $n = 3$, the underlying pattern can be clearly identified.

$$\int_{t_0}^{t} (t-x)\, g(x)\, dx, \qquad \int_{t_0}^{t} \frac{(t-x)^2}{2!}\, g(x)\, dx \;\Rightarrow\; \int_{t_0}^{t} \frac{(t-x)^{n-1}}{(n-1)!}\, g(x)\, dx. \tag{4.9}$$

Induction proof sketch: Assume

$$(L_n^{-1} g)(t) = \int_{t_0}^{t} \frac{(t-x)^{n-1}}{(n-1)!}\, g(x)\, dx. \tag{4.10}$$

Then

$$\begin{aligned}
(L_{n+1}^{-1} g)(t) &= \int_{t_0}^{t} (L_n^{-1} g)(r)\, dr && (4.11)\\
&= \int_{t_0}^{t} \left[\int_{t_0}^{r} \frac{(r-x)^{n-1}}{(n-1)!}\, g(x)\, dx \right] dr && (4.12)\\
&= \int_{t_0}^{t} \left[\frac{1}{(n-1)!} \int_{r=x}^{t} (r-x)^{n-1} dr \right] g(x)\, dx && (4.13)\\
&= \int_{t_0}^{t} \frac{(t-x)^{n}}{n!}\, g(x)\, dx, && (4.14)
\end{aligned}$$

which is the claimed kernel with $n \mapsto n+1$. Hence for all $n \in \mathbb{N}$,

$$\left(L_n^{-1} g\right)(t) = \int_{t_0}^{t} \frac{(t-x)^{n-1}}{(n-1)!}\, g(x)\, dx \tag{4.15}$$

Origin of the Initial-Data Polynomial. Repeated integration introduces constants of integration that match the initial conditions (4.4). After n integrations one obtains the polynomial

$$\sum_{k=0}^{n-1} \frac{y_0^{(k)}}{k!} (t-t_0)^k \tag{4.16}$$

which is the Taylor expansion of y about t_0 up to order $n-1$.

Volterra Form and Picard Iteration. Combining (4.15) and (4.16) yields

$$y(t) = \sum_{k=0}^{n-1} \frac{y_0^{(k)}}{k!} (t-t_0)^k + \int_{t_0}^{t} \frac{(t-x)^{n-1}}{(n-1)!}\, f\Big(x, y(x), y'(x), \ldots, y^{(n-1)}(x)\Big)\, dx. \tag{4.17}$$

With the seed $y_{[0]}(t) = \sum_{k=0}^{n-1} \frac{y_0^{(k)}}{k!}(t-t_0)^k$, the Picard iterates are

$$y_{[m+1]}(t) = \sum_{k=0}^{n-1} \frac{y_0^{(k)}}{k!}(t-t_0)^k + \int_{t_0}^{t} \frac{(t-x)^{n-1}}{(n-1)!} f\Big(x, y_{[m]}(x), {y_{[m]}}'(x), \ldots, {y_{[m]}}^{(n-1)}(x)\Big)\, dx. \tag{4.18}$$

4.2.2 Numerical experiments

In this section, the efficiency and accuracy of PIMn to (4.1)-(4.2) shall be demonstrated through 2 examples. The PIMn algorithm is coded in the computer algebra package Maple.

Example 1.

First consider the non-linear first order IVP by Yahaya et al. (2007),

$$x' = x(b+ax), \quad b > 0, \quad a < 0, \quad x(0) > 0, \tag{4.19}$$

where a and b are constant. This has an exact solution

$$x(t) = \begin{cases} \dfrac{be^{bt}}{\dfrac{b+ax(0)}{x(0)} - ae^{bt}}, & \text{for } b \neq 0, \\ \dfrac{x(0)}{1-ax(0)t}, & \text{for } b = 0. \end{cases} \tag{4.20}$$

According to the PIMn framework (3.4), the iterative scheme for equation (4.19) can be expressed as follows. It is worth noting that while the PIMn framework is generally constructed for higher-order problems, applying it to a first-order problem ($n = 1$) directly reduces the formulation to the classical PIM. This demonstration serves to verify the consistency of the framework with established classical methods

$$x_{(n+1)}(t) = x(0) + \int_0^t x_n(t)(b+ax_n(t))\, ds, \tag{4.21}$$

By substituting the initial conditions $x(0) = 0.1$ and the parameter values $a = -3$ and $b = 1$ into equations (4.21), the following expressions are obtained.

For $n = 0$:

$$x_1(t) = 0.1 + 0.07t. \tag{4.22}$$

The iterative process was continued up to $n = 5$, at which point convergence was achieved when the tolerance criterion $\varepsilon < 10^{-10}$ was satisfied. Due to the extensive nature of the polynomial expansions, only the first iterations ($n = 0$is presented here. Further details and other iterations equation's can be found in Appendix A

Table 4.1 presents the numerical comparison between the exact solution and the results obtained from six iterations of the PIMn for the given parameters $a = -3$, $b = 1$, and $x(0) = 0.1$. The results indicate an excellent agreement between the exact and approximate solutions, with absolute errors ranging from 10^{-10} to 10^{-9} across all time steps. This demonstrates that the PIMn provides highly accurate results even with a small number of iterations. The minimal deviation observed confirms the method's reliability and stability in solving the nonlinear problem, making it an efficient approach for approximating solutions within the defined interval.

Table 4.1
Numerical Comparison Between the Exact Solution and 6-iterations PIMn for $a = -3$, $b = 1$, and $x(0) = 0.1$.

Time	Exact Solution	6 Iteration of PIMn	Absolute Error
0.0	0.1000000000	0.1000000000	0.0000000000
0.1	0.1071367895	0.1071367895	0.0000000000
0.2	0.1145329055	0.1145329055	0.0000000000
0.3	0.1221638511	0.1221638510	0.0000000001
0.4	0.1300011386	0.1300011385	0.0000000001
0.5	0.1380126120	0.1380126119	0.0000000001
0.6	0.1461628997	0.1461628996	0.0000000001
0.7	0.1544139905	0.1544139903	0.0000000002
0.8	0.1627259130	0.1627259127	0.0000000003
0.9	0.1710574954	0.1710574952	0.0000000002
1.0	0.1793671754	0.1793671752	0.0000000002

Example 2.

Now consider the non-linear second-order IVP by Yahaya et al. (2007)

$$y''(t) = -y'^2(t), \quad 0 \leq t \leq 5, \tag{4.23}$$

$$y(0) = 1, \quad y'(0) = 2, \tag{4.24}$$

$$y_{\text{exact}} = 1 + \ln(1+2t). \tag{4.25}$$

According to the PIM framework, the iterative scheme for the equation (4.23) can be expressed as follows:

$$y_{n+1}(t) = y(0) + y'(0)t + \int_0^t (t-x)(-y_n(x))^2 dx, \tag{4.26}$$

By substituting the initial conditions (4.24) and the parameter values $a = 0.375$ and $b = 1.0$ into equation (4.26), the following expressions are obtained.

For $n = 0$:

$$y_1(t) = 1 + 2t, \tag{4.27}$$

The iterative process was continued up to $n = 5$, at which point convergence was achieved when the tolerance criterion $\varepsilon < 10^{-5}$ was satisfied. Continuing this iterative process, higher order approximations are generated. For brevity, only the six iteration ($n = 5$) is presented in detail, while the complete series are provided in the Appendix A.

Table 4.2 presents a numerical comparison between the exact solution and the results obtained from 6-iterations of the PIM*n*. The findings show that the PIM*n* provides highly accurate results for smaller time intervals, as both solutions coincide at (t = 0) and remain very close for $t \leq 0.3$, with absolute errors less than 10^{-5}. As time increases, the accuracy slightly decreases due to error accumulation from successive iterations, which is a common characteristic of the standard Picard method when applied to nonlinear problems. The absolute error increases gradually from 2.29×10^{-3} at $t = 0.6$ to 2.79×10^{-2} at $t = 1.0$. This indicates that while six iterations are sufficient to achieve reliable accuracy within shorter intervals.

Table 4.2
Numerical Comparison Between the Exact Solution and 6-iteration PIMn Solutions for $a = 0.375$, $b = 1$ and $y(0) = 1$, $y'(0) = 2$.

Time	Exact Solution	6 iteration of PIMn	Absolute Error
0.0	1.0000000000	1.0000000000	0
0.1	1.1823215570	1.1823215070	0.00000005
0.2	1.3364722370	1.3364681043	0.0000041327
0.3	1.4700036290	1.4699557659	0.000047863
0.4	1.5877866650	1.5875349956	0.0002516694
0.5	1.6931471810	1.6922812448	0.0008659362
0.6	1.7884573600	1.7861682809	0.0022890791
0.7	1.8754687370	1.8704124029	0.0050563341
0.8	1.9555114450	1.9457093295	0.0098021155
0.9	2.0296194170	2.0124165516	0.0172028654
1.0	2.0986122890	2.0706972995	0.0279149895

4.3 Riccati Equation

The Riccati equation is one of the most fundamental nonlinear first order differential equations that frequently arises in various branches of applied mathematics, control theory, and fluid mechanics. It takes the general form

$$\frac{dy}{dt} = a(t) + b(t)y + c(t)y^2, \tag{4.28}$$

where $a(t)$, $b(t)$, and $c(t)$ are continuous functions of t. The nonlinear term $c(t)y^2$ introduces mathematical complexity that often prevents the derivation of a closed-form analytical solution. Consequently, iterative and semi–analytical approaches such as the PIM and its improved variant, the MPIM, are employed to obtain accurate approximate solutions.

In this study, the Riccati equation is considered in the simplified autonomous form (Odibat, 2020)

$$\frac{du(t)}{dt} = 1 - u^2(t), \quad t > 0, \tag{4.29}$$

with the initial condition

$$u(0) = u_0, \tag{4.30}$$

where $u_0 \in \mathbb{R}$. the corresponding exact solution is:

$$u(t) = \frac{\tanh(t) + u_0}{u_0 \tanh(t) + 1}. \tag{4.31}$$

To investigate the performance, convergence, and stability of the iterative schemes under different initial states, two distinct initial conditions are considered in the subsequent analysis.

4.3.1 PIM and MPIM solutions

Case 1: $u_0 = -0.5$

PIM:

According to the PIM framework in (3.21), the iterative scheme for the equation (4.29) can be expressed as follows:

$$u_{(n+1)}(t) = -0.5 + \int_0^t (1 - u_n(s)^2) ds. \tag{4.32}$$

By substituting the initial conditions defined in (4.29) into equations (4.32), the following expressions are obtained.

For $n = 0$**:**

$$u_1(t) = -0.5 + 0.75t. \tag{4.33}$$

The iterative process converges at $n = 3$, where the approximate solution $u_4(t)$ is obtained. At this stage, the tolerance condition $\varepsilon < 10^{-5}$ is achieved, indicating that further iterations would yield negligible changes in accuracy. Further details and other iterations equation's can be found in Appendix B.

MPIM:

According to the MPIM framework in (3.31), the iterative scheme for the riccati equation (4.29) can be expressed as follows:

$$u_{(n+1),j}(t) = u_{m,j-1}(t_{j-1}) + \int_{t_{j-1}}^t (1 - u_{n,j}(s)^2) ds. \tag{4.34}$$

By substituting the initial conditions defined in (4.29) into equations (4.34), the following expressions are obtained.

For $n = 3, j = 1$

$$\begin{aligned} u_{4,1} = &-0.5 + 0.75t + 0.75t^2 + 0.1875t^3 - 0.9375t^4 - 2.53125t^5 + 2.53125t^6 \\ &+ 2.91797t^7 - 1.37109t^8 - 3.74414t^9 + 1.107422t^{10} + 2.88721t^{11} \\ &- 1.54248t^{12} - 0.53394t^{13} + 0.53394t^{14} - 0.10011t^{15}. \end{aligned} \tag{4.35}$$

The iterative process was continued up to $n = 3,\ j = 600$. Given the extensive nature of the iterative expressions, only the representative iterations ($n = 3, j = 1$) and ($n = 3, j = 600$) are presented in this section, while the further details and other iterations equation's can be found in the Appendix B.

Case 2: $u_0 = 1.5$

PIM:

According to the PIM framework in (3.21), the iterative scheme for the equation (4.29) can be expressed as follows:

$$u_{(n+1)}(t) = 1.5 + \int_0^t (1 - u_n(s)^2)ds. \tag{4.36}$$

By substituting the initial conditions defined in (4.29) into equations (4.36), the following expressions are obtained.

For $n = 0$**:**

$$u_1(t) = 1.5 - 1.25t. \tag{4.37}$$

The iterative process converges at $n = 4$, where the approximate solution $u_5(t)$ is obtained. At this stage, the tolerance condition $\varepsilon < 10^{-5}$ is achieved, indicating that further iterations would yield negligible changes in accuracy. Further details and other iterations equation's can be found Appendix F

MPIM:

According to the MPIM framework in (3.31), the iterative scheme for the riccati equation (4.29) can be expressed as follows:

$$u_{(n+1),j}(t) = u_{m,j-1}(t_{j-1}) + \int_{t_{j-1}}^t (1 - u_{n,j}(s)^2)ds. \tag{4.38}$$

By substituting the initial conditions defined in (4.29) into equations (4.38), the following expressions are obtained.

For $n = 4, j = 1$:

$$\begin{aligned} u_{5,1}(t) = {} & 1.5000 - 1.2500t + 3.7500t^2 - 12.812t^3 + 47.812t^4 - 189.53t^5 + 480.47t^6 \\ & - 1298.9t^7 + 3399.3t^8 - 8406.4t^9 + 19395t^{10} - 42801t^{11} + 89605t^{12} \\ & - 1.7559 \times 10^5 t^{13} + 3.2369 \times 10^5 t^{14} - 5.5354 \times 10^5 t^{15} + 8.7871 \times 10^5 t^{16} \\ & - 1.2933 \times 10^6 t^{17} + 1.7505 \times 10^6 t^{18} - 2.1689 \times 10^6 t^{19} + 2.4366 \times 10^6 t^{20} \\ & - 2.4552 \times 10^6 t^{21} + 2.1968 \times 10^6 t^{22} - 1.7221 \times 10^6 t^{23} + 1.1638 \times 10^6 t^{24} \\ & - 6.6503 \times 10^5 t^{25} + 3.1213 \times 10^5 t^{26} - 1.1523 \times 10^5 t^{27} + 31596t^{28} \\ & - 5957.2t^{29} + 682.12t^{30} - 35.527t^{31}. \end{aligned} \tag{4.39}$$

The iterative process was continued up to $n = 4$, $j = 500$. Given the extensive nature of the iterative expressions, only the representative iterations ($n = 4, j = 1$), ($n = 4, j = 2$) and ($n = 4, j = 500$) are presented in this section, while the complete series are provided in the Appendix B.

4.3.2 Results and discussion

Case 1: $u_0 = -0.5$

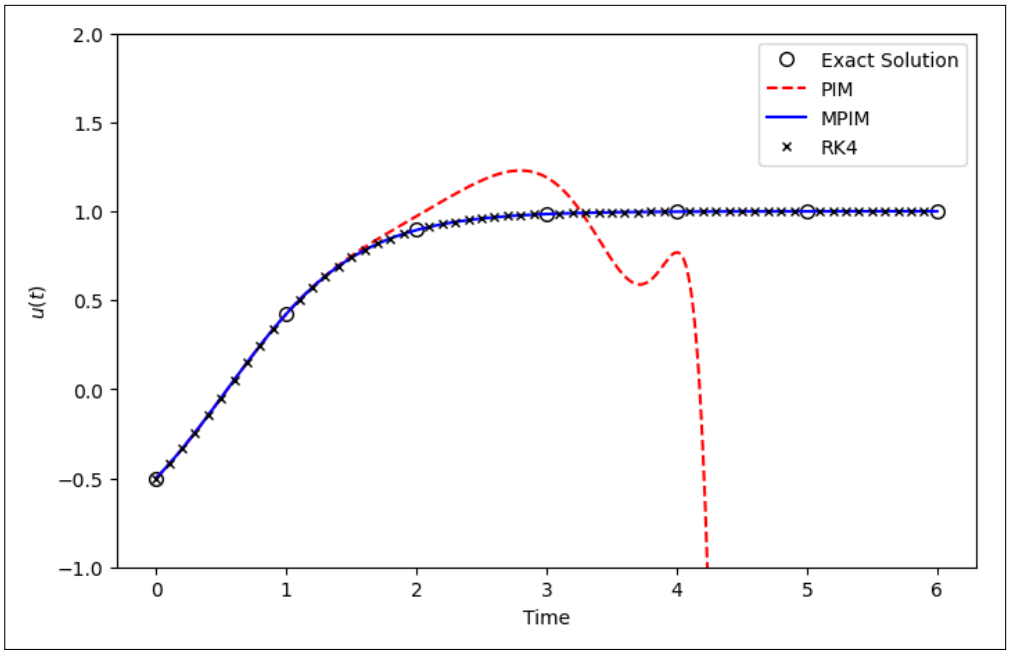

Figure 4.1 Solution of Riccati Equation using 4-iteration PIM, 4-iteration MPIM ($\Delta t = 0.01$), RK4 ($\Delta t = 0.01$) and the Exact Solution.

Figure 4.1 illustrates the numerical solutions of the Riccati equation obtained using the 4-iteration PIM, the 4-iteration MPIM with ($\Delta t = 0.01$), and the RK4 method with ($\Delta t = 0.01$), together with the exact solution for the initial condition $u_0 = -0.5$. At the initial stage, all three numerical methods show good agreement with the exact solution. However, as time approaches $t \approx 1.7$, the PIM solution begins to deviate from the exact solution, whereas both MPIM and RK4 remain closely aligned. Within the time interval $2 < t < 3$, the PIM exhibits a significant deviation, indicating a loss of numerical stability. For $t > 3$, the PIM solution decreases sharply and diverges further from the exact solution, demonstrating poor long-term reliability. In contrast, the MPIM and RK4 solutions closely follow the exact solution throughout the entire time interval $0 \leq t \leq 6$, highlighting the superior stability and accuracy of the MPIM for long-term prediction.

Table 4.3
Comparison of Absolute Errors Between Exact Solutions,4-iteration PIM, 4-iteration MPIM on the Time Step ($\Delta t = 0.01$), RK4 on the Time Step ($\Delta t = 0.01$).

Time	$\lvert\text{Exact} - \text{PIM}\rvert$	$\lvert\text{Exact} - \text{MPIM}\rvert$	$\lvert\text{Exact} - \text{RK4}\rvert$
1	3.5220×10^{-5}	3.6202×10^{-4}	1.8800×10^{-7}
2	7.6511×10^{-2}	1.3144×10^{-3}	1.2000×10^{-8}
3	2.0554×10^{-1}	4.7725×10^{-4}	2.6600×10^{-7}
4	2.2861×10^{-1}	1.0650×10^{-4}	2.4800×10^{-7}
5	4.9852×10^{2}	2.0151×10^{-5}	3.6200×10^{-7}
6	3.1935×10^{4}	3.5194×10^{-6}	1.3600×10^{-7}

Table 4.3 compares the absolute errors of the numerical solutions obtained using the 4-iteration PIM, the 4-iteration MPIM, and the RK4 method with respect to the exact solution for the Riccati equation with the initial condition $u_0 = -0.5$. At the early time levels, the absolute errors of the three methods are relatively small. However, the error behaviour becomes more distinguishable as time increases. For the classical PIM, the absolute error increases from 3.5220×10^{-5} at $t = 1$ to 7.6511×10^{-2} at $t = 2$, before reaching 4.9852×10^{2} at $t = 5$ and 3.1935×10^{4} at $t = 6$. This shows that the PIM experiences substantial error accumulation over the considered interval. In comparison, the MPIM produces much smaller absolute errors than the classical PIM. The MPIM error remains within the range of 10^{-3} to 10^{-6}, decreasing from 3.6202×10^{-4} at $t = 1$ to 3.5194×10^{-6} at $t = 6$. This indicates that the modified approach is more effective in controlling the growth of numerical error. Meanwhile,

the RK4 method gives the lowest absolute errors overall, with error values mostly in the order of 10^{-7} to 10^{-8}. Therefore, although the MPIM does not reach the same accuracy level as RK4, it demonstrates a substantial improvement over the classical PIM in terms of error reduction and numerical stability for this case.

Case 2: $u_0 = 1.5$

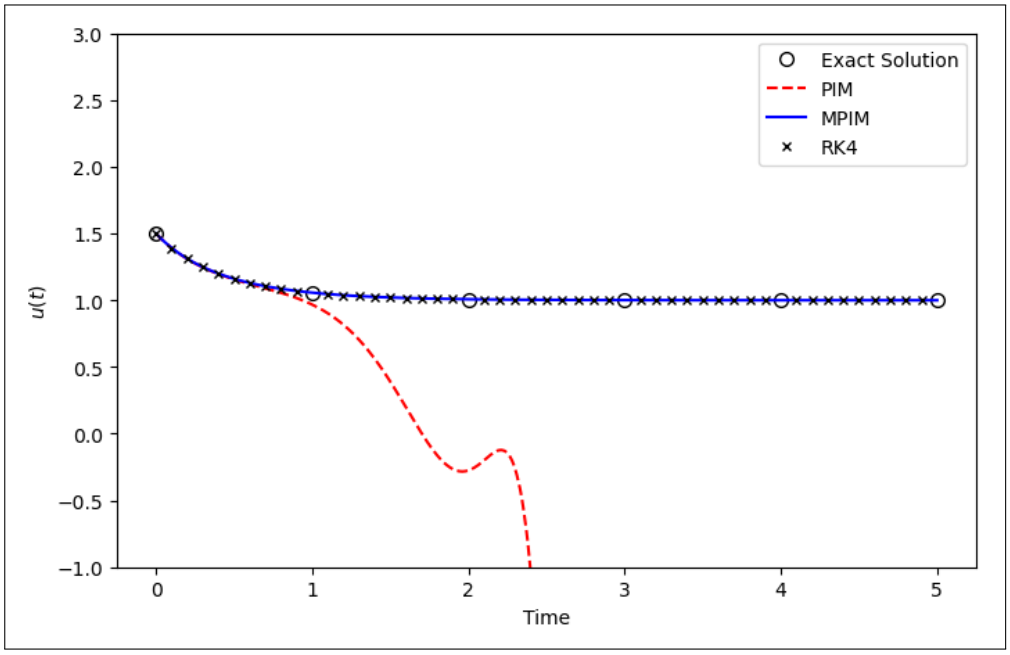

Figure 4.2 Solution of Riccati Equation using 5-iteration PIM, 5-iteration MPIM($\Delta t = 0.01$) RK4 ($\Delta t = 0.01$) and the Exact Solution.

Figure 4.2 presents the numerical solutions of the Riccati equation obtained using the 5-iteration PIM, the 4-iteration MPIM with ($\Delta t = 0.01$), and the RK4 method with ($\Delta t = 0.01$), together with the exact solution, for the initial condition $u_0 = 1.5$. At the initial stage, all three numerical methods are in good agreement with the exact solution. However, as time increases beyond $t \approx 1$, the PIM solution begins to deviate noticeably from the exact solution. Within the interval $1 < t < 2$, the deviation of the PIM becomes more pronounced, indicating a loss of numerical stability. For $t \gtrsim 2$, the PIM solution decreases rapidly and diverges significantly from the exact solution, eventually producing nonphysical behavior. In contrast, both the MPIM and RK4 solutions remain stable and closely follow the exact solution throughout the entire time interval $0 \leq t \leq 5$, demonstrating the robustness and reliability of the MPIM for handling different initial conditions in long-term simulations.

Table 4.4
Comparison of Absolute Errors Between Exact solutions and 5-iteration PIM, 5-iteration MPIM on the Time Step ($\Delta t = 0.01$), RK4 on the Time Step ($\Delta t = 0.01$).

Time	$\lvert\text{Exact} - \text{PIM}\rvert$	$\lvert\text{Exact} - \text{MPIM}\rvert$	$\lvert\text{Exact} - \text{RK4}\rvert$
1	8.8256×10^{-2}	1.3640×10^{-3}	1.2800×10^{-7}
2	1.2825	3.3026×10^{-4}	1.9100×10^{-7}
3	7.4411×10^{1}	6.5143×10^{-5}	7.0000×10^{-9}
4	1.5790×10^{3}	1.1661×10^{-5}	1.9400×10^{-7}
5	1.2565×10^{7}	1.9690×10^{-6}	1.6000×10^{-7}

Table 4.4 presents the comparison of absolute errors between the exact solution and the numerical solutions obtained using the 5-iteration PIM, the 5-iteration MPIM, and the RK4 method for the Riccati equation with the initial condition $u_0 = 1.5$. It can be observed that the absolute error of the PIM increases rapidly as time progresses. The error starts at 8.8256×10^{-2} at $t = 1$, increases to 1.2825 at $t = 2$, and reaches 7.4411×10^{1}, 1.5790×10^{3}, and 1.2565×10^{7} at $t = 3$, $t = 4$, and $t = 5$, respectively. This indicates that the classical PIM suffers from significant error accumulation over the considered interval. In contrast, the MPIM produces much smaller absolute errors compared with the classical PIM. The MPIM error decreases from 1.3640×10^{-3} at $t = 1$ to 1.9690×10^{-6} at $t = 5$, showing that the modified approach is able to control the growth of error more effectively. Meanwhile, the RK4 method gives the smallest absolute errors overall, ranging from 10^{-7} to 10^{-9}. Therefore, although MPIM does not reach the same accuracy level as RK4, it demonstrates a substantial improvement over the classical PIM in terms of error reduction and numerical stability for this case.

4.4 Duffing Equation

Duffing equation is a nonlinear differential equation, which is used in many sciences such as physical, engineering, and even biological problems. It is discovered by electrical engineer German Duffing in 1918 and has named after him. It is worth to mention, this equation is founded both with Van der pol's equation, and Van der pol's considered as most well-known examples of nonlinear oscillation in research papers. Also, the Japanese scientist Ueda proposed excellent Poincare' maps of the Duffing equation which called Japanese attractor or Ueda attractor (Sheu et al., 2007). Duffing equation occurs as a result of the motion of a body subjected to a nonlinear spring power, linear sticky damping, and periodic powering. Oscillations of mechanical systems under the action of a periodic external force can be revealed given by using Duffing equation, see (Yusufoğlu, 2006) and references therein.

The reason behind the importance of Duffing equations is used in many of the areas such as physical, engineering, and even biological problems (Sheu et al., 2007), such as research large amplitude oscillation of centrifugal governor systems, nonlinear vibration of beams and plates (Feng et al., 2006; Nourazar & Mirzabeigy, 2013), magnetic-pliancy mechanical systems (Nourazar & Mirzabeigy, 2013), classical oscillator in chaotic systems (Balaji, 2014), periodic orbit extraction, nonlinear mechanical oscillators and prediction of diseases (Balaji, 2014; Turkyilmazoglu, 2012). Duffing equation produces a helpful model for researching nonlinear oscillations and chaotic dynamical systems (Feng et al., 2006). Both chaos and chaotic system are nonlinear by nature. In addition, we can find them in many natural and artificial systems, and differentiate by sensitivity to initial condition (Khan et al., 2012).

In this section, the applications of the Picard nth-order and Multistage Picard *n*th order for the damping and undamping Duffing equations will be shown to assess the efficiency of the method. the general form of Duffing equation (Yusufoğlu, 2006) will be solved which given in the following form:

$$u''(x)+k_1u'(x)+k_2u(x)+k_3u^3(x)=f(x). \tag{4.40}$$

With initial condition

$$u(0) = a, \quad \text{and} \quad u'(0) = b, \tag{4.41}$$

where k_1, k_2, k_3, a, and b are real constants. In this equation, it is clear that it is a nonlinear ordinary differential equation and it is from the second order (Sheu et al., 2007). In this formulation, k_1 denotes the damping coefficient, k_2 represents the stiffness parameter, and k_3 characterizes the nonlinearity of the restoring force. The parameter r defines the amplitude of the periodic driving force, whereas w denotes its frequency (Bender & Orszag, 2013).

4.4.1 Damping Duffing Equations

Consider damping Duffing equation by Bülbül and Sezer (2013):

$$u''(x) + u'(x) + u^3(x) = 0. \tag{4.42}$$

With initial conditions:

$$u(0) = 1, \quad \text{and} \quad u'(0) = 1. \tag{4.43}$$

4.4.1.1 PIMn and MPIMn Solutions

PIM*n*:

According to the PIMn framework in (3.22), the iterative scheme for the equation (4.42) can be expressed as follows:

$$u_{(n+1)}(t) = u(0)(t) + u'(0)(t)t + \int_0^t (t-s)(-u_n'(s) - u_n^3(s))ds. \tag{4.44}$$

By substituting the initial conditions defined in (4.43) into equations (4.44), the following expressions are obtained.

For $n = 0$:

$$u_1(t) = 1 + t - t^2 - \frac{1}{2}t^3 - \frac{1}{4}t^4 - \frac{1}{20}t^5. \tag{4.45}$$

The iterative process converges at $n = 2$, where the approximate solution $u_3(t)$ is obtained. At this stage, the tolerance condition $\varepsilon < 10^{-5}$ is achieved, indicating that

further iterations would yield negligible changes in accuracy. Further details regarding the equation can be found in Appendix C.

MPIM*n*:

According to the MPIM*n* framework in (3.58), the iterative scheme for the equation (4.42) can be expressed as follows

$$u_{(n+1),j}(t) = u_{m,j-1}(t_{j-1}) + u'_{m,j-1}(t_{j-1}) + \int_{t_{j-1}}^{t} (t-s)(-u'_n(s) - u_n^3(s))ds. \quad (4.46)$$

By substituting the initial conditions defined in (4.43) into equations (4.46), the following expressions are obtained.

For $n = 2, j = 1$**:**

$$\begin{aligned} u_{3,1}(t) = 1 &- 2t - 0.5t^2 + 0.5t^3 + 1.875t^4 + 0.8t^5 - 0.475t^6 - 0.45714t^7 \\ &- 0.20625t^8 + 0.0125t^9 + 0.06125t^{10} + 0.040227t^{11} \\ &+ 0.016563t^{12} + 0.0046635t^{13} + 0.0009375t^{14} + 0.000125t^{15} \\ &+ 7.8125 \times 10^{-6} t^{16}. \end{aligned} \quad (4.47)$$

The iterative process was continued up to $n = 2$, $j = 600$. Given the extensive nature of the iterative expressions, only the representative iterations ($n = 2, j = 1$), ($n = 2, j = 2$) and ($n = 2, j = 600$) are presented in this section, while the complete series are provided in the Appendix C.

4.4.1.2 Result and Discussion

Figure 4.3 illustrates the numerical solutions of the damping Duffing equation obtained using the 3-iteration PIM*n*, the 3-iteration MPIM*n* with a time step of $\Delta t = 0.01$, RK4 with a time step of $\Delta t = 0.01$, and RK45. In this case, RK45 is used as the reference solution due to its adaptive error control capability, good numerical stability for nonlinear systems, and its recognition as a standard numerical solver in commercial software such as Maple. From the figure, it can be observed that the MPIM*n*, RK4, and RK45 solutions show close agreement throughout the interval considered. At the initial stage, the PIM*n* solution also follows the same trend as the reference solution.

However, as t increases, the PIMn solution begins to deviate from the RK45 reference, particularly after approximately $t = 0.5$. The deviation becomes more noticeable as time progresses, where the PIMn curve increases rapidly and moves away from the overall trend of the RK45, RK4, and MPIMn solutions. In contrast, the MPIMn solution remains close to the RK45 reference and is also consistent with the RK4 solution over the given time interval. This indicates that the multistage implementation in MPIMn helps to reduce the accumulation of numerical error compared with the standard PIMn. Therefore, based on this graphical comparison, MPIMn provides a more stable approximation than PIMn for the damping Duffing equation within the interval studied. However, further quantitative evaluation using absolute error analysis is required to support the graphical observation more rigorously.

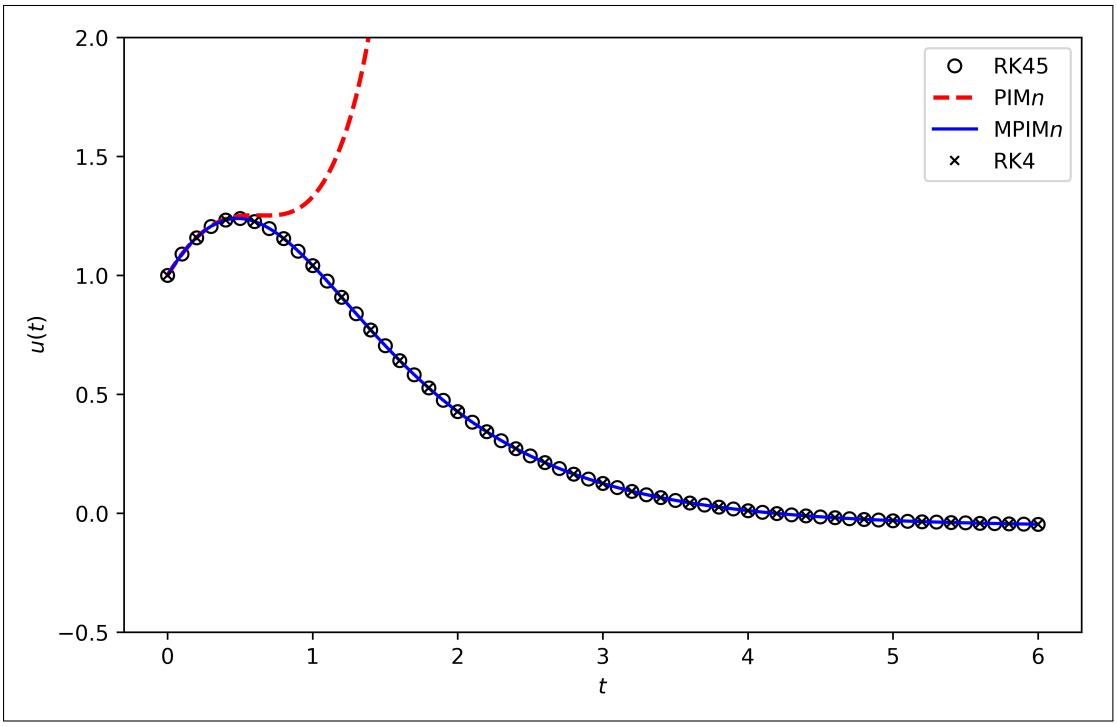

Figure 4.3 Solution of Damping Duffing Equation using 3-iteration PIMn, 3-iteration MPIMn ($\Delta t = 0.01$), RK4 ($\Delta t = 0.01$) and the RK45.

Table 4.5 presents the absolute error comparison between the RK45 reference solution and the numerical solutions obtained using PIMn, MPIMn, and RK4 for the damping Duffing equation. In this case, the absolute error is computed as the difference between the RK45 solution and each numerical method.

It can be observed that the error of PIMn increases rapidly as time progresses. At $t = 1$, the error is 2.8717576×10^{-1}, indicating that the PIMn solution has

already started to deviate from the RK45 reference solution. The error then increases substantially to 2.96540311×10^1 at $t = 2$, 6.973084626×10^3 at $t = 3$, and continues to grow until it reaches 1.110047361×10^8 at $t = 6$. This trend suggests that the standard PIMn experiences significant error accumulation for this problem over the interval considered.

In contrast, the MPIMn method produces very small errors throughout the same interval. The errors remain in the order of 10^{-6}, ranging from 2.06529×10^{-6} to 9.89566×10^{-6}. This indicates that the MPIMn solution remains very close to the RK45 reference solution. Similarly, RK4 also shows small errors, mostly in the order of 10^{-5}, with values ranging from 9.1218×10^{-6} to 4.36285×10^{-5}.

Overall, the results show that both MPIMn and RK4 provide close approximations to the RK45 reference solution, while PIMn shows a clear increase in error as time increases. Based on this error comparison, MPIMn demonstrates better agreement with RK45 than PIMn for the damping Duffing equation within the tested interval. This supports the graphical observation that the multistage implementation in MPIMn helps to control error growth more effectively than the standard PIMn.

Table 4.5
Comparison of Absolute Errors Between the RK45 Reference Solution and 3-iteration PIMn, 3-iteration MPIMn, and RK4 for the Damping Duffing equation.

Time	$\lvert$RK45 $-$ PIM$n\rvert$	$\lvert$RK45 $-$ MPIM$n\rvert$	$\lvert$RK45 $-$ RK4$\rvert$
1	2.8717576×10^{-1}	8.2015×10^{-6}	4.0890×10^{-5}
2	2.96540311×10^{1}	4.9034×10^{-6}	9.1218×10^{-6}
3	$6.973084626 \times 10^{3}$	2.06529×10^{-6}	2.71181×10^{-5}
4	3.54458234×10^{5}	6.46964×10^{-6}	3.90126×10^{-5}
5	8.07186216×10^{5}	8.8029×10^{-6}	4.26325×10^{-5}
6	$1.110047361 \times 10^{8}$	9.89566×10^{-6}	4.36285×10^{-5}

4.4.2 Undamping Duffing Equations

Consider the undamping Duffing equation by Bülbül and Sezer (2013):

$$u''(x) + 3u(x) - 2u^3(x) = cos(x)sin(2x). \tag{4.48}$$

With initial conditions:

$$u(0) = 0, \quad \text{and} \quad u'(0) = 1. \tag{4.49}$$

4.4.2.1 PIMn and MPIMn Solutions

PIM*n*:

According to the PIM*n* framework in (3.22), the iterative scheme for the equation (4.48) can be expressed as follows:

$$u_{(n+1)}(t) = u(0)(t) + u'(0)(t)\,t + \int_0^t (t-s)(cos(t)sin(2t) - 3u(t) + 2u^3(t))ds. \tag{4.50}$$

By substituting the initial conditions defined in (4.49) into equations (4.50), the following expressions are obtained.

For $n = 0$

$$u_1(t) = -\frac{1}{2}\sin(t) - \frac{1}{18}\sin(3t) + \frac{5}{3}t - \frac{1}{2}t^3 + \frac{1}{10}t^5 \tag{4.51}$$

The iterative process converges at $n = 1$, where the approximate solution $u_2(t)$ is obtained. At this stage, the tolerance condition $\varepsilon < 10^{-5}$ is achieved, indicating that further iterations would yield negligible changes in accuracy. Further details and other iterations equation's can be found in Appendix C.

MPIM*n*:

According to the MPIM*n* framework in (3.58), the iterative scheme for the equation (4.48) can be expressed as follows

$$u_{(n+1),j}(t) = u_{m,j-1}(t_{j-1}) + u'_{m,j-1}(t_{j-1}) + \int_{t_{j-1}}^t (t-s)(cos(t)sin(2t) - 3u(t) + 2u^3(t))ds. \tag{4.52}$$

By substituting the initial conditions defined in (4.49) into equations (4.52), the following expressions are obtained.

For $n = 1, j = 1$:

$$\begin{aligned} u_{2,1}(t) = &-1.3158 \times 10^{6} \sin t - 0.78993 \sin(2t) - 0.84098 \sin(3t) - 0.011366 \sin(4t) \\ &- 0.00074074 \sin(5t) - 0.00015956 \sin(6t) - 0.000047241 \sin(7t) - 1.0584 \\ &\times 10^{-6} \sin(9t) + 1.1935 \times 10^{6} \cos t\, t + 1.1181 \cos(2t)\, t + 1.9698 \cos(3t)\, t \\ &+ 0.025662 \cos(4t)\, t + \ldots + 7.3529 \times 10^{-6} t^{17} \end{aligned} \tag{4.53}$$

The iterative process was continued up to $n = 2$, $j = 600$. Given the extensive nature of the iterative expressions, only the representative iterations ($n = 2, j = 1$), ($n = 2, j = 2$) and ($n = 2, j = 600$) are presented in this section, while the complete series are provided in the Appendix C.

4.4.2.2 Result and Discussion

Figure 4.4 illustrates the numerical solutions of the undamped Duffing equation obtained using the 2-iteration PIM*n*, the 2-iteration MPIM*n* with a time step of $\Delta t = 0.01$, RK4 with the same time step, and the exact solution. At the early stage of the simulation, both PIM*n* and MPIM*n* show close agreement with the RK4 and exact solutions, indicating that both methods are able to capture the initial behaviour of the undamped Duffing system.

As time increases, the PIM*n* solution begins to move away from the exact solution and the RK4 solution, particularly near the peak region around $t \approx 1.5$. After this point, the deviation becomes more visible, where the PIM*n* curve increases beyond the main trend shown by the exact, RK4, and MPIM*n* solutions. This suggests that, for this example, the 2-iteration PIM*n* approximation is affected by error growth as the time interval increases.

In contrast, the MPIM*n* solution remains close to the exact solution and RK4 solution throughout the interval considered. The curve follows the general oscillatory pattern of the exact solution, including the increasing and decreasing behaviour of $u(t)$. This observation indicates that the multistage formulation in MPIM*n* helps the approximation remain closer to the reference solutions compared with the standard PIM*n* under the same number of iterations.

Overall, the graphical result shows that MPIM*n* gives better agreement with the

exact solution than PIMn for the undamped Duffing equation within the tested interval. However, this conclusion is based on the visual comparison, and the absolute error analysis is used to provide a more quantitative evaluation of the differences among the methods.

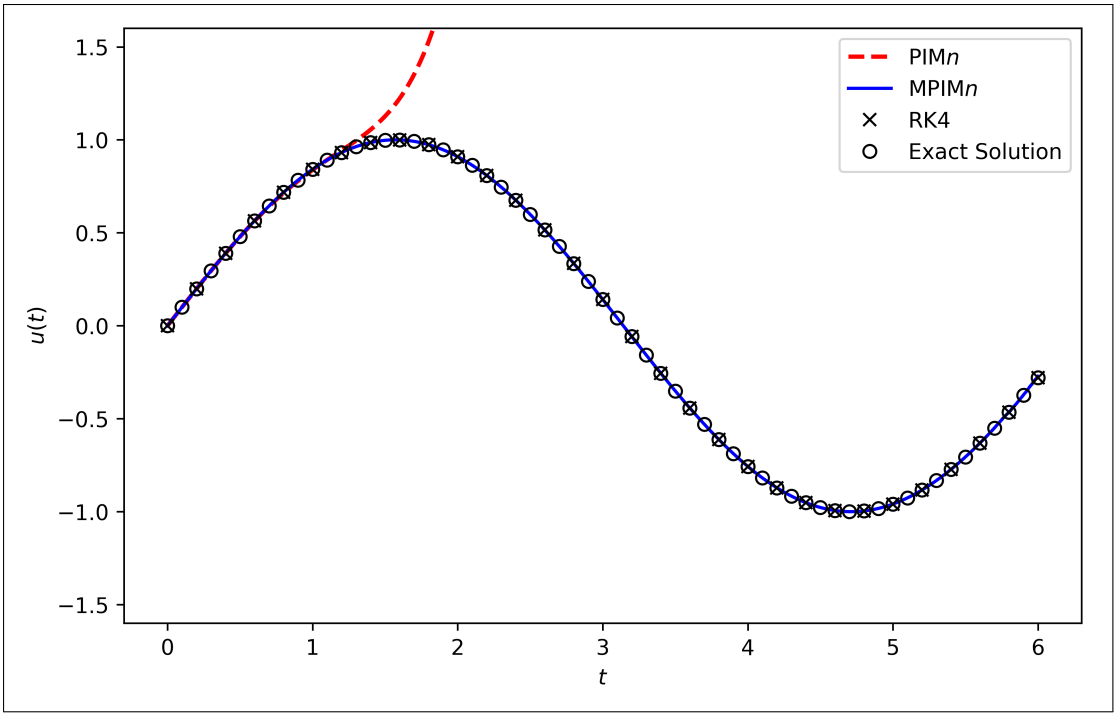

Figure 4.4 Solution of Undamping Duffing Equation using 2-iteration PIMn, 2-iteration MPIMn ($\Delta t = 0.01$), the RK4 ($\Delta t = 0.01$) and the Exact Solution.

Table 4.6 presents the absolute error comparison between the exact solution and the numerical solutions obtained using the 2-iteration PIMn, 2-iteration MPIMn with the time step $\Delta t = 0.01$, and RK4 with the same time step for the undamped Duffing equation.

From the table, it can be observed that the absolute error of PIMn increases as time progresses. At $t = 1$, the error is 0.0033798110, which is relatively small compared with the later time levels. However, the error increases to 1.184910276 at $t = 2$ and 15.56542451 at $t = 3$. A further increase is observed at the later time levels, where the error reaches 78.23168002 at $t = 4$, 259.7355927 at $t = 5$, and 680.0608448 at $t = 6$. This trend indicates that the PIMn solution gradually moves away from the exact solution over the interval considered when only two iterations are used.

In comparison, MPIMn gives smaller absolute errors than PIMn at all selected time levels. The errors are very small at the early and intermediate time levels, with

values of 8.0×10^{-10} at $t = 1$, 2.8×10^{-9} at $t = 2$, and 3.61×10^{-8} at $t = 3$. At $t = 4$ and $t = 5$, the errors are 1.23×10^{-8} and 4.857×10^{-7}, respectively. Although the MPIMn error increases to 0.0070272460 at $t = 6$, the value remains smaller than the corresponding PIMn error at the same time level. This suggests that, under the present computational setting, MPIMn is able to maintain closer agreement with the exact solution than PIMn.

The RK4 method also produces small absolute errors throughout the tested interval. Its error increases from 6.9952×10^{-6} at $t = 1$ to 1.96272×10^{-5} at $t = 2$, 4.55429×10^{-5} at $t = 3$, and 3.81905×10^{-4} at $t = 6$. Compared with MPIMn, RK4 gives larger errors from $t = 1$ to $t = 5$. However, at $t = 6$, RK4 gives a smaller error than MPIMn, where the RK4 error is 3.81905×10^{-4}, while the MPIMn error is 0.0070272460.

Overall, the results indicate that MPIMn gives lower absolute errors than PIMn for all selected time levels in this example. MPIMn also remains closer to the exact solution than RK4 from $t = 1$ to $t = 5$, whereas RK4 gives a smaller error at $t = 6$. Therefore, based on this error comparison, MPIMn shows better agreement with the exact solution than PIMn for the undamped Duffing equation within the tested interval and under the specified time step and iteration number.

Table 4.6
Comparison Absolute Error Between Exact Solution and 2-iteration PIMn, 2-iteration MPIMn on the Time Step ($\Delta t = 0.01$), and RK4 on the Time Step ($\Delta t = 0.01$) for the Undamping Duffing Equation

Time	|Exact – PIMn|	|Exact – MPIMn|	|Exact – RK4|
1	0.0033798110	$8.000000000 \times 10^{-10}$	$6.995200000 \times 10^{-6}$
2	1.184910276	$2.800000000 \times 10^{-9}$	$1.962720000 \times 10^{-5}$
3	15.56542451	$3.610000000 \times 10^{-8}$	$4.554290000 \times 10^{-5}$
4	78.23168002	$1.230000000 \times 10^{-8}$	$1.177370000 \times 10^{-5}$
5	259.7355927	$4.857000000 \times 10^{-7}$	$1.207530000 \times 10^{-4}$
6	680.0608448	0.0070272460	$3.819050000 \times 10^{-4}$

4.5 Pohlhausen-Flow Equation

The Pohlhausen flow represents a special case of the Falkner–Skan family of boundary layer equations, which describe the steady two–dimensional laminar flow of an incompressible, viscous fluid over a flat or wedge–shaped surface. It is derived from the similarity transformation of the Navier–Stokes equations under appropriate assumptions for velocity and pressure gradients. This equation is particularly important in boundary–layer theory, as it provides one of the earliest analytical approximations to laminar flow profiles within the viscous sublayer.

The general equation of the Pohlhausen flow, derived from the Falkner–Skan family of equations (Asaithambi, 2021), is given by

$$f''' + ff'' + \beta\left(1 - (f')^2\right) = 0, \tag{4.54}$$

where β denotes the pressure–gradient parameter and the primes represent differentiation with respect to the similarity variable η. The equation is subject to the boundary conditions

$$f(0) = A, \qquad f'(0) = B, \qquad f''(0) = C, \tag{4.55}$$

where A, B, and C are prescribed constants.

The Pohlhausen equation is particularly useful as a benchmark problem because its exact solution allows for direct comparison with approximate and numerical schemes such as the PIMn and the MPIMn. It also serves as a validation case for higher–order numerical solvers developed for the Falkner–Skan family of equations.

In the context of the present study, the Pohlhausen flow equation is utilized to assess the accuracy, stability, and convergence behaviour of both PIMn and the MPIMn. Its relatively simple structure, combined with the availability of an analytical solution, makes it an ideal candidate for verifying the reliability of these modified iterative approaches in modelling nonlinear viscous flow dynamics.

4.5.1 PIM*n* and MPIM*n* solutions

Consider the nonlinear boundary value problem by Asaithambi (2021) where β is the pressure–gradient parameter and the primes denote differentiation with respect to the similarity variable η, the case corresponding to $(\beta_0, \beta) = (0, 1)$ yields the Pohlhausen flow equation (Pohlhausen, 1921),

$$f''' + \left(1 - (f')^2\right) = 0, \tag{4.56}$$

subject to the boundary conditions

$$f(0) = 0, \quad f'(0) = 0, \quad f''(0) = 1, \tag{4.57}$$

and the exact solution for this problem given by

$$f'(t) = 3\tanh^2\left(\frac{t}{\sqrt{2}} + \tanh^{-1}\sqrt{\frac{2}{3}}\right) - 2. \tag{4.58}$$

This solution satisfies the boundary conditions (4.57) and represents the dimensionless velocity distribution within the laminar boundary layer.

PIM*n*:

According to the PIM*n* framework in (3.22), the iterative scheme for the equation (4.56) can be expressed as follows:

$$f_{(n+1)}(t) = f(0) + f'(0)\,t + \frac{f''(0)\,t^2}{2} + \int_0^t \frac{(t-s)^2}{2!}\left((f_n'(s))^2 - 1\right) ds. \tag{4.59}$$

The initial seed is taken as $y_0(t) = \alpha t$. By substituting the initial conditions defined in (4.57) and $\alpha = 1.154701$ into equations (4.59), the following expressions are obtained.

For $n = 0$:

$$f_1(t) = 0.57735\,t^2 - \frac{1}{6}t^3. \tag{4.60}$$

The iterative process converges at $n = 2$, where the approximate solution $f_3(t)$ is obtained. At this stage, the tolerance condition $\varepsilon < 10^{-10}$ is achieved, indicating

that further iterations would yield negligible changes in accuracy. Further details and other iterations equation's can be found in Appendix D.

MPIM*n*:

According to the MPIM*n* framework in (3.58), the iterative scheme for the equation (4.56) can be expressed as follows

$$f_{(n+1),j}(t) = f_{m,j-1}(t_{j-1}) + f'_{m,j-1}(t_{j-1}) + \frac{f''_{m,j-1}(t_{j-1}^2)}{2} + \int_{t_{j-1}}^{t} \frac{(t-s)^2}{2!}((f')^2 - 1)ds. \tag{4.61}$$

By substituting the initial conditions defined in (4.57) into equations (4.61), the following expressions are obtained.

For $n = 2, j = 1$:

$$f_{3,1}(t) = 0.57735t^2 - 0.16667t^3 + 0.022222t^5 - 0.0096225t^6 + 0.0011905t^7 + 0.00076369t^8 - 0.00022046t^9 + 0.000012470t^{11}. \tag{4.62}$$

The iterative process was continued up to $n = 2$, $j = 600$. Given the extensive nature of the iterative expressions, only the representative iterations ($n = 2, j = 1$) is presented in this section, while the complete series are provided in the Appendix D.

4.5.2 Results and discussion

Figure 4.5 illustrates the numerical solutions of the Pohlhausen flow equation obtained using 3-iteration of PIM*n*, 3-iteration of MPIM*n*, RK4 with $\Delta t = 0.01$, together with the exact solution. From the figure, it is clearly observed that MPIM*n* and RK4 almost perfectly overlap with the exact solution throughout the entire interval $0 \leq t \leq 6$, indicating excellent accuracy and stability. The plotted points of RK4 and the continuous curve of MPIM*n* are visually indistinguishable from the exact solution, demonstrating that both methods are capable of accurately capturing the true behavior of the solution. In contrast, PIM*n* begins to deviate from the exact solution as t increases. Although it initially follows the same trend near $t = 0$, the red dashed curve gradually diverges after approximately $t \approx 1.5$, and the deviation becomes

significant beyond $t \approx 2.5$. The PIMn solution fails to maintain the increasing trend of the exact solution and instead bends downward, indicating instability and accumulation of approximation error. Overall, the graphical comparison highlights that the multistage modification significantly improves the stability and accuracy of the classical Picard approach, while confirming the robustness of the Picard-based framework as an efficient numerical approach for modeling boundary-layer type fluid flow problems.

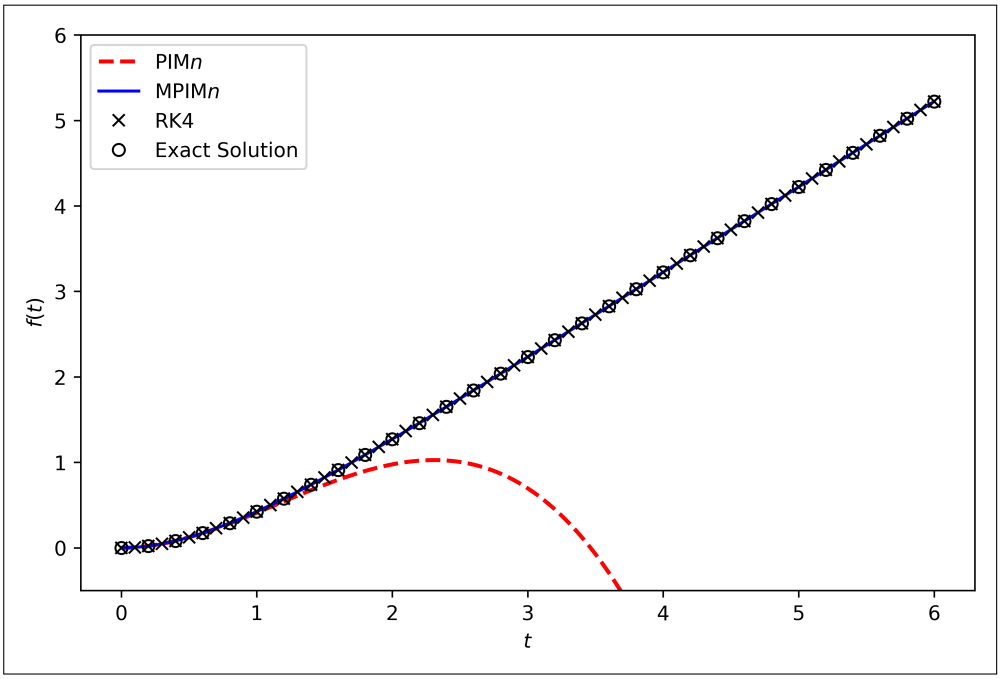

Figure 4.5 Solution of Pohlhausen-Flow Equation using 3-iteration PIMn, 3-iteration MPIMn on the Time Step ($\Delta t = 0.01$), RK4 on the Time Step ($\Delta t = 0.01$) and the Exact Solution.

Table 4.7 presents the comparison of absolute errors between the exact solution and the 3-iteration PIMn, 3-iteration MPIMn, and RK4 using the same time step size $\Delta t = 0.01$ for the Pohlhausen flow equation. It can be observed that the error of PIMn increases significantly as time increases, starting from 1.4178×10^{-2} at $t = 1$, rising to 2.9576×10^{-1} at $t = 2$, 1.5376×10^{0} at $t = 3$, 4.6535×10^{0} at $t = 4$, 1.0622×10^{1} at $t = 5$, and reaching 2.0437×10^{1} at $t = 6$. This indicates that the standard PIMn experiences considerable error accumulation over the given interval. In contrast, MPIMn maintains very small absolute errors throughout the interval. The errors are 2.4500×10^{-7} at $t = 1$, 1.3490×10^{-6} at $t = 2$, 5.5730×10^{-6} at $t = 3$, 2.2547×10^{-5} at $t = 4$, 9.1951×10^{-5} at $t = 5$, and 3.7702×10^{-4} at $t = 6$. It

is important to note that the MPIMn errors are identical to the RK4 errors at $t = 1$ and $t = 2$ based on the reported rounded values. For the remaining time points, the errors of MPIMn remain very close to those of RK4, with only minor differences observed from $t = 3$ to $t = 6$. Overall, the results show that the multistage modification improves the accuracy of PIMn and reduces the accumulation of error over time. The close agreement between MPIMn and RK4 suggests that MPIMn is able to achieve comparable numerical accuracy to the classical RK4 method for the Pohlhausen flow equation within the considered time interval.

Table 4.7
Comparison of Absolute Errors Between Exact Solution and 3-iteration PIMn, 3-iteration MPIMn on the Time Step ($\Delta t = 0.01$) and the RK4 with the Time Step ($\Delta t = 0.01$) for the Pohlhausen Flow Equation.

Time	$\lvert$**Exact** $-$ **PIMn**$\rvert$	$\lvert$**Exact** $-$ **MPIMn**$\rvert$	$\lvert$**Exact** $-$ **RK4**$\rvert$
1	1.4178×10^{-2}	2.4500×10^{-7}	2.4500×10^{-7}
2	2.9576×10^{-1}	1.3490×10^{-6}	1.3490×10^{-6}
3	1.5376×10^{0}	5.5730×10^{-6}	5.5750×10^{-6}
4	4.6535×10^{0}	2.2547×10^{-5}	2.2554×10^{-5}
5	1.0622×10^{1}	9.1951×10^{-5}	9.1980×10^{-5}
6	2.0437×10^{1}	3.7702×10^{-4}	3.7714×10^{-4}

4.6 Concluding Remarks

This chapter presented the application of PIM, MPIM, PIMn, and MPIMn to selected nonlinear ordinary differential equations. The chapter began with the development of the PIMn formulation, which extends the classical Picard Iterative Method to directly solve higher-order initial value problems without converting them into systems of first-order equations. This formulation provides a more direct and systematic approach for handling higher-order nonlinear differential equations. The performance of PIM and MPIM was first examined through the first-order nonlinear Riccati equation. The results showed that the classical PIM is capable of producing accurate approximations within a limited interval, but its accuracy decreases as time increases. In contrast, MPIM demonstrated better numerical stability and maintained closer agreement with the exact solution and RK4 results. This indicates that the multistage modification improves the long-term performance of the Picard-based method. The proposed PIMn and MPIMn were then applied to the nonlinear Duffing

equation to evaluate their capability in solving second-order nonlinear oscillatory problems. The results showed that PIM*n* provides acceptable accuracy at the early stage of the simulation, but its error increases as the time interval becomes longer. Meanwhile, MPIM*n* produced more stable and accurate results, showing closer agreement with both the exact solution and RK4.

In addition, the Pohlhausen flow equation was included to further assess the applicability of PIM*n* and MPIM*n* for higher-order nonlinear differential equations. The numerical results demonstrated that PIM*n* can be applied directly to this higher-order problem, but its accuracy is limited over a longer interval. On the other hand, MPIM*n* showed improved stability and smaller absolute errors, indicating that the multistage structure is effective in reducing error accumulation. Overall, the findings in this chapter confirm that the proposed PIM*n* formulation provides a direct framework for solving higher-order nonlinear ODEs, while the MPIM and MPIM*n* improve the accuracy and stability of the classical Picard approach. The comparison with exact solutions and RK4 further supports the reliability of the proposed methods, particularly for nonlinear problems that require stable approximations over longer time intervals.

CHAPTER 5
APPLICATION OF PIM AND MPIM TO PDES

5.1 Introduction

In this chapter, the numerical performance of the Picard Iterative Method (PIM) and the Multistage Picard Iterative Method (MPIM) is extended and validated for solving nonlinear partial differential equations (PDEs). While these methods have been successfully applied to various nonlinear ordinary differential equations (ODEs) in the earlier chapters, their capability in handling nonlinear PDEs has not been extensively examined. Therefore, this chapter aims to investigate whether PIM and MPIM can accurately and efficiently solve nonlinear PDE models commonly encountered in physical and engineering sciences.

This chapter focuses on the Newell-Whitehead-Segel equation, which is a nonlinear reaction–diffusion equation widely used to describe pattern formation, instability phenomena, and nonlinear wave propagation. The equation is selected as the main PDE model to evaluate the applicability of PIM and MPIM in solving nonlinear PDE problems. In this study, both methods are applied directly to the Newell-Whitehead-Segel equation, and the numerical solutions obtained are compared with the exact solution and the fourth-order Runge-Kutta (RK4) numerical benchmark. The RK4 values are generated using Maple's built-in numerical solver to provide a reliable reference for comparison.

5.2 Newell Whitehead Segel Equation

The Newell–Whitehead–Segel equation is a class of nonlinear partial differential equations widely employed in modeling various phenomena in fluid mechanics. This equation is used for some problems in various systems, for example, Faraday instability, biological systems, nonlinear optics, Rayleigh-Benard convection, and chemical reactions. The equation acquired by Newell, Whitehead, and Segel is as follows (Newell & Whitehead, 1969; Segel, 1969):

$$u_t(x,t) = \alpha u_{xx}(x,t) + bu(x,t) - cu^m(x,t), \tag{5.1}$$

$$u(x,0) = f(x), \tag{5.2}$$

where b, c are real numbers and α, m are positive integers.

In this section, the PIM and MPIM will be used to find the approximate solution of the Newell-Whitehead-Segel equation. The result of the finding will be compared to the exact solution and the RK4. Consider the Newell Whitehead Segel equation by Latif et al. (2020)

$$u_t(x,t) = u_{xx}(x,t) - 3u(x,t), \tag{5.3}$$

and the initial condition

$$u(x,0) = e^{2x}, \tag{5.4}$$

and the corresponding exact solution is

$$u(x,t) = e^{2x+t}, \tag{5.5}$$

where the parameters represent the diffusion rate, growth rate, and nonlinear interaction term, respectively.

5.2.1 PIM and MPIM solutions

PIM:

According to the PIM framework, the iterative scheme for the Newel-Whitehead Segel equation (5.3) can be expressed as follows:

$$u_{(n+1)}(x,t) = u(0)(x,t) + \int_0^t [u_{(n),xx}(x,s) - 3u_{(n)}(x,s)]ds. \tag{5.6}$$

By substituting the initial condition in (5.4) into equations (5.6), the following expressions are obtained.

For $n = 0$:

$$u_1(x,t) = e^{2x} + e^{2x}t. \tag{5.7}$$

The iterative process converges at $n = 3$, where the approximate solution $u_4(t)$ is obtained. At this stage, the tolerance condition $\varepsilon < 10^{-5}$ is achieved, indicating that further iterations would yield negligible changes in accuracy. Further details and other iterations equation's can be found in Appendix H.

MPIM:

According to the MPIM framework, the iterative scheme for the Newel-Whitehead Segel equation (5.3) can be expressed as follows:

$$u_{n+1,j}(x,t) = u_{0,j-1}(x,t_{j-1}) + \int_{t_{j-1}}^{t} [u_{n,j,xx}(x,s) - 3u_{n,j}(x,s)]ds. \tag{5.8}$$

By substituting the initial conditions (5.4) into equations (5.8), the following expressions are obtained.

For $n = 3, j = 1$:

$$u_{4,1}(x,t) = \mathrm{e}^{2x} + \mathrm{e}^{2x}t + \frac{1}{2}\mathrm{e}^{2x}t^2 + \frac{1}{6}\mathrm{e}^{2x}t^3 + \frac{1}{24}\mathrm{e}^{2x}t^4. \tag{5.9}$$

The iterative process was continued up to $n = 3$, $j = 600$. Given the extensive nature of the iterative expressions, only the representative iterations ($n = 3, j = 1$) is presented in this section, while the complete series are provided in the Appendix H.

5.2.2 Results and discussion

Figure 5.1 shows the numerical comparison between the 4-iteration PIM, the 4-iteration MPIM with time step $\Delta t = 0.01$, and RK4 with $\Delta t = 0.01$ for approximating the solution of the Newell–Whitehead–Segel equation. The results are compared with the exact solution over the interval $0 \leq t \leq 6$. From the graph, the MPIM curve follows the trend of the exact solution closely throughout the considered interval. The RK4 approximations also remain close to the exact solution, as indicated by the marker positions lying near the exact solution curve. This suggests that both MPIM and

RK4 provide consistent approximations for the selected step size and time interval. In contrast, the PIM solution shows a visible deviation from the exact solution as time increases. The difference becomes more noticeable after approximately $t = 3$, where the PIM curve grows at a slower rate compared with the exact, MPIM, and RK4 solutions. This indicates that the standard PIM may be less suitable for maintaining accuracy over a longer interval for this nonlinear PDE example. Overall, the graphical comparison suggests that the multistage implementation improves the approximation behaviour of the Picard-based method for the Newell–Whitehead–Segel equation. However, the graphical observation should be supported by the absolute error analysis to further confirm the accuracy of each method.

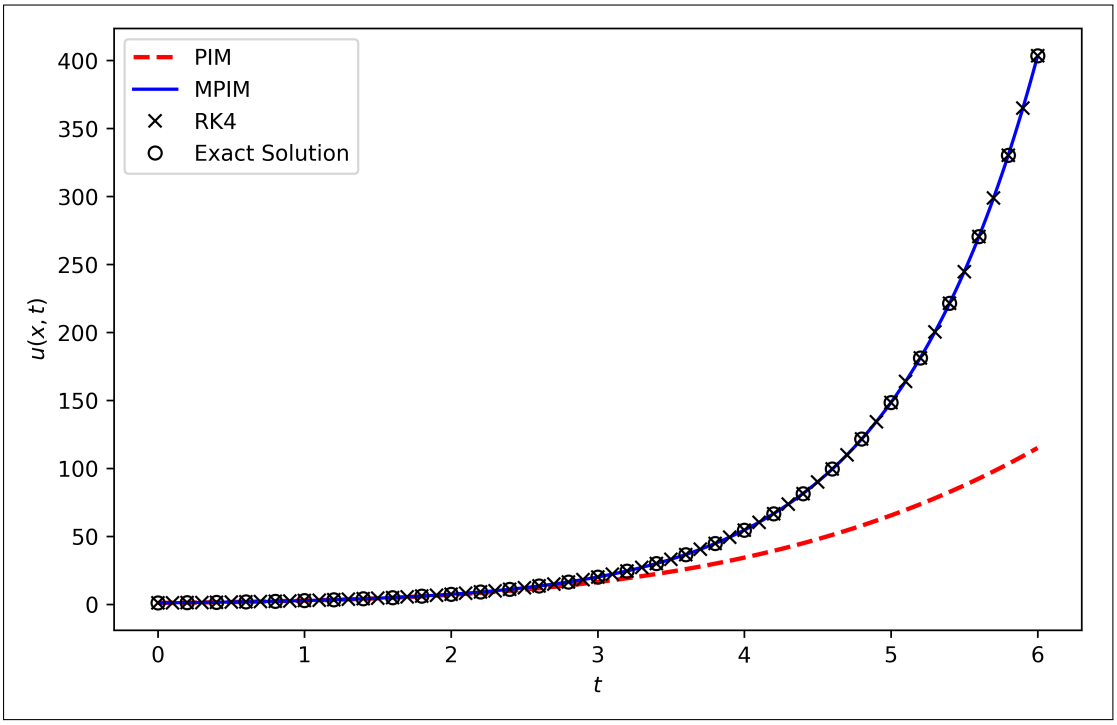

Figure 5.1 Solution of Newell Whitehead Segel Equation using 4-iteration PIM, 4-iteration MPIM($\Delta t = 0.01$) RK4 ($\Delta t = 0.01$) and the Exact Solution.

Table 5.1 presents the absolute errors of PIM, MPIM, and RK4 compared with the exact solution for the Newell-Whitehead-Segel equation at selected time values. From the results, the error of the classical PIM increases as time progresses. At $t = 1$, the PIM error is relatively small, with a value of 9.98495×10^{-3}. However, the error increases to 3.89056×10^{-1} at $t = 2$ and becomes much larger at later time values, reaching 2.88429×10^{2} at $t = 6$. This shows that the standard PIM approximation gradually moves away from the exact solution over the considered interval.

In contrast, MPIM produces much smaller absolute errors throughout the interval. The MPIM error remains in the order of 10^{-10} at $t = 1$, increases gradually to the order of 10^{-8} between $t = 4$ and $t = 5$, and reaches 2.00037×10^{-7} at $t = 6$. A similar pattern is observed for RK4, where the error values are very close to those obtained by MPIM. For example, at $t = 6$, the RK4 error is 2.00043×10^{-7}, which is comparable to the MPIM error.

Overall, the table indicates that MPIM gives a more accurate approximation than the standard PIM for this problem and selected interval. The close agreement between MPIM and RK4 also supports the graphical observation that MPIM follows the exact solution more closely than PIM. However, this conclusion is based on the selected time step, number of iterations, and test problem considered in this study.

Table 5.1
Comparison of Absolute Errors Between Exact solution and 4-iteration PIM, 4-iteration MPIM on the Time Step ($\Delta t = 0.01$) and RK4 on the Time Step ($\Delta t = 0.01$) for the Newell Whitehead Segel Equation.

Time	$\lvert\text{Exact} - \text{PIM}\rvert$	$\lvert\text{Exact} - \text{MPIM}\rvert$	$\lvert\text{Exact} - \text{RK4}\rvert$
1	9.98495×10^{-3}	2.24600×10^{-10}	2.24650×10^{-10}
2	3.89056×10^{-1}	1.22131×10^{-9}	1.22132×10^{-9}
3	3.71054×10^{0}	4.97970×10^{-9}	4.97980×10^{-9}
4	2.06482×10^{1}	1.80481×10^{-8}	1.80485×10^{-8}
5	8.30382×10^{1}	6.13240×10^{-8}	6.13260×10^{-8}
6	2.88429×10^{2}	2.00037×10^{-7}	2.00043×10^{-7}

5.3 Concluding Remarks

In this chapter, the PIM and the MPIM were applied to a nonlinear PDE, namely the Newell-Whitehead-Segel equation. The implementation demonstrates the applicability of both methods in approximating the solution of a nonlinear reaction-diffusion problem. Based on the graphical comparison and absolute error analysis, MPIM shows closer agreement with the exact solution and RK4 benchmark compared with the standard PIM over the selected interval.The results also indicate that the standard PIM produces reasonable approximation at the early stage of the interval but gradually deviates as time increases. In contrast, the multistage formulation helps reduce the accumulation of error by updating the approximation over smaller subintervals. This contributes to better numerical behaviour of MPIM

for the considered PDE example. Overall, the findings suggest that MPIM provides a more reliable approximation than the classical PIM for the Newell-Whitehead-Segel equation under the selected time step and interval. However, the conclusion is limited to the problem, parameter setting, and numerical configuration considered in this study. Further applications to other nonlinear PDE models are recommended to evaluate the broader performance of the method.

CHAPTER 6
APPLICATION OF PIM AND MPIM TO SYSTEM OF ODES

6.1 Introduction

A wide range of analytical and semi-analytical methods have been widely used to obtain approximate solutions for nonlinear differential equations. Despite the considerable development of these techniques, there is still no known work that applies the standard Picard Iterative Method (PIM) to epidemic models such as the Monkeypox Transmission Model. Motivated by this gap in the literature, this chapter focuses on applying the PIM to this epidemiological system and evaluates its capabilities in capturing the nonlinear dynamics of infectious disease transmission.

To address the basic restriction of PIM, particularly its difficulties in achieving global accuracy over long time intervals, this chapter additionally incorporates the proposed Multistage Picard Iterative Method (MPIM). By dividing the time domain into successive subintervals and applying the Picard technique recursively within each segment, MPIM provides a more stable and accurate framework for solving complex nonlinear systems. For the Monkeypox Transmission Model, the numerical results obtained from PIM and MPIM are compared with both the fourth-order Runge-Kutta (RK4) and eighth-order Runge-Kutta (RK8) methods. In addition, results from the Adomian Decomposition Method (ADM) and Multistage Adomian Decomposition Method (MADM) are included as benchmark results to provide a broader comparative assessment of the proposed approach against other established semi-analytical methods.

This chapter provides additional validation of the performance of PIM and MPIM on other classes of nonlinear ODE systems apart from the Monkeypox Transmission Model. First, both methods are applied to a COVID-19 transmission model to evaluate their ability to handle compartmental epidemiological dynamics and to observe how the multistage refinement improves accuracy over the classical Picard framework. The study then proceeds to examine a nonlinear biochemical reaction model, enabling an assessment of the methods' reliability in capturing stiff

and highly nonlinear reaction behaviours. Finally, PIM and MPIM are applied to the multispecies Lotka-Volterra system, allowing an extensive evaluation of accuracy, convergence behaviour, stability, and computational performance in the setting of interacting population models. For these additional systems, the numerical results are compared with the RK4 method only, while the available ADM and MADM results are included as supplementary benchmarks to provide a comprehensive analysis of the strengths and limitations of the classical and multistage iterative methods in solving different nonlinear dynamical systems.

6.2 Monkeypox Transmission Model

In this section, consider the nonlinear epidemiological model that describes the transmission dynamics of the Monkeypox virus among human and rodent populations as proposed by Peter et al. (2022a). The entire human population N_h is separated into five compartments: susceptible S_h, exposed E_h, infected I_h, isolated Q_h, and recovered R_h. Similarly, the rodent population N_r consists of three compartments: susceptible S_r, exposed E_r, and infected I_r. The overall interaction between these two populations is governed by the following nonlinear system of ordinary differential equations:

$$\frac{dS_h}{dt} = \Lambda_h - \beta_1 S_h I_r - \beta_2 S_h I_h - \mu_h S_h, \tag{6.1}$$

$$\frac{dE_h}{dt} = \beta_1 S_h I_r + \beta_2 S_h I_h - (\alpha_1 + \mu_h) E_h, \tag{6.2}$$

$$\frac{dI_h}{dt} = \alpha_1 E_h - (\alpha_2 + \delta_h + \mu_h) I_h, \tag{6.3}$$

$$\frac{dQ_h}{dt} = \alpha_2 I_h - (\tau + \mu_h) Q_h, \tag{6.4}$$

$$\frac{dR_h}{dt} = \tau Q_h - \mu_h R_h, \tag{6.5}$$

$$\frac{dS_r}{dt} = \Lambda_r - \beta_3 S_r I_r - \mu_r S_r, \tag{6.6}$$

$$\frac{dE_r}{dt} = \beta_3 S_r I_r - (\alpha_3 + \mu_r) E_r, \tag{6.7}$$

$$\frac{dI_r}{dt} = \alpha_3 E_r - (\delta_r + \mu_r) I_r. \tag{6.8}$$

The parameters Λ_h and Λ_r denote the recruitment rates of humans and rodents, respectively. The contact rates between humans and rodents are represented by β_1, β_2, and β_3, while α_1, α_2, and α_3 correspond to the progression rates from the exposed

to infected classes in both species. The parameters μ_h and μ_r denote natural death rates, δ_h and δ_r represent disease-induced death rates, τ is the recovery rate from isolation, and ϕ indicates the proportion of undiagnosed human cases.

The system (6.1)–(6.8) is solved under the initial conditions by Peter et al. (2022a)

$$S_h(0) = 10000, \quad E_h(0) = 50, \quad I_h(0) = 0.57, \quad Q_h(0) = 0.43, \quad R_h(0) = 0,$$
$$S_r(0) = 1000, \quad E_r(0) = 100, \quad I_r(0) = 10, \tag{6.9}$$

where all parameters are assumed to be positive constants.

This nonlinear epidemiological system captures the interdependence between the human and rodent populations through both direct and indirect transmission pathways. Its analytical solution is generally intractable due to the strong coupling and nonlinear terms in Equations (6.1)-(6.8).

Hence, the present study seeks to obtain a semi–analytical solution for the system using the PIM and its improved variant, the MPIM. The traditional PIM provides an iterative framework for constructing approximate analytical solutions, while the MPIM extends its applicability by overcoming convergence limitations through stage–wise corrections across subintervals. This enhancement ensures greater stability and accuracy when modelling the complex nonlinear behaviour of Monkeypox transmission over extended time domains.

6.2.1 PIM and MPIM solutions

PIM:

According to PIM, construct an iterative scheme for system (6.1)–(6.8) as follows:

$$S_{h,(n+1)}(t) = S_h(0)(t) + \int_0^t \left(\Lambda_h - \frac{\beta_1 I_{r,(n)}(t) + \beta_2 I_{h,(n)}(t)}{N_h} S_{h,(n)}(t) - \mu_h S_{h,(n)}(t) + \phi Q_{h,(n)}(t) \right) dt, \tag{6.10}$$

$$E_{h,(n+1)}(t) = E_h(0)(t) + \int_0^t \left(\frac{\beta_1 I_{r,(n)}(t) + \beta_2 I_{h,(n)}(t)}{N_h} S_{h,(n)}(t) - (\alpha_1 + \alpha_2 + \mu_h) E_{h,(n)}(t) \right) dt, \tag{6.11}$$

$$I_{h,(n+1)}(t) = I_h(0)(t) + \int_0^t \left(\alpha_2 E_{h,(n)}(t) - (\mu_h + \delta_h + \theta_h) I_{h,(n)}(t) \right) dt, \tag{6.12}$$

$$Q_{h,(n+1)}(t) = Q_h(0)(t) + \int_0^t \left(\alpha_1 E_{h,(n)}(t) - (\phi + \tau + \mu_h + \delta_h) Q_{h,(n)}(t) \right) dt, \tag{6.13}$$

$$R_{h,(n+1)}(t) = R_h(0)(t) + \int_0^t \left(\sigma I_{h,(n)}(t) + \tau Q_{h,(n)}(t) - \mu_h R_{h,(n)}(t) \right) dt, \tag{6.14}$$

$$S_{r,(n+1)}(t) = S_r(0)(t) + \int_0^t \left(\Lambda_r - \frac{\beta_3 S_{r,(n)}(t) I_{r,(n)}(t)}{N_r} - \mu_r S_{r,(n)}(t) \right) dt, \tag{6.15}$$

$$E_{r,(n+1)}(t) = E_r(0)(t) + \int_0^t \left(\frac{\beta_3 S_{r,(n)}(t) I_{r,(n)}(t)}{N_r} - (\mu_r + \alpha_3) E_{r,(n)}(t) \right) dt, \tag{6.16}$$

$$I_{r,(n+1)}(t) = I_r(0)(t) + \int_0^t \left(\alpha_3 E_{r,(n)}(t) - (\mu_r + \delta_r) I_{r,(n)}(t) \right) dt. \tag{6.17}$$

By substituting the parameters and initial values defined in Equation (6.9) into the system of equations (6.10)–(6.17), the following expressions are derived.

For $n = 0$:

$$S_{h,1}(t) = 10000 + 0.1409560350t, \tag{6.18}$$

$$E_{h,1}(t) = 50 - 110.5367256t, \tag{6.19}$$

$$I_{h,1}(t) = 0.57 + 21.14733838t, \tag{6.20}$$

$$Q_{h,1}(t) = 0.43 + 88.76493490t, \tag{6.21}$$

$$R_{h,1}(t) = 0.4684808200t, \tag{6.22}$$

$$S_{r,1}(t) = 1000 - 2.832536016t, \tag{6.23}$$

$$E_{r,1}(t) = 100 + 0.3091162900t, \tag{6.24}$$

$$I_{r,1}(t) = 10 + 2.523170548t. \tag{6.25}$$

The iterative process was continued up to $n = 6$, at which point convergence was achieved when the tolerance criterion $\varepsilon < 10^{-5}$ was satisfied. Due to space limitations and the extensive nature of the iterative solutions, only the first two iterations ($n = 0, 1$) and last iteration ($n = 6$) are presented in this section, while the complete series are provided in the Appendix F.

MPIM:

In contrast to the standard PIM, the MPIM was employed to enhance convergence stability and accuracy over larger computational intervals. The MPIM introduces a time-marching correction mechanism by dividing the main interval into smaller subintervals of fixed step size $h = 0.01$, where each subinterval serves as the initial condition for the subsequent stage. This multistage formulation effectively reduces local truncation errors and ensures global convergence throughout the solution domain.

According to MPIM formulation in (3.31), construct an iterative scheme for system (6.1)-(6.8) as follows:

$$S_{h,(n+1),j}(t) = S_{h,(n),j-1}(t_{j-1}) + \int_{t_{j-1}}^{t} \Bigg(\Lambda_h - \frac{\beta_1 I_{r,(n)}(t) + \beta_2 I_{h,(n)}(t)}{N_h} S_{h,(n)}(t) - \mu_h S_{h,(n)}(t) + \phi Q_{h,(n)}(t) \Bigg) dt, \tag{6.26}$$

$$E_{h,(n+1),j}(t) = E_{h,(n),j-1}(t_{j-1}) + \int_{t_{j-1}}^{t} \Bigg(\frac{\beta_1 I_{r,(n)}(t) + \beta_2 I_{h,(n)}(t)}{N_h} S_{h,(n)}(t) - (\alpha_1 + \alpha_2 + \mu_h) E_{h,(n)}(t) \Bigg) dt, \tag{6.27}$$

$$I_{h,(n+1),j}(t) = I_{h,(n),j-1}(t_{j-1}) + \int_{t_{j-1}}^{t} \left(\alpha_1 E_{h,(n)}(t) - (\mu_h + \delta_h + \rho) I_{h,(n)}(t) \right) dt, \tag{6.28}$$

$$Q_{h,(n+1),j}(t) = Q_{h,(n),j-1}(t_{j-1}) + \int_{t_{j-1}}^{t} \left(\alpha_2 E_{h,(n)}(t) - (\phi + \tau + \mu_h + \delta_h) Q_{h,(n)}(t) \right) dt, \tag{6.29}$$

$$R_{h,(n+1),j}(t) = R_{h,(n),j-1}(t_{j-1}) + \int_{t_{j-1}}^{t} \left(\rho I_{h,(n)}(t) + \tau Q_{h,(n)}(t) - \mu_h R_{h,(n)}(t) \right) dt, \tag{6.30}$$

$$S_{r,(n+1),j}(t) = S_{r,(n),j-1}(t_{j-1}) + \int_{t_{j-1}}^{t} \left(\Lambda_r - \frac{\beta_3 S_{r,(n)}(t) I_{r,(n)}(t)}{N_r} - \mu_r S_{r,(n)}(t) \right) dt, \tag{6.31}$$

$$E_{r,(n+1),j}(t) = E_{r,(n),j-1}(t_{j-1}) + \int_{t_{j-1}}^{t} \left(\frac{\beta_3 S_{r,(n)}(t) I_{r,(n)}(t)}{N_r} - (\mu_r + \alpha_3) E_{r,(n)}(t) \right) dt, \tag{6.32}$$

$$I_{r,(n+1),j}(t) = I_{r,(n),j-1}(t_{j-1}) + \int_{t_{j-1}}^{t} \left(\alpha_3 E_{r,(n)}(t) - (\mu_r + \delta_r) I_{r,(n)}(t) \right) dt. \tag{6.33}$$

By substituting the parameters and initial values defined in Equation (6.9) into the system of equations (6.26)–(6.33), the following expressions are derived.

For $n = 3, j = 1$:

$$S_{h,4,1}(t) = 10000 + 0.14096t + 62.046t^2 - 106.52t^3 + 100.14t^4 + 0.055953t^5 - 0.046785t^6 + 0.021636t^7 + 9.4058 \times 10^{-6}t^8 - 3.5440 \times 10^{-6}t^9 + 1.8742 \times 10^{-12}t^{10} + 1.1716 \times 10^{-18}t^{11}, \quad (6.34)$$

$$E_{h,4,1}(t) = 50 - 110.54t + 130.65t^2 - 102.77t^3 + 60.656t^4 - 0.066753t^5 + 0.055087t^6 - 0.021636t^7 - 9.4058 \times 10^{-6}t^8 + 3.5440 \times 10^{-6}t^9 - 1.8742 \times 10^{-12}t^{10} - 1.1716 \times 10^{-18}t^{11}, \quad (6.35)$$

$$I_{h,4,1}(t) = 0.57 + 21.147t - 24.303t^2 + 19.131t^3 - 11.287t^4 + 0.0020625t^5 - 0.0015842t^6 + 1.3498 \times 10^{-9}t^7, \quad (6.36)$$

$$Q_{h,4,1}(t)) = 0.43000 + 88.765t - 214.31t^2 + 263.33t^3 - 216.71t^4 + 0.0087297t^5 - 0.0067179t^6 + 5.7240 \times 10^{-9}t^7, \quad (6.37)$$

$$R_{h,4,1}(t) = 0.46848t + 45.088t^2 - 71.969t^3 + 66.141t^4 + 6.7569 \times 10^{-6}t^5, \quad (6.38)$$

$$S_{r,4,1}(t) = 1000 - 2.8325t - 0.36008t^2 + 0.00079495t^3 - 0.00015184t^4 + 6.5004 \times 10^{-7}t^5 + 4.3516 \times 10^{-8}t^6 - 1.6868 \times 10^{-10}t^7 - 7.0136 \times 10^{-12}t^8 + 1.2531 \times 10^{-14}t^9 + 4.8948 \times 10^{-18}t^{10} - 1.2129 \times 10^{-20}t^{11}, \quad (6.39)$$

$$E_{r,4,1}(t) = 100 + 0.30912t + 0.35686t^2 - 0.0038026t^3 + 0.00017629t^4 - 3.3992 \times 10^{-7}t^5 - 4.3661 \times 10^{-8}t^6 + 1.6830 \times 10^{-10}t^7 + 7.0136 \times 10^{-12}t^8 - 1.2531 \times 10^{-14}t^9 - 4.8948 \times 10^{-18}t^{10} + 1.2129 \times 10^{-20}t^{11}, \quad (6.40)$$

$$I_{r,4,1}(t) = 10 + 2.5232t + 0.0031858t^2 + 0.0030076t^3 - 0.000024472t^4 - 3.1009 \times 10^{-7}t^5 + 1.4430 \times 10^{-10}t^6 + 3.8242 \times 10^{-13}t^7. \quad (6.41)$$

The iterative process was continued up to $n = 3, j = 12000$. Due to space limitations and the extensive nature of the iterative solutions, the first iterations ($n = 3, j = 1$) is presented in this section, while the complete series are provided in Appendix F.

6.2.2 Results and discussion

Figures 6.1 - 6.8 present the numerical solutions for S_h, E_h, I_h, Q_h, R_h, S_r, E_r, and I_r, respectively, obtained using the 7-iteration PIM, the 4-iteration MPIM ($\Delta t = 0.01$), 7-iteration ADM, the 4-iteration MADM ($\Delta t = 0.01$),as well as the fourth-order and eighth-order Runge-Kutta methods, denoted as RK4 and RK8, respectively, using the same time step. The simulations were performed using $\alpha_1 = 0.423890$, $\alpha_2 = 1.797575$, $\alpha_3 = 0.025289$, $\beta_1 = 0.011503$, $\beta_2 = 0.747322$, $\beta_3 = 0.321102$, $\Lambda_h = 0.34857$, $\Lambda_r = 0.60822$, $\phi = 1.575454$, $\rho = 0.067689$, $\mu_h = \frac{1}{79(365)}$, $\mu_r = \frac{1}{5(365)}$, $\delta_h = 0.015016$, $\delta_r = 0.000025$, and $\tau = 0.999763$, where the parameter values were adopted from Kumar et al. (2023). Since the MPIM and MADM involves repeated iterative computations across multiple subintervals, RK8 is included as an additional higher-order numerical reference for comparison, while RK4 is retained to facilitate comparison with the numerical approach adopted in related Monkeypox modelling studies.

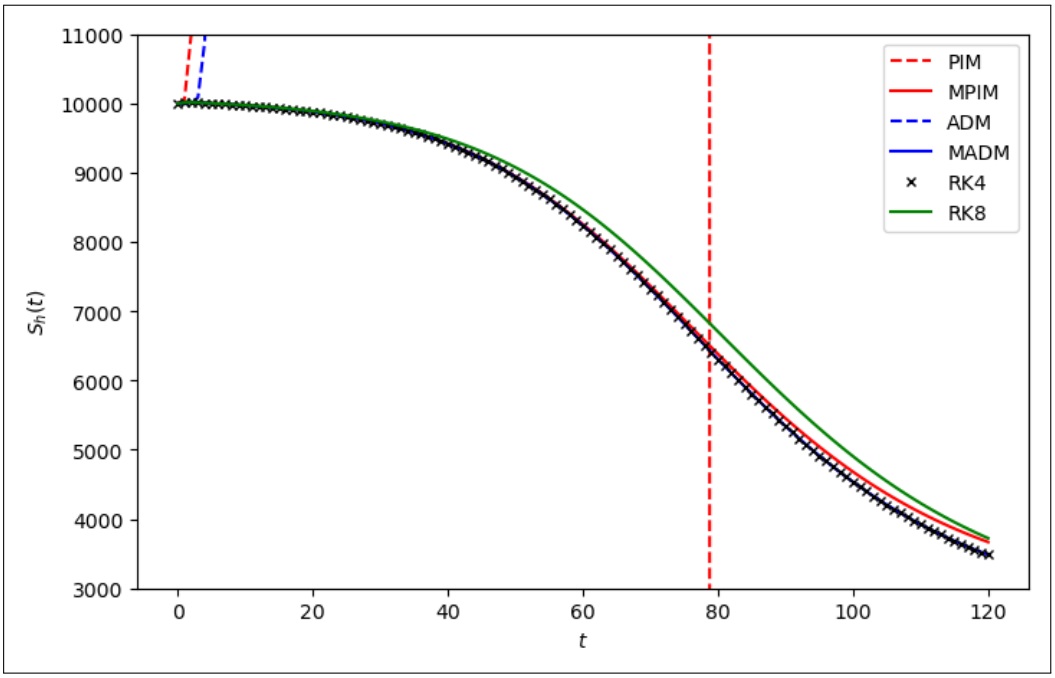

Figure 6.1 Solution of Susceptible Human S_h using 7-iteration PIM, 4-iteration MPIM on the Time Step ($\Delta t = 0.01$), 7-iteration ADM, 7-iteration MADM on the Time Step ($\Delta t = 0.01$), RK4 on the Time Step ($\Delta t = 0.01$) and RK8 on the Time Step ($\Delta t = 0.01$).

Figure 6.1 presents the numerical solutions of the susceptible human population, $S_h(t)$, obtained using the 7-iteration PIM, 4-iteration MPIM with

$\Delta t = 0.01$, 7-iteration ADM, 7-iteration MADM with $\Delta t = 0.01$, RK4 with $\Delta t = 0.01$, and RK8 with $\Delta t = 0.01$. All stable methods show a consistent decreasing pattern of $S_h(t)$ from the initial value of 10000, where the reduction becomes more apparent after approximately $t = 40$ and continues until the end of the interval. However, the classical PIM and ADM fail to preserve this behaviour over the full interval, with the PIM solution diverging at approximately $t = 2$ and the ADM solution diverging at approximately $t = 4$. In contrast, both MPIM and MADM remain stable throughout $0 \leq t \leq 120$, demonstrating the effectiveness of the multistage procedure in controlling the error accumulation observed in the corresponding classical methods. The MADM solution is nearly superimposed with the RK4 solution, while the MPIM solution follows the same decreasing trend but gives slightly higher values than RK4, particularly for larger values of t. The RK8 solution also exhibits the same overall behaviour but remains slightly above the RK4, MPIM, and MADM solutions towards the end of the simulation interval. Overall, the figure shows that the multistage methods provide stable and reliable approximations for $S_h(t)$ over a long computational interval, unlike the standard PIM and ADM methods.

Table 6.1
Comparison of Absolute Errors for Susceptible Human, $S_h(t)$ Between RK4 on the Time Step ($\Delta t = 0.01$), 7-iteration PIM, 4-iteration MPIM on the Time Step ($\Delta t = 0.01$), the 7-iteration of ADM and the 4-iteration of MADM on the Time Step ($\Delta t = 0.01$).

Time	|RK4-PIM|	|RK4-ADM|	|RK4-MPIM|	|RK4-MADM|
10	1.0736×10^{6}	7.9226×10^{5}	1.0603×10^{-2}	$< 1.0 \times 10^{-16}$
20	3.8499×10^{7}	1.1746×10^{8}	9.5088×10^{-2}	1.0000×10^{-6}
30	4.5513×10^{8}	2.1163×10^{9}	4.9546×10^{-1}	$< 1.0 \times 10^{-16}$
40	2.7411×10^{9}	1.6294×10^{10}	1.9938×10^{0}	1.0000×10^{-6}
50	1.0248×10^{10}	7.9002×10^{10}	6.6383×10^{0}	2.0000×10^{-6}
60	2.5567×10^{10}	2.8600×10^{11}	1.8343×10^{1}	2.0000×10^{-6}
70	3.7549×10^{10}	8.4900×10^{11}	4.1345×10^{1}	3.0000×10^{-6}
80	1.4174×10^{10}	2.1800×10^{12}	7.5392×10^{1}	3.0000×10^{-6}
90	3.1600×10^{11}	4.9800×10^{12}	1.1324×10^{2}	3.0000×10^{-6}
100	1.3200×10^{12}	1.0500×10^{13}	1.4594×10^{2}	2.0000×10^{-6}
110	4.0100×10^{12}	2.0500×10^{13}	1.6910×10^{2}	2.0000×10^{-6}
120	1.0200×10^{13}	3.7700×10^{13}	1.8325×10^{2}	1.0000×10^{-6}

Table 6.1 compares the absolute errors of PIM, ADM, MPIM, and MADM using RK4 as the numerical reference solution for the susceptible human compartment,

$S_h(t)$. The PIM and ADM produce very large errors throughout the interval, where the PIM error increases from 1.0736×10^6 at $t = 10$ to 1.0200×10^{13} at $t = 120$, while the ADM error increases from 7.9226×10^5 to 3.7700×10^{13}. These large errors are consistent with the divergence observed in the corresponding graphical results and indicate that the classical PIM and ADM are not suitable for long-term simulation of $S_h(t)$ under the selected parameter setting. In contrast, MPIM produces substantially smaller errors, ranging from 1.0603×10^{-2} at $t = 10$ to 1.8325×10^2 at $t = 120$, which confirms that the multistage implementation significantly reduces the error accumulation of the standard PIM. The MADM provides the smallest errors among all methods, with values between less than 1.0×10^{-16} and 3.0×10^{-6}, indicating an almost identical agreement with the RK4 solution within the numerical precision used in the computation. Therefore, based on RK4 as the reference, MADM gives the closest approximation for $S_h(t)$, followed by MPIM.

Table 6.2
Comparison of Absolute Errors for Susceptible Human, $S_h(t)$ Between RK8 on the Time Step ($\Delta t = 0.01$), 7-iteration PIM, 4-iteration MPIM on the Time Step ($\Delta t = 0.01$), the 7-iteration of ADM and the 4-iteration of MADM on the Time Step ($\Delta t = 0.01$).

Time	|RK8-PIM|	|RK8-MPIM|	|RK8-ADM|	|RK8-MADM|
10	1.0736×10^6	3.2874×10^0	7.9226×10^5	3.2980×10^0
20	3.8499×10^7	1.1705×10^1	1.1746×10^8	1.1800×10^1
30	4.5513×10^8	3.0531×10^1	2.1163×10^9	3.1027×10^1
40	2.7411×10^9	6.7572×10^1	1.6294×10^{10}	6.9566×10^1
50	1.0248×10^{10}	1.2957×10^2	7.9002×10^{10}	1.3621×10^2
60	2.5567×10^{10}	2.1212×10^2	2.8628×10^{11}	2.3046×10^2
70	3.7549×10^{10}	2.8941×10^2	8.4903×10^{11}	3.3076×10^2
80	1.4174×10^{10}	3.2351×10^2	2.1753×10^{12}	3.9890×10^2
90	3.1570×10^{11}	2.9605×10^2	4.9848×10^{12}	4.0930×10^2
100	1.3226×10^{12}	2.2323×10^2	1.0462×10^{13}	3.6917×10^2
110	4.0059×10^{12}	1.3565×10^2	2.0451×10^{13}	3.0475×10^2
120	1.0246×10^{13}	5.5346×10^1	3.7703×10^{13}	2.3859×10^2

Table 6.2 presents the absolute errors when RK8 is used as the numerical reference solution. Similar to the RK4-based comparison, PIM and ADM exhibit extremely large errors, reaching 1.0246×10^{13} and 3.7703×10^{13}, respectively, at $t = 120$, further confirming the instability of the classical iterative methods over the long computational interval. The MPIM and MADM remain considerably closer to

RK8 than PIM and ADM; however, their errors are larger than those obtained when RK4 is used as the reference. The MPIM error increases from 3.2874 at $t = 10$ to a maximum of 3.2351×10^2 at $t = 80$, before decreasing to 5.5346×10^1 at $t = 120$, whereas the MADM error increases from 3.2980 at $t = 10$ to 4.0930×10^2 at $t = 90$ and decreases to 2.3859×10^2 at $t = 120$. This difference indicates that the RK4 and RK8 solutions are not fully identical over the selected interval, particularly at larger values of t. Nevertheless, both tables consistently show that MPIM and MADM are substantially more stable and accurate than the corresponding classical PIM and ADM methods for approximating the susceptible human population.

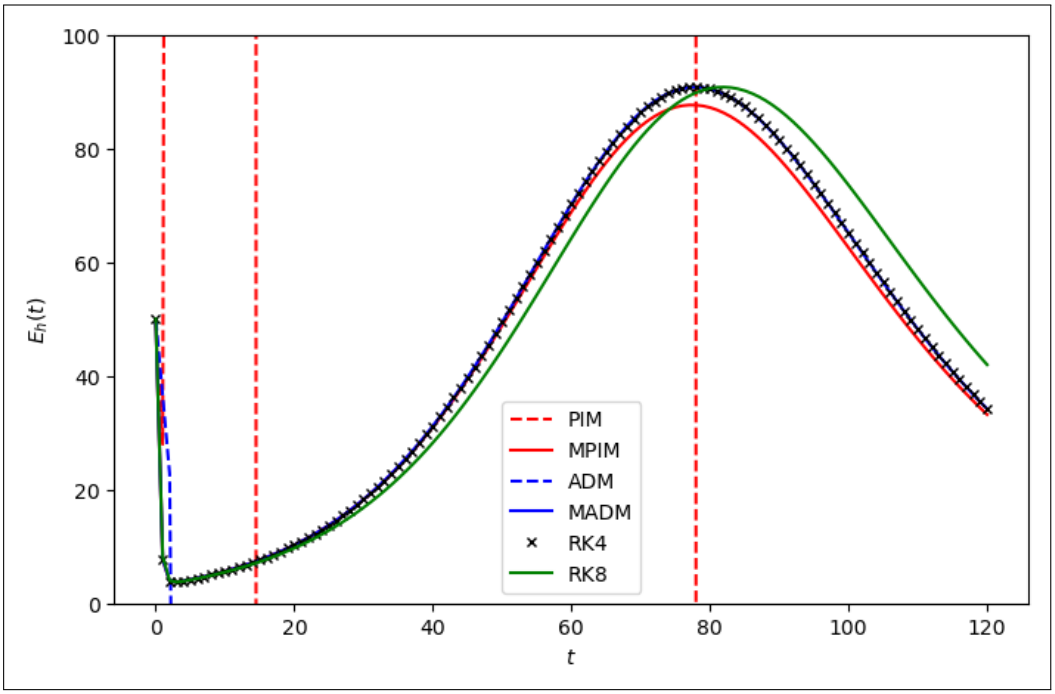

Figure 6.2 Solution of Exposed Human, E_h using 7-iteration PIM, 4-iteration MPIM on the Time Step ($\Delta t = 0.01$), 7-iteration ADM, 7-iteration MADM on the Time Step ($\Delta t = 0.01$), RK4 on the Time Step ($\Delta t = 0.01$) and RK8 on the Time Step ($\Delta t = 0.01$).

Figure 6.2 illustrates the numerical behaviour of the exposed human population, $E_h(t)$, computed using the 7-iteration of PIM, the 4-iteration of MPIM ($\Delta t = 0.01$), the 7-iteration of ADM, the 4-iteration of MADM ($\Delta t = 0.01$), and RK4 ($\Delta t = 0.01$) over the interval $0 \leq t \leq 120$. Based on the figure, the RK4 solution shows that the exposed human population initially decreases to a low value, then increases gradually until reaching a peak around $t = 80$, before decreasing again towards the end of the simulation. The classical PIM and ADM solutions do not

follow this overall pattern over the full interval, as both methods show noticeable deviation from the RK4 reference solution at the early stage of the simulation. This indicates that the non-multistage forms of PIM and ADM are less consistent for capturing the long-time behaviour of $E_h(t)$ in this nonlinear Monkeypox transmission model. In contrast, the multistage methods show better agreement with the RK4 solution. The MPIM solution is able to reproduce the general rise-and-fall pattern of the exposed human population, although a slight difference can be observed near the peak region and during the decreasing phase after approximately $t = 80$. Meanwhile, the MADM solution appears to follow the RK4 profile more closely throughout the interval, including around the peak and the later decreasing stage. Therefore, the graphical results suggest that the multistage implementation improves the numerical behaviour of both PIM and ADM by updating the approximation at each subinterval.

Table 6.3
Comparison of Absolute Errors for the Exposed Human Compartment Between RK4 with Time Step ($\Delta t = 0.01$), 7-iteration PIM, 4-iteration MPIM with Time Step ($\Delta t = 0.01$), 7-iteration ADM, and 4-iteration MADM with Time Step ($\Delta t = 0.01$).

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
10	3.5045×10^{5}	2.0191×10^{-3}	1.0075×10^{6}	4.7800×10^{-7}
20	1.3875×10^{7}	1.2913×10^{-2}	1.4933×10^{8}	2.1000×10^{-7}
30	3.2458×10^{8}	5.6401×10^{-2}	2.6903×10^{9}	$< 1.0 \times 10^{-16}$
40	2.3097×10^{9}	1.9955×10^{-1}	2.0712×10^{10}	5.0000×10^{-7}
50	9.1420×10^{9}	5.8369×10^{-1}	1.0042×10^{11}	2.5000×10^{-7}
60	2.3149×10^{10}	1.3679×10^{0}	3.6388×10^{11}	1.5000×10^{-7}
70	3.2801×10^{10}	2.4533×10^{0}	1.0792×10^{12}	5.3000×10^{-7}
80	2.2795×10^{10}	3.2738×10^{0}	2.7649×10^{12}	2.3000×10^{-7}
90	3.3044×10^{11}	3.2891×10^{0}	6.3359×10^{12}	3.5000×10^{-7}
100	1.3466×10^{12}	2.6094×10^{0}	1.3297×10^{13}	1.6000×10^{-7}
110	4.0437×10^{12}	1.7331×10^{0}	2.5994×10^{13}	7.0000×10^{-8}
120	1.0304×10^{13}	1.0072×10^{0}	4.7921×10^{13}	1.5000×10^{-7}

Table 6.3 presents the absolute errors of the numerical methods for the exposed human compartment, $E_h(t)$, by taking the RK4 solution as the numerical reference. The classical PIM and ADM produce very large errors over the whole interval, where the PIM error increases from 3.5045×10^5 at $t = 10$ to 1.0304×10^{13} at $t = 120$, while the ADM error increases from 1.0075×10^6 to 4.7921×10^{13}. This indicates that both classical iterative methods experience substantial error accumulation when applied directly over the long computational interval. In comparison, the MPIM produces

considerably smaller errors, starting from 2.0191×10^{-3} at $t = 10$, increasing to 3.2891 at $t = 90$, and then decreasing to 1.0072 at $t = 120$. This result shows that the multistage modification improves the numerical behaviour of the classical PIM for approximating $E_h(t)$ over the selected interval. The MADM produces errors between less than 1.0×10^{-16} and 5.3000×10^{-7}, indicating very close agreement with the RK4 solution within the computational precision used. Therefore, relative to RK4, the multistage methods remain stable and give substantially smaller errors than their corresponding classical PIM and ADM methods.

Table 6.4
Comparison of Absolute Errors for the Exposed Human Compartment Between RK8 with Time Step ($\Delta t = 0.01$), 7-iteration PIM, 4-iteration MPIM with Time Step ($\Delta t = 0.01$), 7-iteration ADM, and 4-iteration MADM with Time Step ($\Delta t = 0.01$).

Time	|RK8-PIM|	|RK8-MPIM|	|RK8-ADM|	|RK8-MADM|
10	3.5045×10^{5}	1.6188×10^{-1}	1.0075×10^{6}	1.6390×10^{-1}
20	1.3875×10^{7}	5.6259×10^{-1}	1.4933×10^{8}	5.7551×10^{-1}
30	3.2458×10^{8}	1.3903×10^{0}	2.6903×10^{9}	1.4467×10^{0}
40	2.3097×10^{9}	2.7579×10^{0}	2.0712×10^{10}	2.9574×10^{0}
50	9.1420×10^{9}	4.2846×10^{0}	1.0042×10^{11}	4.8683×10^{0}
60	2.3149×10^{10}	4.6147×10^{0}	3.6388×10^{11}	5.9826×10^{0}
70	3.2801×10^{10}	2.0066×10^{0}	1.0792×10^{12}	4.4599×10^{0}
80	2.2795×10^{10}	3.3693×10^{0}	2.7649×10^{12}	9.5424×10^{-2}
90	3.3044×10^{11}	8.5837×10^{0}	6.3359×10^{12}	5.2946×10^{0}
100	1.3466×10^{12}	1.1006×10^{1}	1.3297×10^{13}	8.3969×10^{0}
110	4.0437×10^{12}	1.0603×10^{1}	2.5994×10^{13}	8.8700×10^{0}
120	1.0304×10^{13}	8.7620×10^{0}	4.7921×10^{13}	7.7548×10^{0}

Table 6.4 shows the absolute errors when RK8 is used as the numerical reference solution. The PIM and ADM again give extremely large errors, reaching 1.0304×10^{13} and 4.7921×10^{13}, respectively, at $t = 120$, which is consistent with the instability observed in the RK4-based comparison. In contrast, the MPIM and MADM retain errors of much smaller magnitude throughout the interval. The MPIM error ranges from 1.6188×10^{-1} at $t = 10$ to a maximum of 1.1006×10^{1} at $t = 100$, before reducing to 8.7620 at $t = 120$. Meanwhile, the MADM error ranges from 9.5424×10^{-2} at $t = 80$ to 8.8700 at $t = 110$, with an error of 7.7548 at $t = 120$. Although the relative errors of MPIM and MADM with respect to RK8 vary across the selected time points, both methods remain substantially closer to RK8 than PIM and ADM. Overall, the two comparisons consistently indicate that introducing the

multistage procedure enables MPIM to overcome the large error growth exhibited by the classical PIM, thereby providing a more stable approximation for the exposed human compartment over the considered interval.

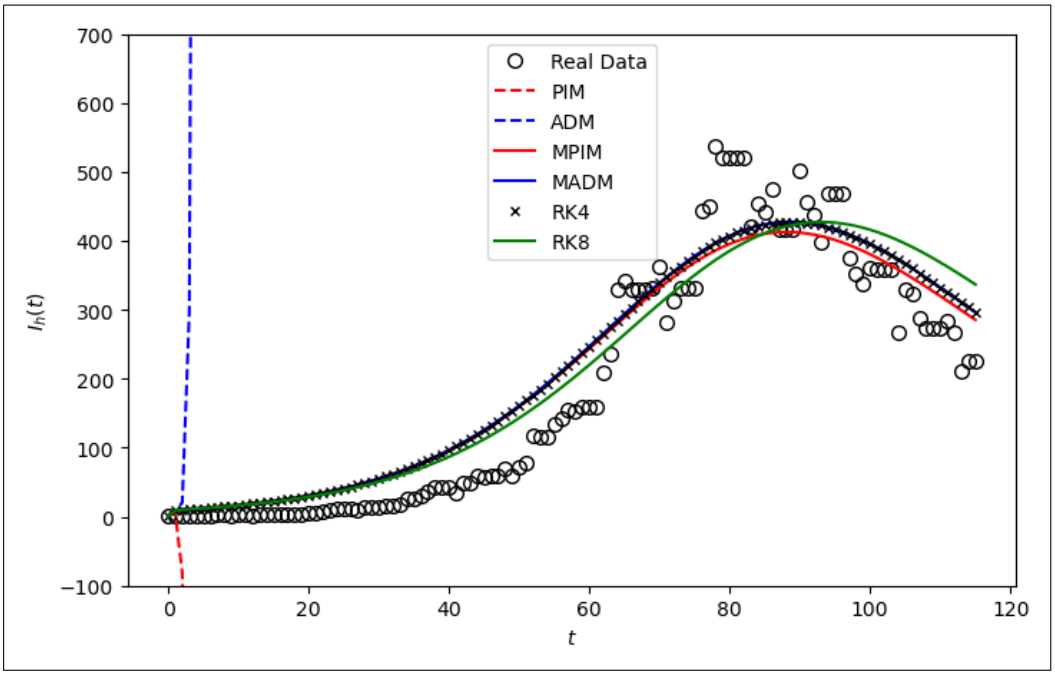

Figure 6.3 Solution of Infected Human, I_h using 7-iteration PIM, 4-iteration MPIM on the Time Step ($\Delta t = 0.01$), 7-iteration ADM, 7-iteration MADM on the Time Step ($\Delta t = 0.01$), RK4 on the Time Step ($\Delta t = 0.01$) and RK8 on the Time Step ($\Delta t = 0.01$) with the Real Data Obtained from Kumar et al. (2023).

The figure compares the infected human population, $I_h(t)$, obtained using the 7-iteration PIM, 4-iteration MPIM, 7-iteration ADM, 7-iteration MADM, RK4, and RK8 methods with the real data reported by Kumar et al. (2023). The PIM and ADM solutions become unstable at the early stage of the simulation and therefore do not provide meaningful approximations over the considered interval. In contrast, the MPIM and MADM solutions remain stable and follow the overall numerical trend produced by the RK4 and RK8 methods. All stable numerical solutions show an increase in the infected human population from the initial time, reaching a peak at approximately $t = 90$ to $t = 95$, followed by a gradual decrease towards the end of the interval. Although the real data exhibit noticeable fluctuations, the MPIM, MADM, RK4, and RK8 solutions capture the general increasing and decreasing pattern of the infected human compartment. The inclusion of RK8 also provides an additional higher-order numerical reference for evaluating the performance of the proposed

multistage approach.

Table 6.5
Comparison of Absolute Errors for the Infected Human Compartment Between the Real Data (RD) from Kumar et al. (2023) and the 7-iteration PIM, 4-iteration MPIM with Time Step ($\Delta t = 0.01$), 7-iteration ADM, 4-iteration MADM with Time Step ($\Delta t = 0.01$), RK4 with Time Step ($\Delta t = 0.01$) and RK8 with Time Step ($\Delta t = 0.01$).

Time	|RD-PIM|	|RD-MPIM|	|RD-ADM|	|RD-MADM|	|RD-RK4|	|RD-RK8|
10	9.7334×10^{4}	1.4519×10^{1}	2.1982×10^{6}	1.4521×10^{1}	1.4521×10^{1}	1.4037×10^{1}
20	1.7569×10^{6}	2.4715×10^{1}	3.2532×10^{8}	2.4736×10^{1}	2.4736×10^{1}	2.3005×10^{1}
30	9.7517×10^{6}	4.0607×10^{1}	5.8572×10^{9}	4.0712×10^{1}	4.0712×10^{1}	3.6218×10^{1}
40	3.3985×10^{7}	5.3303×10^{1}	4.5081×10^{10}	5.3708×10^{1}	5.3708×10^{1}	4.3943×10^{1}
50	9.2317×10^{7}	8.8900×10^{1}	2.1853×10^{11}	9.0202×10^{1}	9.0202×10^{1}	7.2261×10^{1}
60	2.1453×10^{8}	8.5441×10^{1}	7.9178×10^{11}	8.8875×10^{1}	8.8875×10^{1}	6.2014×10^{1}
70	4.4745×10^{8}	3.0916×10^{1}	2.3480×10^{12}	2.3712×10^{1}	2.3712×10^{1}	5.4099×10^{1}
80	8.6115×10^{8}	1.2446×10^{2}	6.0152×10^{12}	1.1274×10^{2}	1.1274×10^{2}	1.3457×10^{2}
90	1.5564×10^{9}	8.9858×10^{1}	1.3784×10^{13}	7.5062×10^{1}	7.5062×10^{1}	7.6828×10^{1}
100	2.6732×10^{9}	2.0918×10^{1}	2.8927×10^{13}	3.5863×10^{1}	3.5863×10^{1}	5.6526×10^{1}
110	4.4003×10^{9}	4.7133×10^{1}	5.6545×10^{13}	5.9747×10^{1}	5.9747×10^{1}	9.6013×10^{1}

Table 6.5 presents the absolute errors between the real data and the numerical solutions for the infected human compartment, $I_h(t)$. The errors produced by the 7-iteration PIM and 7-iteration ADM are substantially larger than those of the other methods, increasing from 9.7334×10^4 and 2.1982×10^6, respectively, at $t = 10$, to 4.4003×10^9 and 5.6545×10^{13}, respectively, at $t = 110$. This indicates that both PIM and ADM are unable to maintain reliable approximations over the considered time interval. In contrast, the MPIM, MADM, RK4, and RK8 methods produce errors within the order of 10^1 to 10^2 throughout the interval. The reported MADM and RK4 errors are identical at all selected time points, whereas the MPIM produces comparable errors and gives smaller errors than MADM and RK4 at $t = 10$ to $t = 60$, as well as at $t = 100$ and $t = 110$. RK8 produces the smallest errors from $t = 10$ to $t = 60$, although it does not give the smallest error at every later time point. Since the comparison is made with real data rather than an exact solution, the reported errors reflect both numerical approximation differences and the discrepancy between the mathematical model and the observed data. Overall, the results show that the multistage approaches, particularly MPIM and MADM, provide substantially more reliable approximations than the standard PIM and ADM for the infected human compartment.

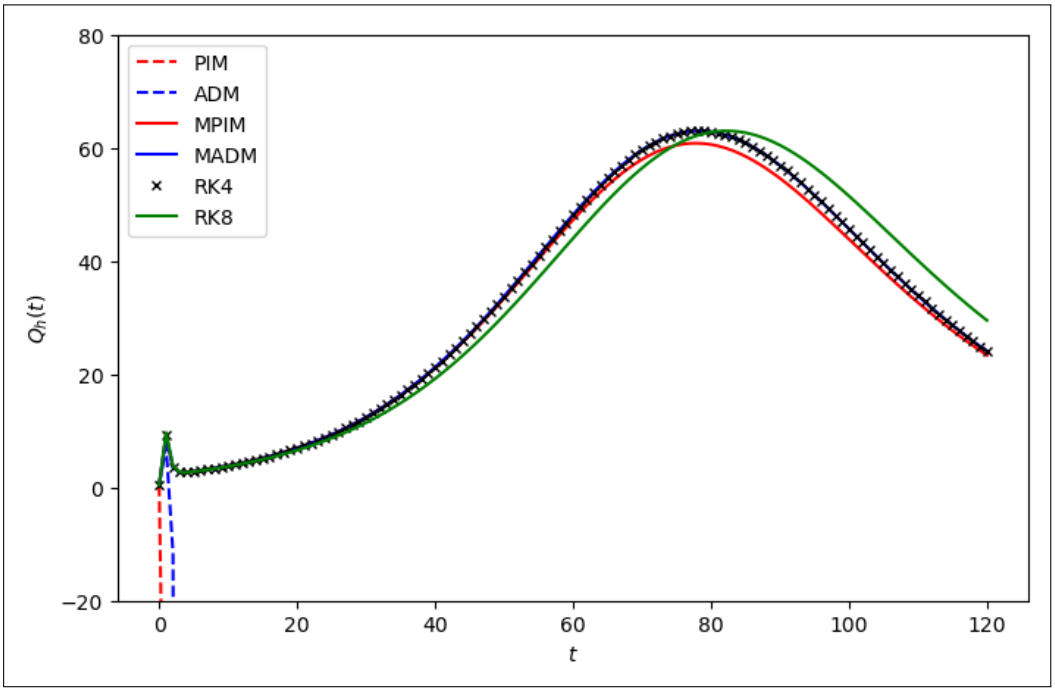

Figure 6.4 Solution of Isolation Human, Q_h using 7-iteration PIM, 4-iteration MPIM on the Time Step ($\Delta t = 0.01$), 7-iteration ADM, 7-iteration MADM on the Time Step ($\Delta t = 0.01$), RK4 on the time step ($\Delta t = 0.01$) and RK8 on the Time Step ($\Delta t = 0.01$).

Figure 6.4 illustrates the numerical solutions for the isolated human compartment, $Q_h(t)$, obtained using the PIM, MPIM, ADM, MADM, RK4, and RK8 methods. The PIM and ADM solutions exhibit instability at the initial stage of the simulation, as indicated by their rapid deviation into negative values near $t = 0$. Hence, these methods do not provide reliable approximations for $Q_h(t)$ over the selected interval. In contrast, the MPIM, MADM, RK4, and RK8 solutions remain stable throughout $0 \leq t \leq 120$ and show a similar overall behaviour. The solutions initially increase slowly, followed by a more rapid increase until reaching their maximum values at approximately $t = 80$, before gradually decreasing towards the end of the interval. The MADM and RK4 solutions are nearly indistinguishable throughout the interval, while the MPIM solution shows a slightly lower peak value. The RK8 solution follows a comparable trend but produces slightly higher values after the peak. These results indicate that the multistage methods, particularly MPIM and MADM, provide stable numerical approximations for the isolated human compartment.

Table 6.6
Comparison of Absolute Errors for the Isolated Human Compartment Between RK4 with Time Step ($\Delta t = 0.01$), 7-iteration PIM, 4-iteration MPIM with Time Step ($\Delta t = 0.01$), 7-iteration ADM, and 4-iteration MADM with Time Step ($\Delta t = 0.01$).

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
10	1.9301×10^{6}	1.2887×10^{-3}	3.8575×10^{6}	2.2855×10^{-7}
20	3.3053×10^{7}	8.4324×10^{-3}	5.6998×10^{8}	2.9236×10^{-7}
30	1.7330×10^{8}	3.7180×10^{-2}	1.0255×10^{10}	3.1600×10^{-7}
40	5.6488×10^{8}	1.3246×10^{-1}	7.8898×10^{10}	6.8137×10^{-8}
50	1.4243×10^{9}	3.9032×10^{-1}	3.8236×10^{11}	1.6400×10^{-7}
60	3.0590×10^{9}	9.2309×10^{-1}	1.3851×10^{12}	6.7526×10^{-7}
70	5.8895×10^{9}	1.6735×10^{0}	4.1070×10^{12}	1.9220×10^{-7}
80	1.0475×10^{10}	2.2587×10^{0}	1.0520×10^{13}	1.1643×10^{-7}
90	1.7546×10^{10}	2.2926×10^{0}	2.4105×10^{13}	2.2666×10^{-7}
100	2.8040×10^{10}	1.8337×10^{0}	5.0585×10^{13}	1.6751×10^{-7}
110	4.3140×10^{10}	1.2253×10^{0}	9.8875×10^{13}	2.3970×10^{-7}
120	6.4325×10^{10}	7.1544×10^{-1}	1.8227×10^{14}	5.4270×10^{-7}

Table 6.6 presents the absolute differences between the numerical solutions for the isolated human compartment, $Q_h(t)$, and the RK4 solution. The PIM and ADM produce substantially large differences, increasing from 1.9301×10^6 and 3.8575×10^6, respectively, at $t = 10$, to 6.4325×10^{10} and 1.8227×10^{14}, respectively, at $t = 120$. In contrast, the MPIM produces differences ranging from 1.2887×10^{-3} to 2.2926×10^0, while the MADM produces considerably smaller differences, remaining within the order of 10^{-8} to 10^{-7} throughout the interval. This indicates that the MADM solution is almost identical to the RK4 solution, whereas the MPIM also remains relatively close to RK4, although its difference increases near the peak of the isolated human population. Overall, the results show that the multistage methods provide substantially closer approximations to the RK4 reference solution than the standard PIM and ADM methods.

Table 6.7
Comparison of Absolute Errors for the Isolated Human Compartment Between RK8 with Time Step ($\Delta t = 0.01$), 7-iteration PIM, 4-iteration MPIM with Time Step ($\Delta t = 0.01$), 7-iteration ADM, and 4-iteration MADM with Time Step ($\Delta t = 0.01$).

Time	|RK8-PIM|	|RK8-MPIM|	|RK8-ADM|	|RK8-MADM|
10	1.9301×10^{6}	1.0991×10^{-1}	3.8575×10^{6}	1.1120×10^{-1}
20	3.3053×10^{7}	3.8250×10^{-1}	5.6998×10^{8}	3.9093×10^{-1}
30	1.7330×10^{8}	9.4795×10^{-1}	1.0255×10^{10}	9.8513×10^{-1}
40	5.6488×10^{8}	1.8903×10^{0}	7.8898×10^{10}	2.0228×10^{0}
50	1.4243×10^{9}	2.9641×10^{0}	3.8236×10^{11}	3.3545×10^{0}
60	3.0590×10^{9}	3.2513×10^{0}	1.3851×10^{12}	4.1744×10^{0}
70	5.8895×10^{9}	1.5249×10^{0}	4.1070×10^{12}	3.1984×10^{0}
80	1.0475×10^{10}	2.1753×10^{0}	1.0520×10^{13}	8.3351×10^{-2}
90	1.7546×10^{10}	5.8573×10^{0}	2.4105×10^{13}	3.5646×10^{0}
100	2.8040×10^{10}	7.6343×10^{0}	5.0585×10^{13}	5.8006×10^{0}
110	4.3140×10^{10}	7.4161×10^{0}	9.8875×10^{13}	6.1908×10^{0}
120	6.4325×10^{10}	6.1561×10^{0}	1.8227×10^{14}	5.4406×10^{0}

Table 6.7 presents the absolute differences between the numerical solutions for the isolated human compartment, $Q_h(t)$, and the RK8 solution. Similar to the comparison with RK4, the PIM and ADM produce very large differences, ranging from the order of 10^6 at $t = 10$ to 10^{10} and 10^{14}, respectively, at $t = 120$. In contrast, both MPIM and MADM produce much smaller differences, within the order of 10^{-1} to 10^0 over the selected interval. The MPIM gives slightly smaller differences than MADM from $t = 10$ to $t = 70$, whereas MADM produces smaller differences from $t = 80$ to $t = 120$. These results indicate that both multistage methods remain close to the higher-order RK8 reference solution, while the standard PIM and ADM solutions deviate substantially as time increases.

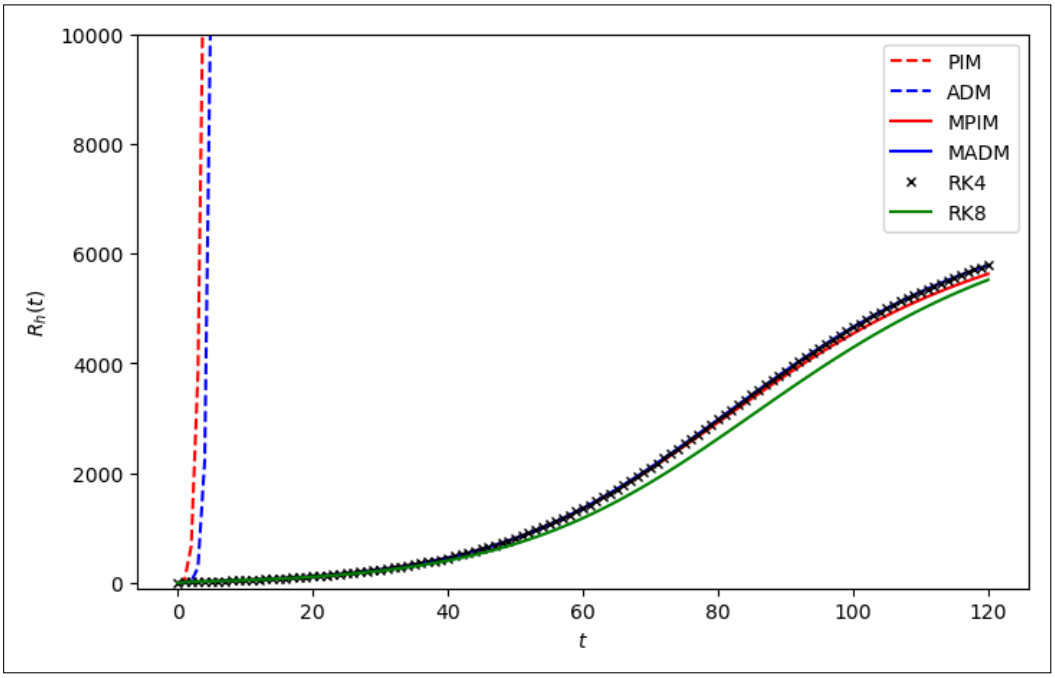

Figure 6.5 Solution of Recovered Human, R_h using 7-iteration PIM, 4-iteration MPIM on the Time Step ($\Delta t = 0.01$), 7-iteration ADM, 7-iteration MADM ($\Delta t = 0.01$), RK4 on the Time Step ($\Delta t = 0.01$) and RK8 on the Time Step ($\Delta t = 0.01$).

Figure 6.5 illustrates the numerical solutions for the recovered human compartment, $R_h(t)$, obtained using the PIM, MPIM, ADM, MADM, RK4, and RK8 methods. The PIM and ADM solutions become unstable at the early stage of the simulation, as shown by their rapid divergence to excessively large values near the beginning of the interval, and therefore do not provide reliable approximations for $R_h(t)$. In contrast, the MPIM, MADM, RK4, and RK8 solutions remain stable throughout $0 \leq t \leq 120$ and exhibit a similar increasing trend over time. The recovered human population increases gradually at the initial stage, followed by a more rapid growth after approximately $t = 60$, and continues to rise until the end of the interval. The MADM and RK4 solutions are almost indistinguishable, while the MPIM solution remains very close to them with only slight deviation. The RK8 solution follows the same overall behaviour but gives slightly lower values, particularly at the later stage of the interval. These results show that the multistage methods, especially MPIM and MADM, provide stable and reliable numerical approximations for the recovered human compartment.

Table 6.8
Comparison of Absolute Errors for the Recovered Human compartment between RK4 with Time Step ($\Delta t = 0.01$), 7-iteration PIM, 4-iteration MPIM with Time Step ($\Delta t = 0.01$), 7-iteration ADM, and 4-iteration MADM with Time Step ($\Delta t = 0.01$).

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
10	5.9390×10^{5}	4.7477×10^{-3}	1.8910×10^{6}	2.5694×10^{-7}
20	1.0025×10^{7}	5.0751×10^{-2}	2.7896×10^{8}	1.3744×10^{-8}
30	5.1672×10^{7}	2.8441×10^{-1}	5.0155×10^{9}	4.4055×10^{-8}
40	1.6479×10^{8}	1.1981×10^{0}	3.8571×10^{10}	9.3752×10^{-7}
50	4.0450×10^{8}	4.1502×10^{0}	1.8688×10^{11}	1.2004×10^{-6}
60	8.4181×10^{8}	1.1976×10^{1}	6.7685×10^{11}	1.9241×10^{-6}
70	1.5636×10^{9}	2.8398×10^{1}	2.0067×10^{12}	2.7066×10^{-6}
80	2.6726×10^{9}	5.4803×10^{1}	5.1397×10^{12}	2.5727×10^{-6}
90	4.2874×10^{9}	8.7157×10^{1}	1.1776×10^{13}	2.2887×10^{-6}
100	6.5426×10^{9}	1.1826×10^{2}	2.4709×10^{13}	2.1807×10^{-6}
110	9.5886×10^{9}	1.4292×10^{2}	4.8296×10^{13}	1.6965×10^{-6}
120	1.3591×10^{10}	1.5986×10^{2}	8.9024×10^{13}	6.9645×10^{-7}

Table 6.8 presents the absolute differences between the numerical solutions for the recovered human compartment, $R_h(t)$, and the RK4 solution. The PIM and ADM produce substantially large differences, increasing from 5.9390×10^5 and 1.8910×10^6, respectively, at $t = 10$, to 1.3591×10^{10} and 8.9024×10^{13}, respectively, at $t = 120$. In contrast, the MPIM produces differences ranging from 4.7477×10^{-3} to 1.5986×10^2, while the MADM produces much smaller differences within the order of 10^{-8} to 10^{-6} throughout the interval. This indicates that the MADM solution is almost identical to the RK4 solution. Although the difference between the MPIM and RK4 solutions increases as time progresses, the MPIM remains substantially closer to RK4 than the standard PIM and ADM methods. Overall, the results demonstrate that the multistage methods provide more reliable approximations to the RK4 reference solution for the recovered human compartment.

Table 6.9
Comparison of Absolute Errors for the Recovered Human Compartment Between RK8 with Time Step ($\Delta t = 0.01$), 7-iteration PIM, 4-iteration MPIM with Time Step ($\Delta t = 0.01$), 7-iteration ADM, and 4-iteration MADM with Time Step ($\Delta t = 0.01$).

Time	|RK8-PIM|	|RK8-MPIM|	|RK8-ADM|	|RK8-MADM|
10	5.9391×10^{5}	2.3853×10^{0}	1.8910×10^{6}	2.3900×10^{0}
20	1.0025×10^{7}	8.5179×10^{0}	2.7896×10^{8}	8.5687×10^{0}
30	5.1672×10^{7}	2.2394×10^{1}	5.0155×10^{9}	2.2678×10^{1}
40	1.6479×10^{8}	5.0344×10^{1}	3.8571×10^{10}	5.1542×10^{1}
50	4.0450×10^{8}	9.9155×10^{1}	1.8688×10^{11}	1.0330×10^{2}
60	8.4181×10^{8}	1.6918×10^{2}	6.7685×10^{11}	1.8116×10^{2}
70	1.5636×10^{9}	2.4473×10^{2}	2.0067×10^{12}	2.7312×10^{2}
80	2.6726×10^{9}	2.9535×10^{2}	5.1397×10^{12}	3.5015×10^{2}
90	4.2874×10^{9}	2.9725×10^{2}	1.1776×10^{13}	3.8441×10^{2}
100	6.5426×10^{9}	2.5247×10^{2}	2.4709×10^{13}	3.7073×10^{2}
110	9.5886×10^{9}	1.8195×10^{2}	4.8296×10^{13}	3.2487×10^{2}
120	1.3591×10^{10}	1.0706×10^{2}	8.9024×10^{13}	2.6692×10^{2}

Table 6.9 presents the absolute differences between the numerical solutions for the recovered human compartment, $R_h(t)$, and the RK8 solution. Similar to the comparison with RK4, the PIM and ADM produce very large differences, ranging from the order of 10^5 and 10^6, respectively, at $t = 10$, to 10^{10} and 10^{13}, respectively, at $t = 120$. In comparison, the MPIM and MADM produce considerably smaller differences, ranging from the order of 10^0 to 10^2 throughout the selected interval. The MPIM produces smaller differences than MADM at all reported time points, indicating that its solution is closer to the RK8 reference solution for this compartment. Nevertheless, both multistage methods remain substantially closer to RK8 than the standard PIM and ADM methods, which deviate considerably as time increases.

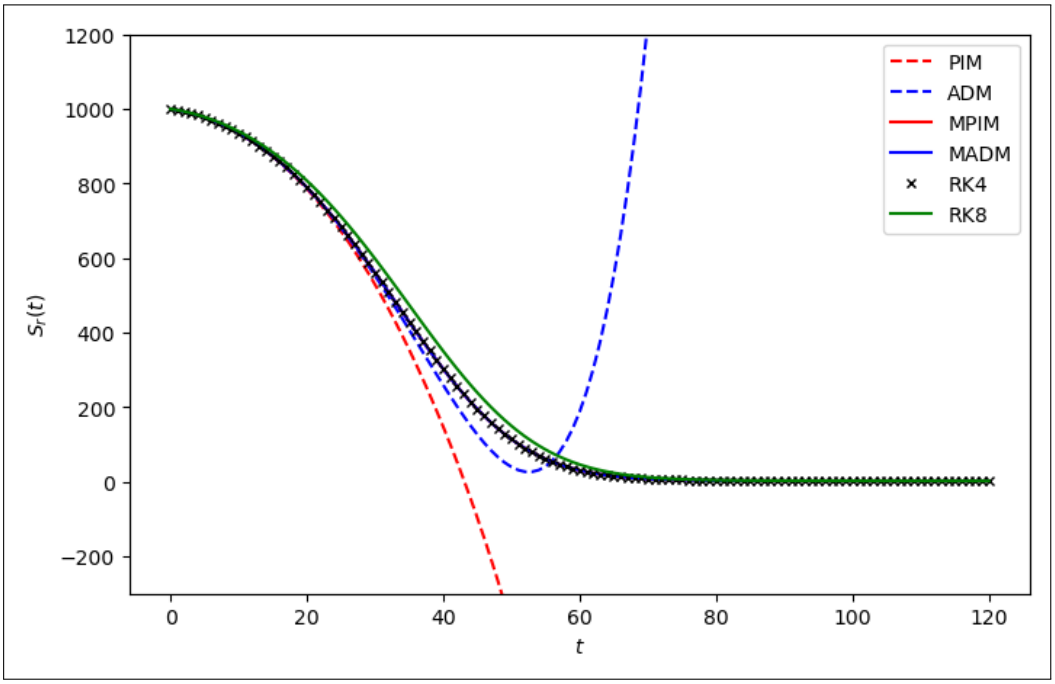

Figure 6.6 Solution of Susceptible Rodent, S_r using 7-iteration PIM, 4-iteration MPIM on the Time Step ($\Delta t = 0.01$), 7-iteration ADM, 7-iteration MADM on the Time Step($\Delta t = 0.01$), RK4 on the Time Step ($\Delta t = 0.01$) and RK8 on the Time Step ($\Delta t = 0.01$).

Figure 6.6 illustrates the numerical solutions for the susceptible rodent compartment, $S_r(t)$, obtained using the PIM, MPIM, ADM, MADM, RK4, and RK8 methods. The PIM solution decreases rapidly and becomes negative at approximately $t = 43$, indicating that it does not provide a physically meaningful approximation over the full interval. Similarly, the ADM solution initially decreases but subsequently increases rapidly after approximately $t = 50$, resulting in an unstable solution. In contrast, the MPIM, MADM, RK4, and RK8 solutions remain stable and exhibit a consistent decreasing trend throughout $0 \leq t \leq 120$. The susceptible rodent population decreases gradually from its initial value and approaches zero as time increases. The MPIM and MADM solutions closely follow the RK4 solution, while the RK8 solution shows a similar overall behaviour with only slight deviation from the other stable numerical solutions. These results indicate that the multistage methods provide stable and reliable approximations for the susceptible rodent compartment over the selected time interval.

Table 6.10
Comparison of Absolute Errors for the Susceptible Rodent Compartment Between RK4 with Time Step ($\Delta t = 0.01$), 7-iteration PIM, 4-iteration MPIM with Time Step ($\Delta t = 0.01$), 7-iteration ADM, and 4-iteration MADM with Time Step ($\Delta t = 0.01$).

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
10	1.1468×10^{-1}	1.7217×10^{-4}	2.0669×10^{-3}	1.3625×10^{-7}
20	3.6525×10^{0}	1.9020×10^{-3}	4.4298×10^{-1}	6.1148×10^{-8}
30	3.1800×10^{1}	8.4433×10^{-3}	7.9773×10^{0}	2.0997×10^{-7}
40	1.5519×10^{2}	1.9716×10^{-2}	4.3257×10^{1}	2.6594×10^{-7}
50	5.0321×10^{2}	2.4097×10^{-2}	7.4300×10^{1}	1.7154×10^{-7}
60	1.2787×10^{3}	1.5902×10^{-2}	1.5717×10^{2}	1.0770×10^{-7}
70	3.0466×10^{3}	6.7371×10^{-3}	1.2228×10^{3}	3.3755×10^{-7}
80	7.3895×10^{3}	2.8533×10^{-3}	3.9979×10^{3}	1.2830×10^{-7}
90	1.7864×10^{4}	2.0727×10^{-3}	9.7113×10^{3}	9.4517×10^{-8}
100	4.1363×10^{4}	2.1916×10^{-3}	2.0033×10^{4}	4.7796×10^{-7}
110	9.0025×10^{4}	2.4959×10^{-3}	3.7149×10^{4}	4.6171×10^{-7}
120	1.8375×10^{5}	2.8421×10^{-3}	6.3826×10^{4}	1.9961×10^{-7}

Table 6.10 presents the absolute differences between the numerical solutions for the susceptible rodent compartment, $S_r(t)$, and the RK4 solution. The differences produced by the PIM and ADM increase substantially over time, from 1.1468×10^{-1} and 2.0669×10^{-3}, respectively, at $t = 10$, to 1.8375×10^{5} and 6.3826×10^{4}, respectively, at $t = 120$. In contrast, the MPIM produces very small differences, ranging from 1.7217×10^{-4} to 2.8421×10^{-3}, while the MADM produces the smallest differences, remaining within the order of 10^{-8} to 10^{-7} throughout the interval. This indicates that the MADM solution is almost identical to the RK4 solution, whereas the MPIM also provides a close approximation with only minor deviations. Overall, the results show that the multistage methods provide substantially more accurate approximations to the RK4 reference solution than the standard PIM and ADM methods for the susceptible rodent compartment.

Table 6.11
Comparison of Absolute Errors for the Susceptible Rodent Compartment Between RK8 with Time Step ($\Delta t = 0.01$), 7-iteration PIM, 4-iteration MPIM with Time Step ($\Delta t = 0.01$), 7-iteration ADM, and 4-iteration MADM with Time Step ($\Delta t = 0.01$).

Time	|RK8-PIM|	|RK8-MPIM|	|RK8-ADM|	|RK8-MADM|
10	5.0379×10^{0}	4.9230×10^{0}	4.9252×10^{0}	4.9232×10^{0}
20	2.1755×10^{1}	1.8101×10^{1}	1.8546×10^{1}	1.8103×10^{1}
30	6.9271×10^{1}	3.7462×10^{1}	4.5448×10^{1}	3.7471×10^{1}
40	2.0305×10^{2}	4.7841×10^{1}	9.1117×10^{1}	4.7860×10^{1}
50	5.3895×10^{2}	3.5713×10^{1}	1.1004×10^{2}	3.5737×10^{1}
60	1.2943×10^{3}	1.5520×10^{1}	1.4164×10^{2}	1.5536×10^{1}
70	3.0509×10^{3}	4.2894×10^{0}	1.2185×10^{3}	4.2962×10^{0}
80	7.3904×10^{3}	9.0638×10^{-1}	3.9969×10^{3}	9.0924×10^{-1}
90	1.7864×10^{4}	2.1681×10^{-1}	9.7110×10^{3}	2.1888×10^{-1}
100	4.1363×10^{4}	9.3421×10^{-2}	2.0033×10^{4}	9.5612×10^{-2}
110	9.0025×10^{4}	6.1598×10^{-2}	3.7149×10^{4}	6.4093×10^{-2}
120	1.8375×10^{5}	4.5717×10^{-2}	6.3826×10^{4}	4.8559×10^{-2}

Table 6.11 presents the absolute differences between the numerical solutions for the susceptible rodent compartment, $S_r(t)$, and the RK8 solution. The PIM and ADM produce large differences that generally increase as time progresses, reaching 1.8375×10^5 and 6.3826×10^4, respectively, at $t = 120$. In comparison, the MPIM and MADM produce much smaller differences, which decrease from approximately 10^0 at the early stage to the order of 10^{-2} at the end of the interval. The MPIM gives marginally smaller differences than MADM at all selected time points, although the differences between these two multistage solutions are very close. These findings indicate that both MPIM and MADM remain close to the RK8 reference solution, whereas the standard PIM and ADM solutions exhibit substantial deviations over the considered interval.

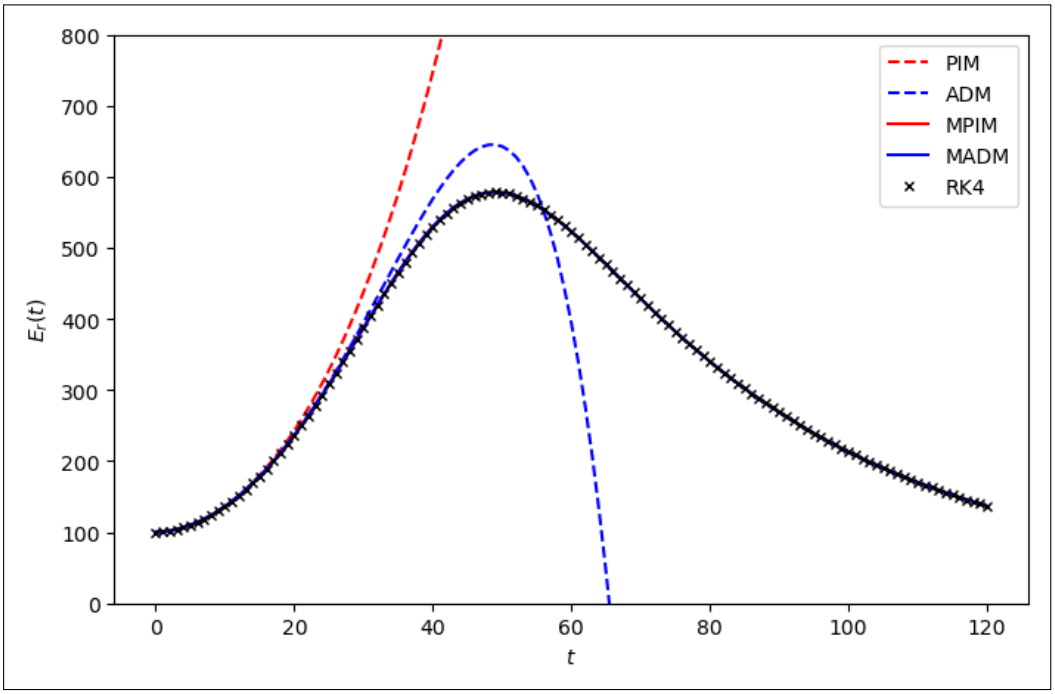

Figure 6.7 Solution of Exposed Rodent, E_r using 7-iteration PIM, 4-iteration MPIM on the Time Step ($\Delta t = 0.01$), 7-iteration ADM, 7-iteration MADM on the Time Step($\Delta t = 0.01$), RK4 on the Time Step ($\Delta t = 0.01$) and RK8 on the Time Step ($\Delta t = 0.01$).

Figure 6.7 illustrates the numerical solutions for the exposed rodent compartment, $E_r(t)$, obtained using the PIM, MPIM, ADM, MADM, RK4, and RK8 methods. The PIM solution increases rapidly after approximately $t = 30$ and diverges from the other numerical solutions, while the ADM solution initially follows the general trend but later decreases sharply and becomes unstable after approximately $t = 55$. Hence, both PIM and ADM do not provide reliable approximations for $E_r(t)$ over the full interval. In contrast, the MPIM, MADM, RK4, and RK8 solutions remain stable throughout $0 \leq t \leq 120$ and exhibit a similar overall behaviour. The exposed rodent population increases gradually from its initial value, reaches a peak at approximately $t = 50$ to $t = 55$, and then decreases steadily towards the end of the interval. The MPIM and MADM solutions closely follow the RK4 solution, while the RK8 solution shows the same overall pattern with slightly higher values at the later stage of the interval. These results indicate that the multistage methods provide stable and reliable approximations for the exposed rodent compartment.

Table 6.12
Comparison of Absolute Errors for the Exposed Rodent Compartment Between RK4 with Time Step ($\Delta t = 0.01$), 7-iteration PIM, 4-iteration MPIM with Time Step ($\Delta t = 0.01$), 7-iteration ADM, and 4-iteration MADM with Time Step ($\Delta t = 0.01$).

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
10	2.2244×10^{-1}	1.6197×10^{-4}	2.0056×10^{-3}	4.0860×10^{-7}
20	6.6541×10^{0}	1.7092×10^{-3}	4.2524×10^{-1}	7.3330×10^{-9}
30	5.0753×10^{1}	7.2304×10^{-3}	7.4520×10^{0}	2.7636×10^{-7}
40	2.1789×10^{2}	1.5602×10^{-2}	3.9207×10^{1}	2.1881×10^{-7}
50	6.4848×10^{2}	1.5759×10^{-2}	6.5086×10^{1}	2.0244×10^{-7}
60	1.5627×10^{3}	4.8979×10^{-3}	1.2919×10^{2}	2.6332×10^{-7}
70	3.5742×10^{3}	4.1544×10^{-3}	9.5777×10^{2}	4.9825×10^{-7}
80	8.3554×10^{3}	6.5211×10^{-3}	2.9520×10^{3}	4.4697×10^{-7}
90	1.9587×10^{4}	5.6819×10^{-3}	6.7317×10^{3}	2.5607×10^{-7}
100	4.4319×10^{4}	4.2664×10^{-3}	1.2988×10^{4}	4.4418×10^{-7}
110	9.4878×10^{4}	3.0141×10^{-3}	2.2431×10^{4}	1.2865×10^{-7}
120	1.9138×10^{5}	2.0094×10^{-3}	3.5686×10^{4}	4.5677×10^{-7}

Table 6.12 presents the absolute differences between the numerical solutions for the exposed rodent compartment, $E_r(t)$, and the RK4 solution. Although the ADM difference is relatively small at $t = 10$, it increases substantially as time progresses, reaching 3.5686×10^4 at $t = 120$. The PIM solution also exhibits rapidly increasing differences, from 2.2244×10^{-1} at $t = 10$ to 1.9138×10^5 at $t = 120$. In contrast, the MPIM produces small differences ranging from 1.6197×10^{-4} to 1.5759×10^{-2}, while the MADM produces the smallest differences, remaining within the order of 10^{-9} to 10^{-7} throughout the interval. This shows that the MADM solution is almost identical to the RK4 solution, whereas the MPIM also provides a close approximation with only minor deviations. Overall, the multistage methods give substantially closer approximations to the RK4 reference solution than the standard PIM and ADM methods for the exposed rodent compartment.

Table 6.13
Comparison of Absolute Errors for the Exposed Rodent Compartment Between RK8 with Time Step ($\Delta t = 0.01$), 7-iteration PIM, 4-iteration MPIM with Time Step ($\Delta t = 0.01$), 7-iteration ADM, and 4-iteration MADM with Time Step ($\Delta t = 0.01$).

Time	|RK8-PIM|	|RK8-MPIM|	|RK8-ADM|	|RK8-MADM|
10	3.4925×10^{0}	3.2699×10^{0}	3.2721×10^{0}	3.2701×10^{0}
20	1.9118×10^{1}	1.2462×10^{1}	1.2889×10^{1}	1.2464×10^{1}
30	7.4433×10^{1}	2.3673×10^{1}	3.1132×10^{1}	2.3681×10^{1}
40	2.4042×10^{2}	2.2513×10^{1}	6.1736×10^{1}	2.2529×10^{1}
50	6.4909×10^{2}	5.9416×10^{-1}	6.5696×10^{1}	6.0992×10^{-1}
60	1.5394×10^{3}	2.3212×10^{1}	1.5240×10^{2}	2.3207×10^{1}
70	3.5413×10^{3}	3.2924×10^{1}	9.9070×10^{2}	3.2928×10^{1}
80	8.3228×10^{3}	3.2656×10^{1}	2.9847×10^{3}	3.2662×10^{1}
90	1.9558×10^{4}	2.9239×10^{1}	6.7609×10^{3}	2.9245×10^{1}
100	4.4294×10^{4}	2.5379×10^{1}	1.3014×10^{4}	2.5383×10^{1}
110	9.4856×10^{4}	2.1736×10^{1}	2.2453×10^{4}	2.1739×10^{1}
120	1.9137×10^{5}	1.8451×10^{1}	3.5704×10^{4}	1.8453×10^{1}

Table 6.13 presents the absolute differences between the numerical solutions for the exposed rodent compartment, $E_r(t)$, and the RK8 solution. The PIM and ADM produce substantially larger differences than the multistage methods, with their differences increasing to 1.9137×10^5 and 3.5704×10^4, respectively, at $t = 120$. In comparison, the MPIM and MADM produce much smaller and closely comparable differences throughout the interval. Their differences initially increase to approximately 10^1 at the middle of the interval, before decreasing to 1.8451×10^1 and 1.8453×10^1, respectively, at $t = 120$. The MPIM gives marginally smaller differences than MADM at most selected time points, although the differences between both methods are nearly identical. These results indicate that both multistage methods remain close to the RK8 reference solution, whereas the standard PIM and ADM methods deviate substantially as time increases.

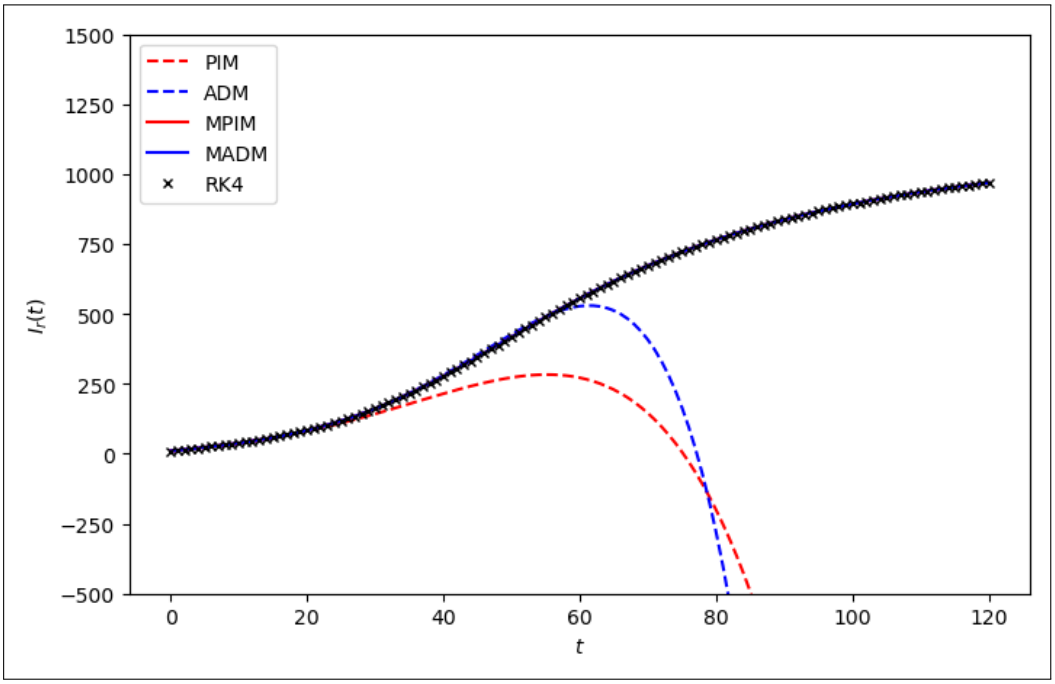

Figure 6.8 Solution of Infected Rodent, I_r using 7-iteration PIM, 4-iteration MPIM on the Time Step ($\Delta t = 0.01$), 7-iteration ADM, 7-iteration MADM on the Time Step($\Delta t = 0.01$), RK4 on the Time Step ($\Delta t = 0.01$) and RK8 on the Time Step ($\Delta t = 0.01$).

Figure 6.8 illustrates the numerical solutions for the infected rodent compartment, $I_r(t)$, obtained using the PIM, MPIM, ADM, MADM, RK4, and RK8 methods. The PIM and ADM solutions initially follow the general increasing trend, but both become unstable after approximately $t = 60$ and subsequently decrease sharply into negative values, indicating that they do not provide reliable approximations over the full interval. In contrast, the MPIM, MADM, RK4, and RK8 solutions remain stable throughout $0 \leq t \leq 120$ and exhibit a similar monotonic increasing behaviour. The infected rodent population increases gradually at the early stage, followed by a more rapid increase around the middle of the interval, before continuing to rise at a slower rate towards the end of the interval. The MPIM and MADM solutions closely follow the RK4 solution, while the RK8 solution shows the same overall pattern with slightly lower values at the later stage of the interval. These results indicate that the multistage methods provide stable and reliable approximations for the infected rodent compartment.

Table 6.14
Comparison of Absolute Errors for the Infected Rodent Compartment Between RK4 with Time Step ($\Delta t = 0.01$), 7-iteration PIM, 4-iteration MPIM with Time Step ($\Delta t = 0.01$), 7-iteration ADM, and 4-iteration MADM with Time Step ($\Delta t = 0.01$).

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
10	1.0777×10^{-1}	1.0199×10^{-5}	1.9786×10^{-5}	2.1401×10^{-7}
20	3.0018×10^{0}	1.9282×10^{-4}	1.7787×10^{-2}	1.2222×10^{-8}
30	1.8953×10^{1}	1.2128×10^{-3}	5.2510×10^{-1}	8.4042×10^{-8}
40	6.2699×10^{1}	4.1138×10^{-3}	4.0481×10^{0}	1.5517×10^{-7}
50	1.4528×10^{2}	8.3354×10^{-3}	9.2070×10^{0}	3.8315×10^{-7}
60	2.8398×10^{2}	1.1000×10^{-2}	2.8001×10^{1}	4.7890×10^{-7}
70	5.2776×10^{2}	1.0884×10^{-2}	2.6505×10^{2}	5.6798×10^{-8}
80	9.6623×10^{2}	9.3643×10^{-3}	1.0458×10^{3}	2.4026×10^{-7}
90	1.7240×10^{3}	7.7425×10^{-3}	2.9794×10^{3}	2.4040×10^{-9}
100	2.9572×10^{3}	6.4442×10^{-3}	7.0435×10^{3}	7.4460×10^{-8}
110	4.8540×10^{3}	5.4948×10^{-3}	1.4716×10^{4}	2.5308×10^{-7}
120	7.6364×10^{3}	4.8351×10^{-3}	2.8136×10^{4}	7.1490×10^{-8}

Table 6.14 presents the absolute differences between the numerical solutions for the infected rodent compartment, $I_r(t)$, and the RK4 solution. The PIM and ADM differences increase substantially as time progresses. Although the ADM difference is initially small at $t = 10$, it increases from 1.9786×10^{-5} to 2.8136×10^{4} at $t = 120$, while the PIM difference increases from 1.0777×10^{-1} to 7.6364×10^{3}. In contrast, the MPIM produces very small differences, ranging from 1.0199×10^{-5} to 1.1000×10^{-2}, whereas the MADM produces the smallest differences, remaining within the order of 10^{-9} to 10^{-7} throughout the interval. This indicates that the MADM solution is almost identical to the RK4 solution, while the MPIM also provides a close approximation with only minor deviations. Overall, the multistage methods provide substantially closer approximations to the RK4 reference solution than the standard PIM and ADM methods for the infected rodent compartment.

Table 6.15
Comparison of Absolute Errors for the Infected Rodent Compartment Between RK8 with Time Step ($\Delta t = 0.01$), 7-iteration PIM, 4-iteration MPIM with Time Step ($\Delta t = 0.01$), 7-iteration ADM, and 4-iteration MADM with Time Step ($\Delta t = 0.01$).

Time	|RK8-PIM|	|RK8-MPIM|	|RK8-ADM|	|RK8-MADM|
10	1.5449×10^{0}	1.6526×10^{0}	1.6527×10^{0}	1.6526×10^{0}
20	2.6353×10^{0}	5.6368×10^{0}	5.6548×10^{0}	5.6370×10^{0}
30	5.1685×10^{0}	1.3783×10^{1}	1.4310×10^{1}	1.3785×10^{1}
40	3.7381×10^{1}	2.5315×10^{1}	2.9367×10^{1}	2.5319×10^{1}
50	1.1018×10^{2}	3.5095×10^{1}	4.4310×10^{1}	3.5103×10^{1}
60	2.4528×10^{2}	3.8692×10^{1}	1.0703×10^{1}	3.8703×10^{1}
70	4.9059×10^{2}	3.7158×10^{1}	2.2788×10^{2}	3.7169×10^{1}
80	9.3273×10^{2}	3.3490×10^{1}	1.0123×10^{3}	3.3499×10^{1}
90	1.6946×10^{3}	2.9367×10^{1}	2.9501×10^{3}	2.9375×10^{1}
100	2.9318×10^{3}	2.5368×10^{1}	7.0182×10^{3}	2.5374×10^{1}
110	4.8323×10^{3}	2.1677×10^{1}	1.4695×10^{4}	2.1683×10^{1}
120	7.6181×10^{3}	1.8361×10^{1}	2.8117×10^{4}	1.8366×10^{1}

Table 6.15 presents the absolute differences between the numerical solutions for the infected rodent compartment, $I_r(t)$, and the RK8 solution. The PIM and ADM methods produce increasingly large differences as time progresses. The PIM difference increases from 1.5449×10^{0} at $t = 10$ to 7.6181×10^{3} at $t = 120$, while the ADM difference increases from 1.6527×10^{0} to 2.8117×10^{4} over the same interval. In contrast, the MPIM and MADM produce considerably smaller differences throughout the selected interval. Their differences increase during the early stage, reaching 3.8692×10^{1} and 3.8703×10^{1}, respectively, at $t = 60$, before gradually decreasing to 1.8361×10^{1} and 1.8366×10^{1}, respectively, at $t = 120$. The MPIM produces marginally smaller differences than MADM at all selected time points. Overall, both multistage methods remain substantially closer to the RK8 reference solution than the standard PIM and ADM methods for the infected rodent compartment.

6.3 COVID-19 Model

In this section, consider the nonlinear epidemiological model describing the transmission dynamics of COVID-19 based on the Susceptible–Infected–Recovered (SIR) framework proposed by Ariffin et al. (2021). This model captures the evolution of three key population compartments: susceptible $S(t)$, infected $I(t)$, and recovered

$R(t)$ individuals. The interactions among these compartments are governed by the following system of nonlinear ordinary differential equations (ODEs):

$$\frac{dS}{dt} = -\frac{\beta SI}{N}, \tag{6.42}$$

$$\frac{dI}{dt} = \frac{\beta SI}{N} - \gamma I, \tag{6.43}$$

$$\frac{dR}{dt} = \gamma I, \tag{6.44}$$

where β represents the transmission rate of infection, γ denotes the recovery rate, and N is the total constant population size. The model assumes that there are no births or deaths in the population during the study period, so $N = S + I + R$.

The initial conditions associated with the system (6.42)–(6.44) are given by (Ariffin et al., 2021)

$$S(0) = S_0, \quad I(0) = I_0, \quad R(0) = 0, \tag{6.45}$$

where S_0, I_0, and R_0 are the initial numbers of susceptible, infected, and recovered individuals, respectively. To simplify the system, the normalized variables are introduced as

$$s(t) = \frac{S(t)}{N}, \quad i(t) = \frac{I(t)}{N}, \quad r(t) = \frac{R(t)}{N}, \tag{6.46}$$

which satisfy the conservation relation $s + i + r = 1$. Substituting these expressions into equations (6.42)–(6.44) yields the dimensionless form of the SIR model (Ariffin et al., 2021):

$$\frac{ds}{dt} = -\beta si, \tag{6.47}$$

$$\frac{di}{dt} = \beta si - \gamma i, \tag{6.48}$$

$$\frac{dr}{dt} = \gamma i. \tag{6.49}$$

Where the initial condition are given by

$$s(0) = 0.999, \quad i(0) = 0.001, \quad r(0) = 0. \tag{6.50}$$

The parameters β and γ are positive constants determined empirically based on

epidemiological data. The basic reproduction number, $R_0 = \beta/\gamma$, determines whether the infection will spread ($R_0 > 1$) or die out ($R_0 < 1$).

The system (6.47)–(6.49) is nonlinear due to the product term si, which couples the susceptible and infected classes. Analytical solutions are generally not obtainable in closed form, motivating the use of iterative and semi–analytical numerical methods. In this study, the system is solved using the PIM and the MPIM.

The PIM generates approximate analytical solutions by transforming the system into its integral form and performing successive iterations. However, the standard PIM may suffer from slow convergence or instability when applied to nonlinear epidemic systems over long time intervals. To address this limitation, the MPIM extends the PIM framework by partitioning the computational interval into multiple sub–intervals (stages). At each stage, local corrections are applied and the final value of one stage serves as the initial condition for the next.

This staged formulation enhances global convergence and minimizes local truncation errors, allowing the MPIM to achieve improved accuracy and stability while retaining the analytical transparency of the PIM. Consequently, the MPIM serves as a robust and efficient numerical–analytical tool for studying the nonlinear spread dynamics of COVID-19 and related epidemic systems.

6.3.1 PIM and MPIM solutions

PIM:

According to PIM, construct an iterative scheme for COVID-19 model (6.47)–(6.49) as follows

$$s_{(n+1)}(t) = s(0)(t) + \int_0^t (-\beta s_n(t) i_n(t))dt, \tag{6.51}$$

$$i_{(n+1)}(t) = i(0)(t) + \int_0^t (\beta s_n(t) i_n(t) - \gamma i_n(t))dt, \tag{6.52}$$

$$r_{(n+1)}(t) = r(0)(t) + \int_0^t (\gamma i_n(t))dt, \tag{6.53}$$

by substituting the initial condition in (6.50) and the parameters $\beta = 0.22$ and $\gamma = 0.09$ into systems (6.51)–(6.53), the following expression are obtained.

For $n = 0$:

$$s_1(t) = 0.999 - 0.0002197t, \tag{6.54}$$

$$i_1(t) = 0.001 + 0.0001297t, \tag{6.55}$$

$$r_1(t) = 0.00009t. \tag{6.56}$$

The iterative process converges at $n = 3$, where the approximate solution $s_4(t)$, $i_4(t)$ and $r_4(t)$ are obtained. At this stage, the tolerance condition $\varepsilon < 10^{-10}$ is achieved, indicating that further iterations would yield negligible changes in accuracy. Further details regarding the equation can be found in Appendix G.

MPIM

According to MPIM, an iterative scheme for COVID-19 model (6.47)–(6.49) as follows

$$s_{(n+1),j}(t) = s_{n,j-1}(t_{j-1}) + \int_0^t (-\beta s_{n,j-1}(t) i_{n,j-1}(t))dt, \tag{6.57}$$

$$i_{(n+1,j)}(t) = i_{n,j-1}(t_{j-1}) + \int_0^t (\beta s_{n,j-1}(t) i_{n,j-1}(t) - \gamma i_{n,j-1}(t))dt, \tag{6.58}$$

$$r_{(n+1),j}(t) = r_{n,j-1}(t_{j-1}) + \int_0^t (\gamma i_{n,j-1}(t))dt, \tag{6.59}$$

by substituting the initial condition in (6.50) and the parameters $\beta = 0.22$ and $\gamma = 0.09$ into systems (6.57)–(6.59), the following expression are obtained.

For $n = 3, j = 1$:

$$s_{4,1}(t) = 0.999 - 0.00021978t - 0.000014237t^2 - 6.1205 \times 10^{-7}t^3 - 1.9551 \times 10^{-8} t^4 + 2.4131 \times 10^{-11}t^5 + 5.6407 \times 10^{-13}t^6 + 6.6104 \times 10^{-15}t^7 - 1.0942 \times 10^{-17}t^8 - 1.2118 \times 10^{-19}t^9 + 1.0479 \times 10^{-22}t^{10} + 5.2357 \times 10^{-25}t^{11} - 3.3130 \times 10^{-28}t^{12} + 7.5331 \times 10^{-32}t^{13} - 7.5021 \times 10^{-36}t^{14} + 2.7731 \times 10^{-40}t^{15}, \quad (6.60)$$

$$i_{4,1}(t) = 0.001 + 0.00012978t + 0.0000083972t^2 + 3.6013 \times 10^{-7}t^3 + 1.1448 \times 10^{-8} t^4 - 1.9255 \times 10^{-11}t^5 - 4.8565 \times 10^{-13}t^6 - 6.6327 \times 10^{-15}t^7 + 1.0943 \times 10^{-17}t^8 + 1.2118 \times 10^{-19}t^9 - 1.0479 \times 10^{-22}t^{10} - 5.2357 \times 10^{-25}t^{11} + 3.3130 \times 10^{-28}t^{12} - 7.5331 \times 10^{-32}t^{13} + 7.5021 \times 10^{-36}t^{14} - 2.7731 \times 10^{-40}t^{15}, \quad (6.61)$$

$$r_{4,1}(t) = 0.00009t + 0.0000058401t^2 + 2.5192 \times 10^{-7}t^3 + 8.1029 \times 10^{-9}t^4 - 4.8758 \times 10^{-12}t^5 - 7.8423 \times 10^{-14}t^6 + 2.2320 \times 10^{-17}t^7 - 1.5469 \times 10^{-21}t^8. \quad (6.62)$$

The iterative process was continued up to $n = 3, j = 6000$. Due to space limitations and the extensive nature of the iterative solutions, only the first iterations ($n = 3, j = 1$) is presented in this section, while the complete series is provided in Appendix G.

6.3.2 Results and discussion

Figure (6.9) - (6.11) show the solutions for $s(t)$ and $i(t)$ and $r(t)$, respectively, obtained by the 4-iteration PIM, the 4-iteration MPIM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$) for the case $\beta = 0.22$ and $\gamma = 0.09$ with the initial condition in (6.50). It is observed that the 4-iteration MPIM solutions agree very well with the RK4 solutions for t up to $t = 60$, while the 4-iteration PIM solutions are only valid for $t \leq 20$.

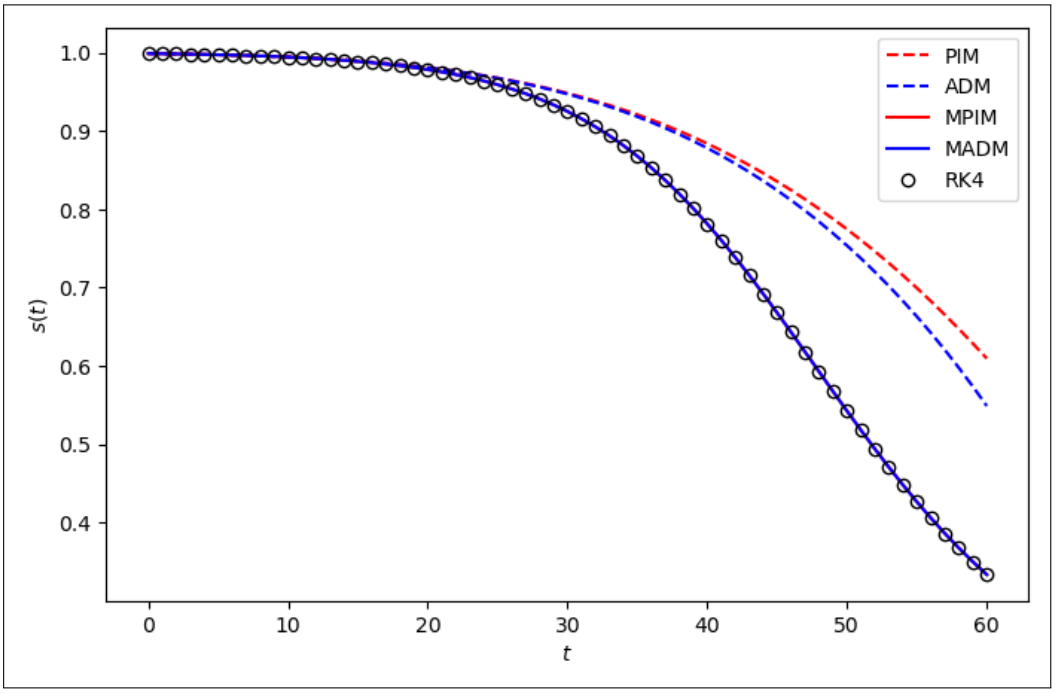

Figure 6.9 Solution of Susceptible COVID-19, $s(t)$ using 4-iteration PIM, 4-iteration MPIM ($\Delta t = 0.01$), 4-iteration ADM, 4-iteration MADM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$).

Figure 6.9 illustrates the numerical behaviour of the susceptible population, $S(t)$, for the COVID-19 model computed using the 4-iteration of PIM, the 4-iteration of MPIM, the 4-iteration of ADM, the 4-iteration of MADM, and RK4 over the interval $0 \leq t \leq 60$. Based on the figure, the RK4 solution shows that the susceptible population decreases gradually at the early stage and then declines more rapidly after approximately $t = 35$. This behaviour indicates a reduction in the susceptible population over time as individuals move into other compartments in the model. The classical PIM and ADM solutions are able to follow the initial decreasing trend for a short interval, but their curves begin to deviate from the RK4 reference solution as time increases. In particular, both PIM and ADM remain higher than the RK4 solution at the later stage of the simulation, suggesting that the non-multistage methods underestimate the rate of decrease in the susceptible population over the longer interval. In contrast, the MPIM and MADM solutions show stronger graphical agreement with the RK4 solution throughout the simulation interval. Both multistage methods follow the decreasing pattern of RK4, including the sharper decline after the middle of the interval. Therefore, the graphical result suggests that the multistage implementation improves the numerical behaviour of both PIM and ADM for the

susceptible compartment in the COVID-19 model. However, this observation is based on the plotted curves, and the corresponding absolute error and CPU execution time results should also be considered for a more detailed comparison.

Table 6.16
Comparison of Absolute Errors for the Susceptible Compartment of the COVID-19 Model Between RK4, 4-iteration PIM, 4-iteration MPIM, 4-iteration ADM, and 4-iteration MADM.

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
10	6.3551×10^{-5}	7.7716×10^{-16}	6.0509×10^{-5}	1.4684×10^{-8}
20	2.5388×10^{-3}	1.6653×10^{-14}	2.4174×10^{-3}	8.2572×10^{-8}
30	2.3324×10^{-2}	5.5067×10^{-14}	2.2191×10^{-2}	1.9166×10^{-7}
40	1.0249×10^{-1}	2.0637×10^{-11}	9.6727×10^{-2}	4.3031×10^{-8}
50	2.3206×10^{-1}	5.3937×10^{-10}	2.1119×10^{-1}	4.5136×10^{-7}
60	2.7599×10^{-1}	3.1844×10^{-10}	2.1538×10^{-1}	3.4991×10^{-7}

Table 6.16 presents the comparison of absolute errors for the susceptible compartment of the COVID-19 model between the RK4 reference solution and the 4-iteration PIM, 4-iteration MPIM, 4-iteration ADM, and 4-iteration MADM over the interval $0 \leq t \leq 60$. Based on the results, the classical PIM and ADM show increasing absolute errors as time progresses. For PIM, the error increases from 6.3551×10^{-5} at $t = 10$ to 2.7599×10^{-1} at $t = 60$, while ADM increases from 6.0509×10^{-5} to 2.1538×10^{-1} over the same interval. This indicates that both non-multistage methods gradually depart from the RK4 solution as the simulation time becomes longer, which is consistent with the graphical observation where PIM and ADM remain above the RK4 curve at later times. In comparison, MPIM produces much smaller errors, with values ranging from the order of 10^{-16} to 10^{-10}. This suggests that the multistage implementation helps to maintain closer agreement with the RK4 solution for the susceptible compartment under the selected computational setting. Meanwhile, MADM also gives small displayed errors, mostly within the order of 10^{-8} to 10^{-7}, indicating that it remains close to the RK4 reference solution throughout the selected time points. Overall, the absolute error results suggest that the multistage approaches reduce the discrepancy from RK4 compared with their corresponding classical forms. However, this conclusion should be interpreted within the numerical precision and computational setting used in this study.

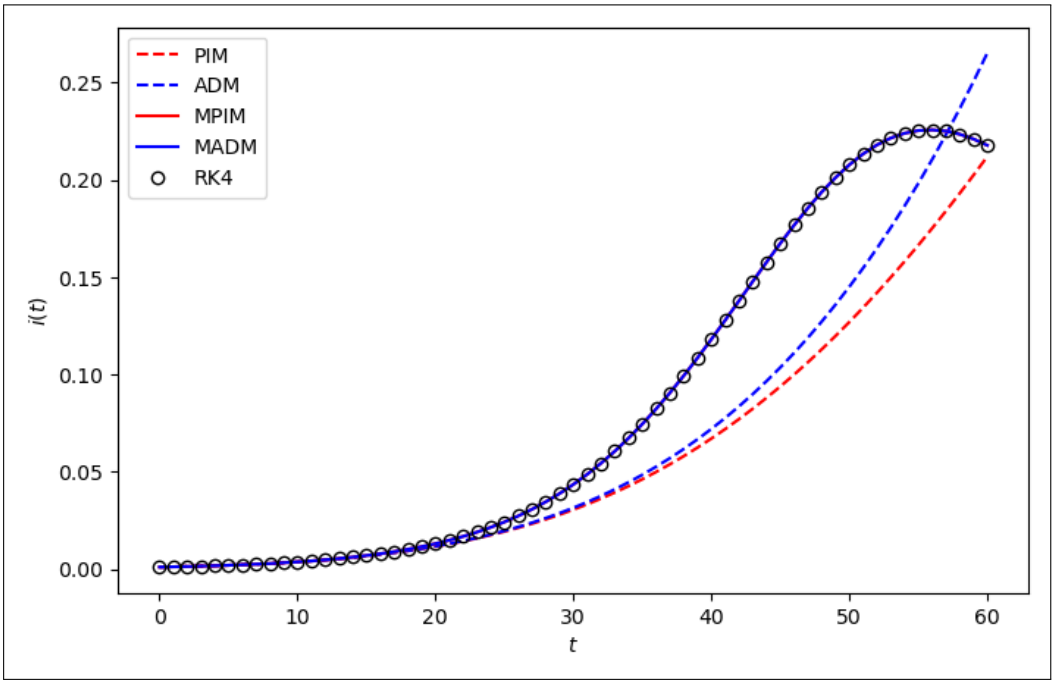

Figure 6.10 Solution of Infected COVID-19, $i(t)$ using 4-iteration PIM, 4-iteration MPIM ($\Delta t = 0.01$), 4-iteration ADM, 4-iteration MADM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$).

Figure 6.10 illustrates the numerical behaviour of the infected population, $I(t)$, for the COVID-19 model computed using the 4-iteration of PIM, the 4-iteration of MPIM, the 4-iteration of ADM, the 4-iteration of MADM, and RK4 over the interval $0 \leq t \leq 60$. Based on the figure, the RK4 solution shows that the infected population increases gradually at the early stage, rises more rapidly after approximately $t = 35$, reaches a peak around $t = 55$, and then begins to decrease slightly towards the end of the simulation. The classical PIM and ADM solutions are able to follow the early increasing trend for a short time, but both methods begin to deviate from the RK4 reference solution as time increases. In particular, the PIM solution remains lower than the RK4 curve at the later stage, suggesting that it underestimates the infected population during the growth phase. Meanwhile, the ADM solution increases beyond the RK4 curve near the end of the interval, indicating a noticeable deviation after approximately $t = 50$. In contrast, the MPIM and MADM solutions show better graphical agreement with the RK4 solution throughout most of the simulation interval. Both multistage methods are able to reproduce the increasing pattern and the peak behaviour of the infected population more consistently than their corresponding classical forms. Therefore, the graphical result suggests that

the multistage implementation improves the numerical behaviour of both PIM and ADM for the infected compartment in the COVID-19 model. However, since this comparison is based on the plotted curves, the corresponding absolute error and CPU execution time results should also be considered for a more detailed evaluation.

Table 6.17
Comparison of Absolute Errors for the Infected Compartment of the COVID-19 model Between RK4, 4-iteration PIM, 4-iteration MPIM, 4-iteration ADM, and 4-iteration MADM.

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
10	3.7307×10^{-5}	8.7170×10^{-17}	3.4831×10^{-5}	8.4442×10^{-9}
20	1.4704×10^{-3}	1.0287×10^{-15}	1.3696×10^{-3}	4.5118×10^{-8}
30	1.3004×10^{-2}	2.0900×10^{-14}	1.2046×10^{-2}	8.3667×10^{-8}
40	5.1025×10^{-2}	7.3460×10^{-12}	4.6080×10^{-2}	1.3156×10^{-8}
50	8.0806×10^{-2}	5.3320×10^{-10}	6.2670×10^{-2}	2.9006×10^{-7}
60	5.7589×10^{-3}	1.5501×10^{-9}	4.7462×10^{-2}	7.6960×10^{-8}

Table 6.17 presents the comparison of absolute errors for the infected compartment of the COVID-19 model between the RK4 reference solution and the 4-iteration PIM, 4-iteration MPIM, 4-iteration ADM, and 4-iteration MADM over the interval $0 \leq t \leq 60$. Based on the results, the classical PIM and ADM show larger errors than the corresponding multistage methods. For PIM, the error increases from 3.7307×10^{-5} at $t = 10$ to 8.0806×10^{-2} at $t = 50$, before decreasing to 5.7589×10^{-3} at $t = 60$. Meanwhile, ADM increases from 3.4831×10^{-5} at $t = 10$ to 6.2670×10^{-2} at $t = 50$, before slightly decreasing to 4.7462×10^{-2} at $t = 60$. This indicates that the classical non-multistage methods begin to show noticeable differences from the RK4 solution as the simulation time increases, which is consistent with the graphical observation. In comparison, MPIM produces much smaller errors, with values ranging from the order of 10^{-17} to 10^{-9}. This suggests that the multistage implementation helps MPIM remain close to the RK4 solution for the infected compartment under the selected computational setting. MADM also gives small displayed errors, mostly within the order of 10^{-9} to 10^{-7}, indicating good agreement with the RK4 reference solution at the selected time points. Overall, the absolute error results suggest that the multistage approaches reduce the discrepancy from RK4 compared with their corresponding classical forms. However, the conclusion should be interpreted within the numerical precision and computational setting used in this study.

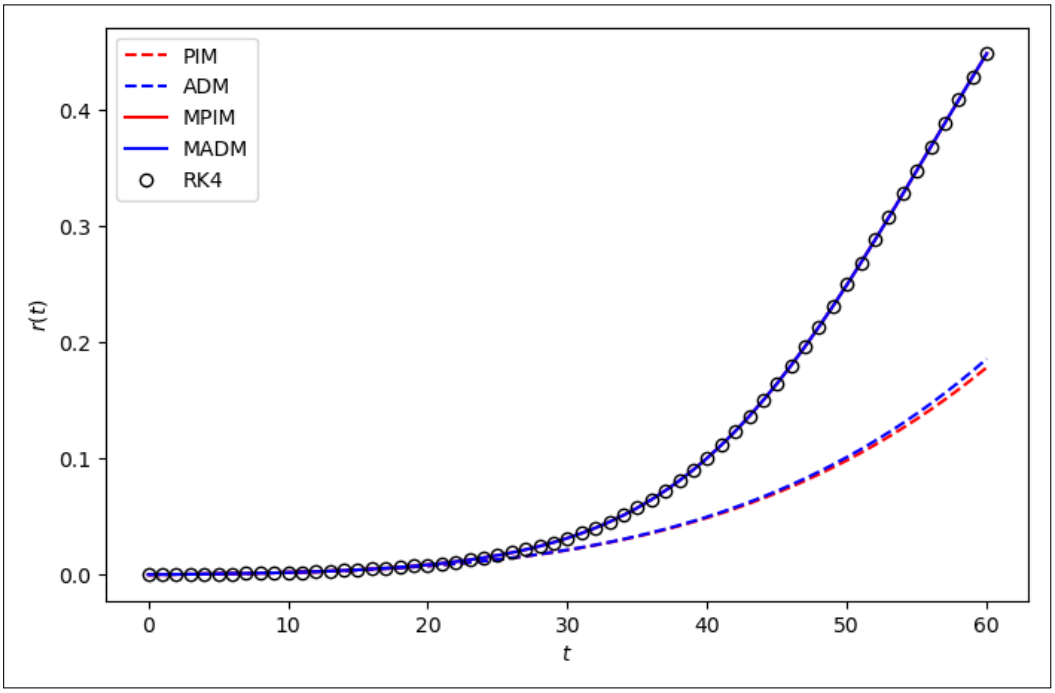

Figure 6.11 Solution of Recovered COVID-19, $r(t)$ using 4-iteration PIM, 4-iteration MPIM ($\Delta t = 0.01$), 4-iteration ADM, 4-iteration MADM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$).

Figure 6.11 illustrates the numerical behaviour of the recovered population, $R(t)$, for the COVID-19 model computed using the 4-iteration of PIM, the 4-iteration of MPIM, the 4-iteration of ADM, the 4-iteration of MADM, and RK4 over the interval $0 \leq t \leq 60$. Based on the figure, the RK4 solution shows that the recovered population increases gradually at the early stage and rises more rapidly after approximately $t = 35$. This indicates an increasing number of recovered individuals as the disease progresses over time. The classical PIM and ADM solutions are able to follow the initial increasing trend for a short time, but both methods begin to deviate from the RK4 reference solution as the simulation time increases. In particular, PIM and ADM remain lower than the RK4 curve at the later stage of the simulation, suggesting that the classical non-multistage methods underestimate the growth of the recovered population over the longer interval. In contrast, the MPIM and MADM solutions show better graphical agreement with the RK4 solution throughout the simulation interval. Both multistage methods follow the increasing pattern of the RK4 solution, including the rapid growth after the middle of the interval. Therefore, the graphical result suggests that the multistage implementation improves the numerical behaviour of both PIM and ADM for the recovered compartment in the

COVID-19 model. However, since this comparison is based on the plotted curves, the corresponding absolute error and CPU execution time results should also be considered for a more detailed evaluation.

Table 6.18
Comparison of Absolute Errors for the Recovered Compartment of the COVID-19 Model Between RK4, 7-iteration PIM, 4-iteration MPIM, 7-iteration ADM, and 4-iteration MADM.

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
10	2.6244×10^{-5}	6.0065×10^{-17}	2.5678×10^{-5}	6.2827×10^{-9}
20	1.0684×10^{-3}	4.1980×10^{-16}	1.0478×10^{-3}	3.7324×10^{-8}
30	1.0320×10^{-2}	7.0985×10^{-15}	1.0145×10^{-2}	1.0784×10^{-7}
40	5.1464×10^{-2}	3.1003×10^{-14}	5.0647×10^{-2}	5.6187×10^{-8}
50	1.5125×10^{-1}	3.0253×10^{-10}	1.4852×10^{-1}	1.6210×10^{-7}
60	2.7023×10^{-1}	6.0391×10^{-10}	2.6284×10^{-1}	4.2757×10^{-7}

Table 6.18 presents the comparison of absolute errors for the recovered compartment of the COVID-19 model between the RK4 reference solution and the 7-iteration PIM, 4-iteration MPIM, 7-iteration ADM, and 4-iteration MADM over the interval $0 \leq t \leq 60$. Based on the results, the classical PIM and ADM show increasing absolute errors as time progresses. For PIM, the error increases from 2.6244×10^{-5} at $t = 10$ to 2.7023×10^{-1} at $t = 60$, while ADM increases from 2.5678×10^{-5} to 2.6284×10^{-1} over the same interval. This indicates that both non-multistage methods gradually depart from the RK4 solution as the simulation time increases, which is consistent with the graphical observation where PIM and ADM remain lower than the RK4 curve at later times. In comparison, MPIM gives much smaller errors, with values ranging from the order of 10^{-17} to 10^{-10}. This suggests that the multistage implementation helps MPIM remain close to the RK4 reference solution for the recovered compartment under the selected computational setting. Meanwhile, MADM also produces small displayed errors, mostly within the order of 10^{-9} to 10^{-7}, indicating a small discrepancy from RK4 at the selected time points. Overall, the absolute error results suggest that the multistage approaches reduce the discrepancy from RK4 compared with their corresponding classical forms. However, this conclusion should be interpreted within the numerical precision and computational setting used in this study.

6.4 Nonlinear Biochemical Reaction

In this section, consider the well-known Michaelis–Menten biochemical reaction model by Schnell and Mendoza (1997), *i.e.* the single enzyme–substrate reaction scheme,

$$E + A \underset{k_2}{\overset{k_1}{\rightleftharpoons}} Y \xrightarrow{k_3} E + X, \tag{6.63}$$

where E is the enzyme, A the substrate, Y the intermediate complex and X the product. The time evolution of scheme (6.63) can be determined from the solution of the system of coupled nonlinear ODEs (Sen, 1988),

$$\frac{dA}{dt} = -k_1 EA + k_{-1} Y, \tag{6.64}$$

$$\frac{dE}{dt} = -k_1 EA + (k_{-1} + k_2) Y, \tag{6.65}$$

$$\frac{dY}{dt} = k_1 EA - (k_{-1} + k_2) Y, \tag{6.66}$$

$$\frac{dX}{dt} = k_2 Y, \tag{6.67}$$

subject to the initial conditions

$$A(0) = A_0, \quad E(0) = E_0, \quad Y(0) = 0, \quad X(0) = 0, \tag{6.68}$$

where the parameters k_1, k_{-1}, and k_2 are positive rate constants for each reaction. The system (6.64)–(6.67) can be reduced to only two equations for A and Y and in dimensionless form of concentrations of substrate, x, and intermediate complex between enzyme and substrate, y, are given by Sen (1988)

$$\frac{dx}{dt} = -x + (\beta - \alpha) y + xy, \tag{6.69}$$

$$\frac{dy}{dt} = \frac{1}{\varepsilon}[x - \beta y - xy], \tag{6.70}$$

subject to the initial conditions

$$x(0) = 1, \quad y(0) = 0. \tag{6.71}$$

where α, β and ε are dimensionless parameters.

The time evolution of the reaction can be determined from the traditional purely numerical methods like the classical fourth–order Runge–Kutta method (RK4), The present study aims to obtain a numerical–analytical solution of the system of coupled nonlinear ordinary differential equations (6.69) and (6.70) using the PIM and MPIM approaches. The limitation of global convergence inherent in the standard PIM is addressed and effectively overcome through the implementation of the MPIM.

6.4.1 PIM and MPIM solutions

In this section, the system of equations (6.69) - (6.70) representing the Michaelis Menten model is solved using the standard PIM and the MPIM. A comparison with the classical RK4 and 5-iteration ADM and MADM is also performed to evaluate the accuracy and convergence behaviour of both iterative methods. The Michaelis Menten system is selected as a benchmark nonlinear problem due to its inherent stiffness and strong coupling between the dependent variables.

PIM:

According to the PIM framework, the iterative scheme for the Michaelis Menten system (6.69)-(6.70) can be expressed as follows:

$$x_{(n+1)}(t) = x(0) + \int_0^t \Big(-x_n(s) + (\beta - \alpha) y_n(s) + x_n(s) y_n(t) \Big)\, ds, \tag{6.72}$$

$$y_{(n+1)}(t) = y(0) + \int_0^t \Big(\frac{1}{\varepsilon} x_n(s) - \beta y_n(s) - x_n(s) y_n(s) \Big)\, ds. \tag{6.73}$$

By substituting the initial conditions in (6.71) and the parameter values $\alpha = 0.375$, $\beta = 1.0$, and $\varepsilon = 0.1$ into equations (6.72) and (6.73), the following expressions are obtained.

For $n = 0$:

$$x_1(t) = 1 - t, \tag{6.74}$$

$$y_1(t) = 10t. \tag{6.75}$$

The iterative process was continued up to $n = 4$, at which point convergence was achieved when the tolerance criterion $\varepsilon < 10^{-2}$ was satisfied. Due to the extensive nature of the polynomial expansions, only the first two iterations ($n = 0$ and $n = 1$) and the last iteration ($n = 4$) are presented here, while the complete series are provided in the Appendix H.

MPIM:

The MPIM is now applied to the Michaelis–Menten system of equations (6.69) and (6.70). This method enhances the conventional PIM by dividing the solution domain $[0, T]$ into multiple subintervals $[t_{j-1}, t_j]$, where $j = 1, 2, \ldots, m$. Each subinterval serves as a correction stage that improves numerical stability and accelerates convergence, particularly for stiff nonlinear systems.

According to the MPIM formulation, the iterative scheme for the Michaelis–-Menten system is expressed as:

$$x_{(n+1),j}(t) = x_{m,j-1}(t_{j-1}) + \int_{t_{j-1}}^{t} \Big(- x_{n,j-1}(s) + (\beta - \alpha) y_{n,j-1}(s) + x_{n,j-1}(s) \, y_{n,j-1}(s) \Big) ds, \tag{6.76}$$

$$y_{(n+1),j}(t) = y_{m,j-1}(t_{j-1}) + \int_{t_{j-1}}^{t} \Big(\frac{1}{\varepsilon} x_{n,j-1}(s) - \beta y_{n,j-1}(s) - x_{n,j-1}(s) y_{n,j-1}(s) \Big) ds. \tag{6.77}$$

By substituting the initial condition in (6.71) and the parameter values $\alpha = 0.375$, $\beta = 1.0$, and $\varepsilon = 0.1$ into equations (6.76) and (6.77), the following expressions are obtained for the fifth iteration ($n = 4$) with stage increments $j = 1$ and $j = 100$.

For $n = 5$, $j = 1$:

$$x_{5,1}(t) = 1 - t + 8.625t^2 - 63.08333t^3 + 373.17969t^4 - 1979.37552t^5 + 5662.80990t^6 - 27572.95430t^7 + 1.15513 \times 10^5 t^8 - 4.09745 \times 10^5 t^9 + 1.27591 \times 10^6 t^{10} + \ldots - 9.10109 \times 10^3 t^{31} \quad (6.78)$$

$$y_{5,1}(t) = 10t - 105t^2 + 762.08333t^3 - 4446.25t^4 + 23128.44271t^5 - 61211.82292t^6 + 2.92297 \times 10^5 t^7 - 1.19956 \times 10^6 t^8 + 4.16425 \times 10^6 t^9 - 1.28975 \times 10^7 t^{10} + \ldots + 9.10109 \times 10^4 t^{31}. \quad (6.79)$$

The iterative process was continued up to $n = 4$, $j = 100$. Given the extensive nature of the iterative expressions, only the representative iterations ($n = 4, j = 1$) is presented in this section, while the complete series are provided in Appendix H.

6.4.2 Results and discussion

Figures 6.12 and 6.13 show the solutions for x and y, respectively, obtained by the 5-iteration of PIM, the 5-iteration of MPIM ($\Delta t = 0.01$), 5-term ADM, 5-term MADM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$) for the case $\alpha = 0.375$, $\beta = 1.0$, $\varepsilon = 0.1$ and the initial condition in (6.71). It is observed that the 5-iteration MPIM solutions agree very well with the RK4 and 5-term ADM, 5-term MADM ($\Delta t = 0.01$) solutions for t up to $t = 1$, while the 5-iteration PIM solution diverges immediately after $t = 0$.

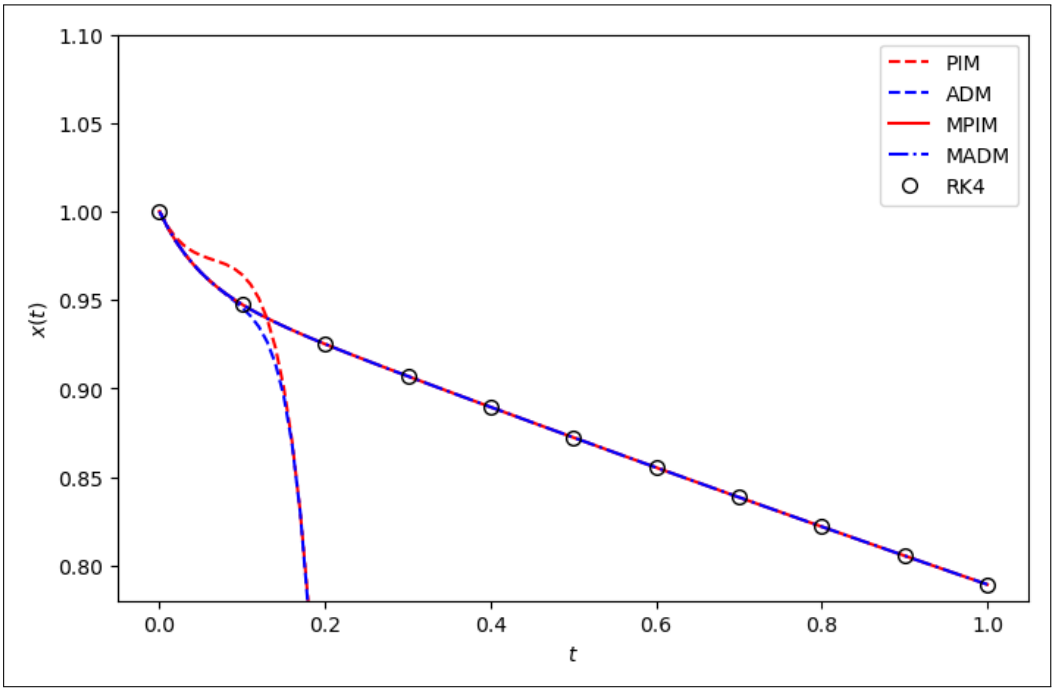

Figure 6.12 Solution of $x(t)$ using 5-iteration PIM, 5-iteration MPIM ($\Delta t = 0.01$), 5-term ADM, 5-term MADM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$).

Figure 6.12 shows the numerical behaviour of the substrate concentration $x(t)$ for the Michaelis-Menten model obtained using the 5-iteration of PIM, the 5-iteration of MPIM, the 5-iteration of ADM, the 5-iteration of MADM, and RK4 methods over the interval $0 \leq t \leq 1$. From the figure, the RK4 solution shows a gradual decrease in $x(t)$, where the substrate concentration reduces from its initial value and approaches approximately $x(1) \approx 0.79$. This decreasing behaviour is consistent with the expected reaction process, as the substrate is consumed over time. The MPIM and MADM solutions follow the RK4 profile closely throughout the considered interval, indicating that the multistage implementation is able to preserve the smooth decreasing trend of the reference solution. In contrast, the classical PIM and ADM solutions are only consistent with the RK4 solution at the very early stage. The PIM curve begins to deviate shortly after the initial time, while the ADM solution shows a sharp downward behaviour and fails to maintain the gradual decreasing pattern observed in RK4. This suggests that the classical non-multistage methods are less suitable for representing the solution over the full interval. Overall, the graphical result indicates that the multistage approaches, namely MPIM and MADM, provide a more consistent numerical behaviour than their corresponding classical forms for

solving the Michaelis-Menten model.

Table 6.19
Comparison of Absolute Errors for $x(t)$ in the Michaelis–Menten Model Between RK4, 5-iteration PIM, 5-iteration MPIM, 5-iteration ADM, and 5-iteration MADM.

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
0.1	1.2528×10^{-1}	2.2160×10^{-6}	1.9882×10^{-2}	6.1686×10^{-7}
0.2	1.0173×10^{1}	5.8785×10^{-7}	3.7953×10^{0}	9.9402×10^{-8}
0.3	1.0314×10^{2}	1.2171×10^{-7}	7.7581×10^{1}	8.7247×10^{-9}
0.4	3.2629×10^{2}	2.3771×10^{-8}	6.4463×10^{2}	5.6877×10^{-9}
0.5	3.3367×10^{3}	5.3230×10^{-9}	3.2896×10^{3}	8.0017×10^{-9}
0.6	2.3871×10^{5}	2.0215×10^{-9}	1.2367×10^{4}	8.4136×10^{-9}
0.7	5.4781×10^{6}	1.4470×10^{-9}	3.7708×10^{4}	8.5240×10^{-9}
0.8	7.2488×10^{7}	1.3494×10^{-9}	9.8724×10^{4}	8.5852×10^{-9}
0.9	6.7377×10^{8}	1.3338×10^{-9}	2.3020×10^{5}	8.6372×10^{-9}
1.0	4.8230×10^{9}	1.3322×10^{-9}	4.9008×10^{5}	8.6861×10^{-9}

Table 6.19 presents the comparison of absolute errors for $x(t)$ in the Michaelis-Menten model between the RK4 reference solution and the 5-iteration PIM, 5-iteration MPIM, 5-iteration ADM, and 5-iteration MADM over the interval $0 \leq t \leq 1$. Based on the results, the classical PIM and ADM show increasing absolute errors as time progresses. For PIM, the error increases from 1.2528×10^{-1} at $t = 0.1$ to 4.8230×10^{9} at $t = 1.0$, while ADM increases from 1.9882×10^{-2} to 4.9008×10^{5} over the same interval. This indicates that the classical non-multistage methods gradually depart from the RK4 solution, which is consistent with the graphical behaviour where PIM and ADM fail to maintain the smooth decreasing trend of $x(t)$ over the full interval. In comparison, MPIM produces much smaller errors, with values decreasing from 2.2160×10^{-6} at $t = 0.1$ to approximately 1.3322×10^{-9} at $t = 1.0$. MADM also gives small displayed errors, remaining within the order of 10^{-7} to 10^{-9} across the selected time points. These results suggest that the multistage implementations reduce the discrepancy from RK4 compared with their corresponding classical methods.

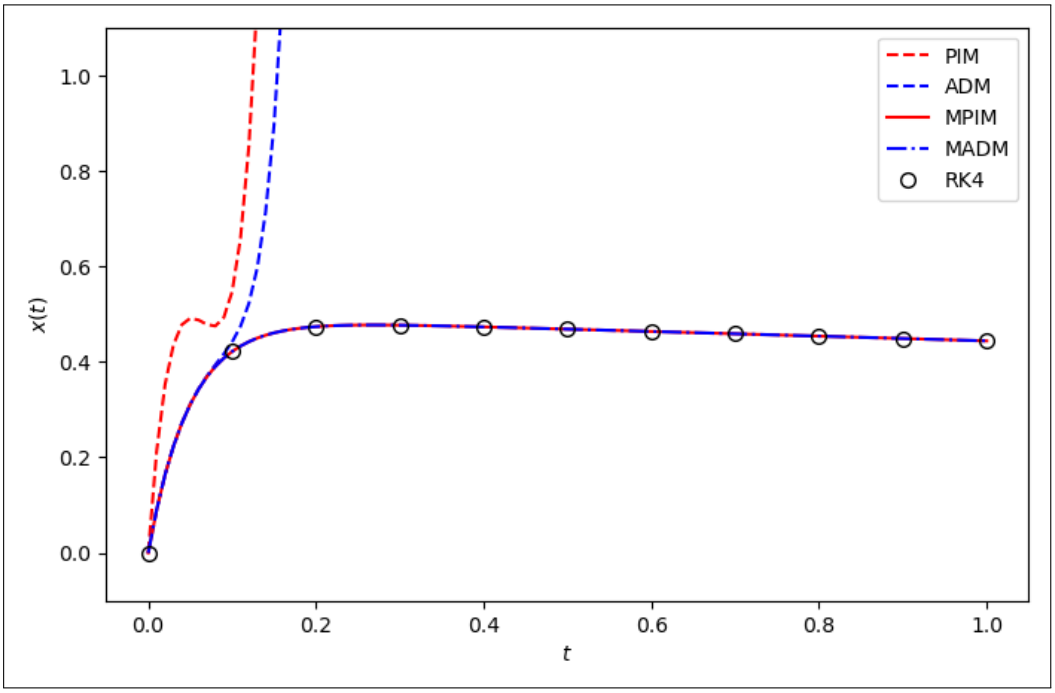

Figure 6.13 Solution of $y(t)$ using 5-iteration PIM, 5-iteration MPIM ($\Delta t = 0.01$), 5-term ADM, 5-term MADM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$).

Figure 6.13 illustrates the numerical behaviour of $y(t)$ for the Michaelis-Menten model computed using the 5-iteration of PIM, the 5-iteration of MPIM, the 5-iteration of ADM, the 5-iteration of MADM, and RK4 over the interval $0 \leq t \leq 1$. From the figure, the RK4 solution shows that $y(t)$ initially increases rapidly from zero and then changes slowly after approximately $t = 0.2$, approaching a nearly steady trend towards the end of the interval. The MPIM and MADM solutions follow the RK4 profile closely throughout the simulation, indicating that the multistage implementation is able to preserve the overall behaviour of $y(t)$. In contrast, the classical PIM and ADM solutions agree with RK4 only at the very early stage before deviating noticeably. The PIM curve rises sharply and moves away from the reference solution, while ADM also fails to maintain the gradual behaviour observed in RK4. This suggests that the classical non-multistage methods are less consistent for representing $y(t)$ over the full interval. Overall, the graphical result indicates that MPIM and MADM provide more consistent numerical behaviour than PIM and ADM for this component of the Michaelis-Menten model.

Table 6.20
Comparison of Absolute Errors for $y(t)$ in the Michaelis–Menten Model Between RK4, 5-iteration PIM, 5-iteration MPIM, 5-iteration ADM, and 5-iteration MADM.

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
0.1	1.2528×10^{-1}	2.2160×10^{-6}	1.9882×10^{-2}	6.1686×10^{-7}
0.2	1.0173×10^{1}	5.8785×10^{-7}	3.7953×10^{0}	9.9402×10^{-8}
0.3	1.0314×10^{2}	1.2171×10^{-7}	7.7581×10^{1}	8.7247×10^{-9}
0.4	3.2629×10^{2}	2.3771×10^{-8}	6.4463×10^{2}	5.6877×10^{-9}
0.5	3.3367×10^{3}	5.3230×10^{-9}	3.2896×10^{3}	8.0017×10^{-9}
0.6	2.3871×10^{5}	2.0215×10^{-9}	1.2367×10^{4}	8.4136×10^{-9}
0.7	5.4781×10^{6}	1.4470×10^{-9}	3.7708×10^{4}	8.5240×10^{-9}
0.8	7.2488×10^{7}	1.3494×10^{-9}	9.8724×10^{4}	8.5852×10^{-9}
0.9	6.7377×10^{8}	1.3338×10^{-9}	2.3020×10^{5}	8.6372×10^{-9}
1.0	4.8230×10^{9}	1.3322×10^{-9}	4.9008×10^{5}	8.6861×10^{-9}

Table 6.20 presents the comparison of absolute errors for $y(t)$ in the Michaelis-Menten model between the RK4 reference solution and the 5-iteration PIM, 5-iteration MPIM, 5-iteration ADM, and 5-iteration MADM over the interval $0 \leq t \leq 1$. Based on the results, the classical PIM and ADM show increasing errors as time progresses. For PIM, the error increases from 1.2528×10^{-1} at $t = 0.1$ to 4.8230×10^{9} at $t = 1.0$, while ADM increases from 1.9882×10^{-2} to 4.9008×10^{5} over the same interval. This indicates that the classical non-multistage methods gradually depart from the RK4 solution, which is consistent with the graphical observation where PIM and ADM fail to maintain the expected behaviour of $y(t)$ over the full interval. In comparison, MPIM gives much smaller errors, with values decreasing from 2.2160×10^{-6} at $t = 0.1$ to 1.3322×10^{-9} at $t = 1.0$. MADM also shows small displayed errors, remaining within the order of 10^{-7} to 10^{-9} across the selected time points. Overall, the absolute error results suggest that the multistage implementations reduce the discrepancy from RK4 compared with their corresponding classical forms.

6.5 Multispecies Lotka-Volterra Equations

In this section, consider multispecies Lotka-Volterra equations as a model problem to solve the system of ODEs (3.65). A physical chemist by the name of Alfred James Lotka developed the notion of an evolutionary system based on two fundamental changes, those involving matter between components of a system and

those involving exchange of energy (Kingsland, 1995). Unlike those grounded in chemistry, Lotka believed that these ideas could be applied to any biological system. Around the same time as Lotka's book, "Elements of Mathematical Biology," was published, Volterra developed the well-known mathematical model of multispecies interaction. These models, the predator, prey, and competition models, are known today as Lotka-Volterra models. Although simplistic, these models are still used as the foundation for mathematical models in ecology. For more information on Lotka-Volterra models, see Leung (2013). The dynamic behavior of an arbitrary number of competitors can be modelled by the so-called Lotka-Volterra equations (Hofbauer & Sigmund, 1988). These models describe the time history of a biological system and are used in such various fields as engineering, chemistry and biology. In fact, the one-predator, one-prey Lotka-Volterra model is one of the most popular ones to demonstrate a simple nonlinear control system.

Finding accurate solutions to the Lotka-Volterra equations is difficult when the equations are stiff, whether due to a small number of species or a large number of species (Olek, 1994). Unlike the discrete solutions obtained by the purely numerical methods like the RK4, approximate analytical solutions can increase our insight into the natural behavior of complex systems.

The aim of this section is to apply PIM and MPIM to solve the system of ODEs considering the multispecies Lotka-Volterra equations as a model problem. The numerical solutions are compared to the exact solutions and those obtained using the RK4 method.

6.5.1 Analysis of multispecies Lotka–Volterra equations

Consider the general Lotka–Volterra model for an m species system given as:

$$x_i' = x_i \left(b_i + \sum_{j=1}^{m} a_{ij} x_j \right), \quad i = 1, 2, \ldots, m, \tag{6.80}$$

where the prime denotes differentiation with respect to t. These equations may represent either predator–prey or competition cases.

One Species

Now the PIM and MPIM will apply to solve the one-species Lotka-Volterra equation. In the one-species case, equation (6.80) reduces to one-species (Batiha et al., 2007):

$$\frac{dy}{dt} = y(\beta + \alpha y), \quad \beta > 0, \quad \alpha < 0, \quad y(0) > 0, \tag{6.81}$$

where α and β are constants. This equation has an exact solution

$$y(t) = \begin{cases} \dfrac{\beta e^{\beta t}}{\beta + \alpha y(0) - \alpha y(0) e^{\beta t}}, & \text{for } \beta \neq 0, \\ \dfrac{y(0)}{1 - \alpha y(0) t}, & \text{for } \beta = 0, \end{cases} \tag{6.82}$$

where the initial condition

$$y(0) = 0.1, \tag{6.83}$$

and the parameters

$$\alpha = -3 \quad \text{and} \quad \beta = 1. \tag{6.84}$$

Two Species

In the case of two species, equation (6.80) of the multispecies Lotka–Volterra model simplifies as follows (Batiha et al., 2007):

$$\frac{dx}{dt} = x\left(\beta_1 + \alpha_{11} x + \alpha_{12} y\right), \tag{6.85}$$

$$\frac{dy}{dt} = y\left(\beta_2 + \alpha_{21} x + \alpha_{22} y\right), \tag{6.86}$$

where α_{11}, α_{12}, α_{21}, α_{22}, β_1, and β_2 are constans and the initial condition is given as follows

$$x(0) = 4 \quad \text{and} \quad y(0) = 10. \tag{6.87}$$

Three Species

For the three-species case, the multispecies Lotka–Volterra model in equation (6.80) reduces to the following form (Batiha et al., 2007):

$$\frac{dx}{dt} = x(1 - x - \alpha y - \beta z), \tag{6.88}$$

$$\frac{dy}{dt} = y(1 - \beta x - y - \alpha z), \tag{6.89}$$

$$\frac{dz}{dt} = z(1 - \alpha x - \beta y - z), \tag{6.90}$$

where α and β are constants. The initial condition for three species are given by

$$x(0) = 0.2, \quad y(0) = 0.3, \quad z(0) = 0.5. \tag{6.91}$$

6.5.2 PIM and MPIM solution

One Species

In this section, the one–species Lotka–Volterra model represented by equation (6.81) is solved using the standard PIM and MPIM. This system is selected as a test problem to evaluate the accuracy and convergence properties of the PIM and MPIM when applied to nonlinear population dynamics. The model describes the self–interaction of a single species and can exhibit exponential or oscillatory behaviour depending on the growth rate and interaction parameters.

PIM:

According to the PIM framework, the iterative scheme for the one–species Lotka–Volterra system (6.81) can be formulated as follows:

$$y_{(n+1)}(t) = y(0) + \int_0^t y_n(s)\big(\beta + \alpha y_n(s)\big)\, ds, \tag{6.92}$$

where $y_n(t)$ represents the n-th approximation at time t, while α and β denote the interaction and intrinsic growth parameters, respectively.

By substituting the initial condition in (6.83) and the parameter values in (6.84) into equation (6.92), the following expressions are obtained.

For $n = 0$:

$$y_1(t) = 0.1 + 0.07t. \tag{6.93}$$

The iterative process was continued up to $n = 5$, at which point convergence was achieved when the tolerance criterion $\varepsilon < 10^{-10}$ was satisfied. Due to the extensive nature of the polynomial expansions, only the first iterations ($n = 0$ is presented here, while the complete series are provided in Appendix I.

MPIM:

According to the MPIM formulation, the iterative scheme for the one–species Lotka–Volterra system (6.81) is expressed as:

$$y_{(n+1),j}(t) = y_{m,j-1}(t_{j-1}) + \int_0^t y_{n,j-1}(s)\big(\beta + \alpha y_{n,j-1}(s)\big)\, ds. \tag{6.94}$$

By substituting the initial condition in (6.83) and the parameter values (6.84) into equations (6.94), the following expressions are obtained for the six iteration ($n = 5$).

For $n = 5$, $j = 1$:

$$\begin{aligned} y_{6,1} = {} & 0.1 + 0.07t + 0.0280t^2 - 0.0035t^3 - 0.0132t^4 - 0.0061t^5 + 0.0037t^6 \\ & + 0.0059t^7 + 0.0032t^8 - 0.0059t^9 - 0.0028t^{10} + 0.0023t^{11} + 0.0023t^{12} \\ & - 0.0003t^{13} - 0.0013t^{14} - 0.0003t^{15} + \ldots - 6.8218 \times 10^{-23} t^{63} \end{aligned} \tag{6.95}$$

The iterative process was continued up to $n = 5$, $j = 700$. Given the extensive nature of the iterative expressions, only the representative iterations ($n = 5, j = 1$) is presented in this section, while the complete series are provided in Appendix I.

Two Species

In this section, the two–species Lotka–Volterra model represented by equation (6.85) and (6.86) is solved using the standard PIM and MPIM. This system

is selected as a test problem to evaluate the accuracy and convergence properties of the PIM and MPIM when applied to nonlinear population dynamics.

PIM:

According to the PIM framework in (3.21), the solution of system (6.85) and (6.86) are constructed iteratively as:

$$x_{(n+1)}(t) = x(0) + \int_0^t x_n(s)\left(\beta_1 + \alpha_{11}x_n(s) + \alpha_{12}y_n(s)\right)ds, \tag{6.96}$$

$$y_{(n+1)}(t) = y(0) + \int_0^t y_n(s)\left(\beta_2 + \alpha_{21}x_n(s) + \alpha_{22}y_n(s)\right)ds. \tag{6.97}$$

By substituting the parameters $\alpha_{11} = -0.0014$, $\alpha_{12} = -0.0012$, $\alpha_{21} = -0.0009$, $\alpha_{22} = -0.001$, $\beta_1 = 0.1$, $\beta_2 = 0.08$ and the initial condition define in (6.87) into equation (6.96) and (6.97). The approximate solution of PIM are

For $n = 0$:

$$x_1(t) = 4 + 0.3296t, \tag{6.98}$$

$$y_1(t) = 10 + 0.664t. \tag{6.99}$$

The iterative process was continued up to $n = 4$, at which point convergence was achieved when the tolerance criterion $\varepsilon < 10^{-10}$ was satisfied. Due to the extensive nature of the polynomial expansions, only the first iterations ($n = 0$ is presented here, while the complete series are provided in Appendix I.

MPIM:

According to the MPIM formulation, the solution of system (6.85) and (6.86) are constructed iteratively as:

$$x_{(n+1),j}(t) = x_{m,j-1}(t_{j-1}) + \int_{tj-1}^{t} x_{n,j-1}(s)\left(\beta_1 + \alpha_{11}x_{n,j-1}(s) + \alpha_{12}y_{n,j-1}(s)\right)ds, \tag{6.100}$$

$$y_{(n+1),j}(t) = y_{m,j-1}(t_{j-1}) + \int_{t_{j-1}}^{t} y_{n,j-1}(s)\left(\beta_2 + \alpha_{21}x_{n,j-1}(s) + \alpha_{22}y_{n-j-1}(s)\right)ds. \tag{6.101}$$

By substituting the parameters $\alpha_{11} = -0.0014$, $\alpha_{12} = -0.0012$, $\alpha_{21} = -0.0009$, $\alpha_{22} = -0.001$, $\beta_1 = 0.1$, $\beta_2 = 0.08$ and the initial condition (6.87) into the equation (6.100) and (6.101). The following is the approximate solution for two species by MPIM

For $n = 4$, $j = 1$:

$$\begin{aligned} x_{5,1}(t) = {} & 4 + 0.32960t + 0.022126t^2 + 0.0011190t^3 + 2.8937 \times 10^{-5}t^4 - 1.6604 \\ & \times 10^{-6}t^5 - 8.8496 \times 10^{-7}t^6 + 1.3310 \times 10^{-8}t^7 + 2.0992 \times 10^{-9}t^8 \\ & + 5.6296 \times 10^{-11}t^9 - 3.2582 \times 10^{-12}t^{10} - 3.6280 \times 10^{-13}t^{11} + 2.7715 \\ & \times 10^{-15}t^{12} + 7.6902 \times 10^{-16}t^{13} + 1.9528 \times 10^{-17}t^{14} - 1.6489 \times 10^{-18}t^{15} \\ & - 5.4384 \times 10^{-20}t^{16} + \ldots - 1.8265 \times 10^{-45}t^{31}, \end{aligned} \tag{6.102}$$

$$\begin{aligned} y_{5,1}(t) = {} & 10.000 + 0.66400t + 0.034483t^2 + 0.0011079t^3 - 1.6834 \times 10^{-5}t^4 \\ & - 5.5542 \times 10^{-6}t^5 - 5.6323 \times 10^{-7}t^6 + 2.2259 \times 10^{-8}t^7 + 1.7422 \times 10^{-9} \\ & t^8 + 1.8320 \times 10^{-11}t^9 - 3.7820 \times 10^{-12}t^{10} - 2.3633 \times 10^{-13}t^{11} + 5.1442 \\ & \times 10^{-15}t^{12} + 5.8237 \times 10^{-16}t^{13} + 7.5587 \times 10^{-18}t^{14} - 1.2768 \times 10^{-18}t^{15} \\ & - 2.8666 \times 10^{-20}t^{16} + \ldots - 1.1259 \times 10^{-45}t^{31}. \end{aligned} \tag{6.103}$$

The iterative process was continued up to $n = 4$, $j = 7000$. Given the extensive nature of the iterative expressions, only the representative iterations ($n = 4, j = 1$) is presented in this section, while the complete series are provided in Appendix I.

Three Species

In this section, the three–species Lotka–Volterra model represented by equation (6.88)-(6.90) is solved using the standard PIM and MPIM. This system is selected as a test problem to evaluate the accuracy and convergence properties of the PIM and MPIM when applied to nonlinear population dynamics. Each species interacts competitively through the parameters α and β, which represent inter–species and intra–species interaction strengths, respectively.

PIM:

According to the PIM framework, the iterative scheme for the three–species Lotka–Volterra system (6.88)–(6.90) can be formulated as follows:

$$x_{(n+1)}(t) = x(0) + \int_0^t x_n(s)\big(1 - x_n(s) - \alpha y_n(s) - \beta z_n(s)\big)\, ds, \tag{6.104}$$

$$y_{(n+1)}(t) = y(0) + \int_0^t y_n(s)\big(1 - \beta x_n(s) - y_n(s) - \alpha z_n(s)\big)\, ds, \tag{6.105}$$

$$z_{(n+1)}(t) = z(0) + \int_0^t z_n(s)\big(1 - \alpha x_n(s) - \beta y_n(s) - z_n(s)\big)\, ds. \tag{6.106}$$

By substituting the parameter values $\alpha = 0.1$, $\beta = 0.1$, and the initial conditions in (6.91) into equations (6.104)–(6.106), the system can be evaluated iteratively to obtain successive approximations for each species. The approximate series solutions of the PIM are given as follows:

For $n = 0$:

$$x_1(t) = 0.2 + 0.144t, \tag{6.107}$$

$$y_1(t) = 0.3 + 0.189t, \tag{6.108}$$

$$z_1(t) = 0.5 + 0.225t. \tag{6.109}$$

The iterative process was continued up to $n = 5$, at which point convergence was achieved when the tolerance criterion $\varepsilon < 10^{-10}$ was satisfied. For brevity, only the first iterations ($n = 0$) is presented here, while the complete series are provided in the Appendix I.

MPIM:

According to the MPIM formulation, the solution of system (6.88)–(6.90) are constructed iteratively as:

$$x_{(n+1),j}(t) = x_{m,j-1}(t_{j-1}) + \int_{t_{j-1}}^{t} x_{n,j-1}(s)\big(1 - x_{n,j-1}(s) - \alpha y_{n,j-1}(s) - \beta z_{n,j-1}(s)\big)\, ds, \tag{6.110}$$

$$y_{(n+1),j}(t) = y_{m,j-1}(t_{j-1}) + \int_{t_{j-1}}^{t} y_{n,j-1}(s)\big(1 - \beta x_{n,j-1}(s) - y_n(s) - \alpha z_{n,j-1}(s)\big)\, ds, \tag{6.111}$$

$$z_{(n+1),j}(t) = z_{m,j-1}(t_{j-1}) + \int_{t_{j-1}}^{t} z_{n,j-1}(s)\big(1 - \alpha x_{n,j-1}(s) - \beta y_{n,j-1}(s) - z_{n,j-1}(s)\big)\, ds. \tag{6.112}$$

By substituting the parameter values $\alpha = 0.1$, $\beta = 0.1$, and the initial conditions in (6.91) into equations (6.110)–(6.112), the system can be evaluated iteratively to obtain successive approximations for each species. The approximate series solutions of the MPIM are given as follows:

For $n = 5$, $j = 1$:

$$\begin{aligned} x_{6,1}(t) = {} & 0.2 + 0.14400t + 0.066600t^2 + 0.0074664t^3 - 0.016601t^4 - 0.014108t^5 \\ & - 0.0027107t^6 + 0.0025892t^7 + 0.012768t^8 - 0.0036717t^9 - 0.0054828t^{10} \\ & + 8.9753 \times 10^{-5} t^{11} + 0.0020422t^{12} + 5.0772 \times 10^{-4} t^{13} - 2.6117 \times 10^{-4} t^{14} \\ & - 8.3005 \times 10^{-4} t^{15} + 4.3179 \times 10^{-4} t^{16} + \ldots - 2.0093 \times 10^{-22} t^{63}, \end{aligned} \tag{6.113}$$

$$\begin{aligned} y_{6,1}(t) = {} & 0.3 + 0.18900t + 0.051300t^2 - 0.026927t^3 - 0.029247t^4 - 2.1770 \times 10^{-4} t^5 \\ & + 0.014715t^6 + 0.0073370t^7 - 0.0042379t^8 - 0.010299t^9 + 0.0027818t^{10} \\ & + 0.0057168t^{11} - 6.4521 \times 10^{-4} t^{12} - 0.0029271t^{13} + 1.3015 \times 10^{-4} t^{14} \\ & + 2.5977 \times 10^{-4} t^{15} + 0.0021796t^{16} + \ldots - 3.0360 \times 10^{-21} t^{63}, \end{aligned} \tag{6.114}$$

$$z_{6,1}(t) = 0.5 + 0.22500t - 0.027900t^2 - 0.062618t^3 + 0.014936t^4 + 0.031797t^5 - 0.0098742t^6 - 0.018431t^7 + 0.0070230t^8 + 0.012578t^9 - 0.0071008t^{10} - 0.0084889t^{11} + 0.0051920t^{12} + 0.0046929t^{13} - 0.0034072t^{14} - 0.0019947t^{15} + 0.0011691t^{16} + \ldots - 6.4464 \times 10^{-20} t^{63}. \quad (6.115)$$

The iterative process was continued up to $n = 5$, $j = 500$. Given the extensive nature of the iterative expressions, only the representative iterations ($n = 5, j = 1$) is presented in this section, while the complete series are provided in Appendix I.

6.5.3 Results and discussion

One Species

Figure 6.14 shows the solutions for $y(t)$, respectively, obtained by the 6-iteration of PIM, the 6-iteration of MPIM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$) for the case $\alpha = -3$, and $\beta = 1$, and the initial condition in (6.83). Table 6.21 shows the comparison between the RK4 on the time step $\Delta t = 0.01$ and 6-iteration of PIM and 6-iteration of MPIM on the time step $\Delta t = 0.01$ for the case $\alpha = -3$, and $\beta = 1$, and the initial condition in (6.83). It is observed that the 6-iteration of MPIM solutions agree very well with the RK4 solutions for t up to $t = 6$, while the 6-iteration of PIM solutions are only valid for $t \leq 4$.

Figure 6.14 illustrates the numerical behaviour of $y(t)$ for the one-species Lotka-Volterra model computed using the exact solution, 6-iteration PIM, 6-iteration MPIM, 6-iteration ADM, 6-iteration MADM, and RK4 over the interval $0 \leq t \leq 7$. Based on the figure, the exact solution and RK4 show that $y(t)$ increases gradually from the initial value and approaches a nearly steady level after approximately $t = 5$. The MPIM and MADM solutions show good graphical agreement with both the exact solution and RK4 throughout most of the interval, indicating that the multistage implementation is able to preserve the overall increasing and stabilising behaviour of the solution. In contrast, the classical PIM and ADM solutions only follow the reference behaviour for a limited time. The ADM curve begins to deviate after the early stage and decreases sharply before reaching the later part of the interval, while the PIM curve remains close for a longer time but eventually moves away from the

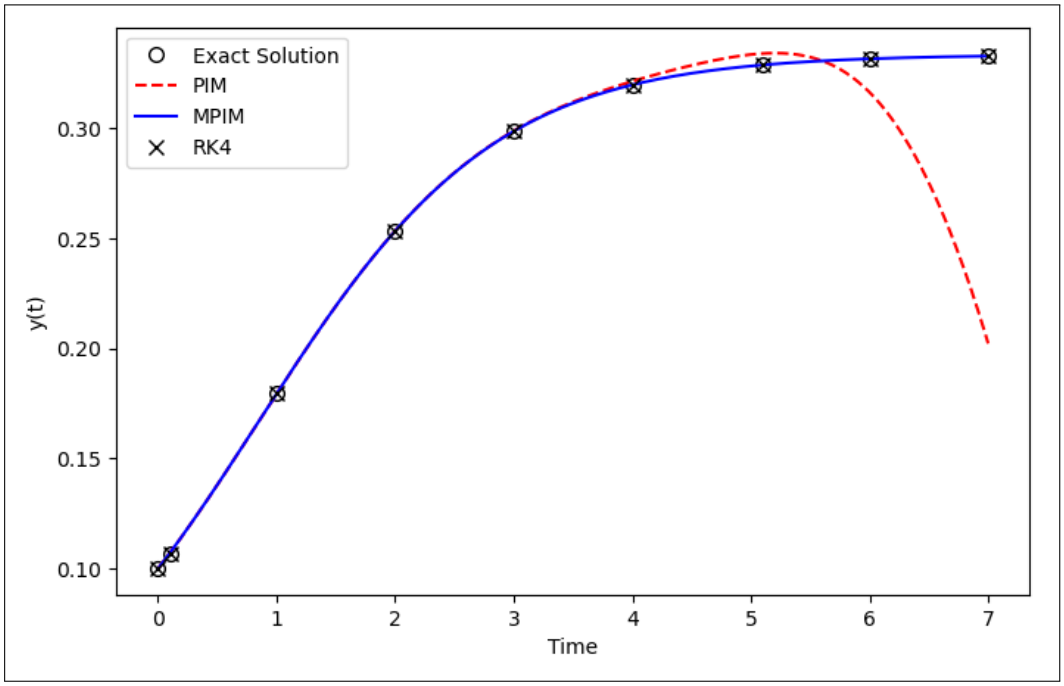

Figure 6.14 Solution of $y(t)$ using 6-iteration PIM, 6-iteration MPIM($\Delta t = 0.01$), 6-iteration ADM, 6-iteration MADM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$)

reference solution near the later stage. This suggests that the non-multistage methods are less consistent for long-time approximation of this nonlinear model. Overall, the graphical result indicates that MPIM and MADM provide more consistent numerical behaviour than the classical PIM and ADM for the one-species Lotka-Volterra model.

Table 6.21
Comparison of Absolute Errors for the One-Species Lotka-Volterra Model Between the Exact Solution, 6-iteration PIM, 6-iteration MPIM, 6-iteration ADM, 6-iteration MADM, and RK4.

Time	|Exact-PIM|	|Exact-MPIM|	|Exact-ADM|	|Exact-MADM|	|Exact-RK4|
1	1.5108×10^{-10}	6.9487×10^{-5}	1.7851×10^{-4}	8.1507×10^{-12}	7.5020×10^{-12}
2	1.6916×10^{-7}	4.3119×10^{-5}	1.0124×10^{-2}	9.4423×10^{-12}	8.2640×10^{-12}
3	7.7335×10^{-5}	1.2603×10^{-4}	8.9314×10^{-2}	3.9987×10^{-11}	4.1674×10^{-11}
4	1.5070×10^{-3}	1.0969×10^{-4}	3.6856×10^{-1}	3.1980×10^{-11}	3.0495×10^{-11}
5	5.1546×10^{-3}	6.6689×10^{-5}	1.0303×10^{0}	4.0903×10^{-11}	3.9953×10^{-11}
6	1.5245×10^{-2}	3.4446×10^{-5}	2.2971×10^{0}	1.4021×10^{-10}	1.4072×10^{-10}
7	1.3041×10^{-1}	1.6319×10^{-5}	4.4333×10^{0}	2.8862×10^{-11}	2.9109×10^{-11}

Table 6.21 presents the comparison of absolute errors for the one-species Lotka-Volterra model between the exact solution and the numerical solutions obtained using the 6-iteration PIM, 6-iteration MPIM, 6-iteration ADM, 6-iteration MADM, and RK4 over the interval $0 \leq t \leq 7$. Based on the results, the ADM errors increase

noticeably as time progresses, from 1.7851×10^{-4} at $t = 1$ to 4.4333×10^{0} at $t = 7$, which is consistent with the graphical result where ADM deviates from the reference solution after the early stage. The PIM errors remain small at the beginning, but increase at later times, reaching 1.3041×10^{-1} at $t = 7$. In comparison, MPIM gives relatively small errors across the selected time points, with values remaining within the order of 10^{-5} to 10^{-4}. Meanwhile, MADM and RK4 show very small displayed errors, mostly within the order of 10^{-12} to 10^{-10}, indicating close agreement with the exact solution under the selected computational setting. Overall, the error results suggest that the multistage methods, particularly MADM, reduce the discrepancy from the exact solution compared with the corresponding classical methods.

Two Species

Figure 6.15 - 6.16 shows the solutions for x and y, respectively, obtained by the 5-iteration of PIM, the 5-iteration of MPIM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$) for the case $\alpha_{11} = -0.0014$, $\alpha_{12} = -0.0012$, $\alpha_{21} = -0.0009$, $\alpha_{22} = -0.001$, $\beta_1 = 0.1$, $\beta_2 = 0.08$ and the initial condition in (6.87). It is observed that the 5-iteration of MPIM solutions agree very well with the RK4 solutions for t up to $t = 70$, while the 5-iteration of PIM solutions are only valid for $t \leq 50$.

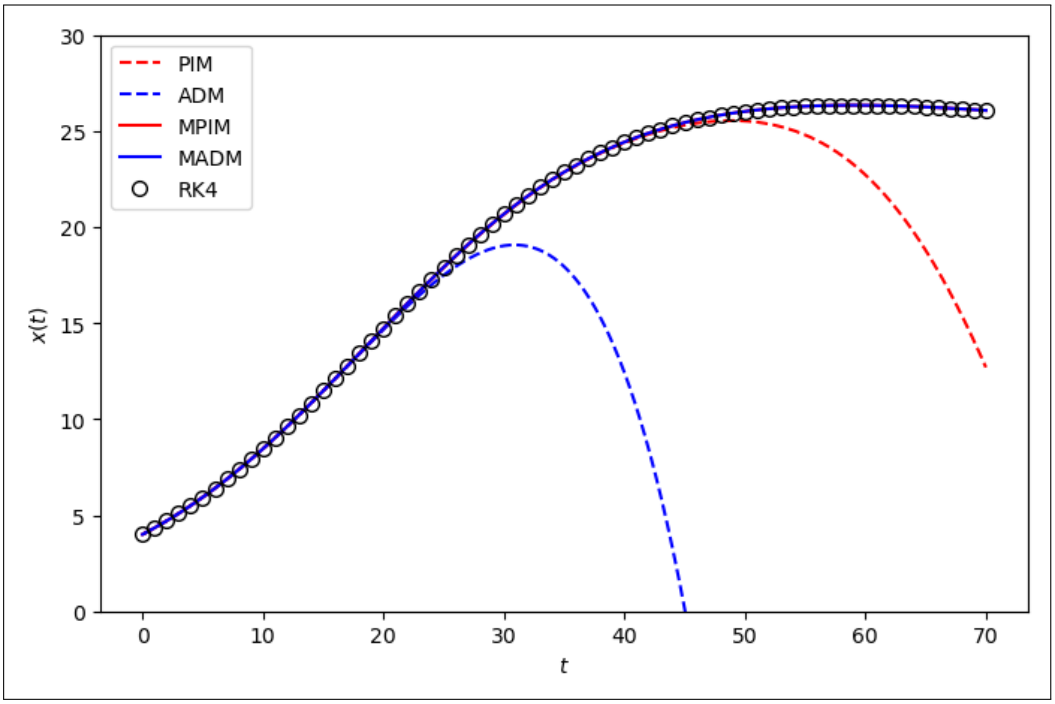

Figure 6.15 Solution of $x(t)$ using 5-iteration PIM, 5-iteration of ADM, 5-iteration MPIM ($\Delta t = 0.01$), 5-iteration of MADM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$) for the Two Species Lotka-Volterra Equations

Figure 6.15 illustrates the numerical behaviour of $x(t)$ for the two-species Lotka-Volterra model computed using the 5-iteration of PIM, the 5-iteration of MPIM, the 5-iteration of ADM, the 5-iteration of MADM, and RK4 over the interval $0 \leq t \leq 70$. Based on the figure, the RK4 solution shows that $x(t)$ increases gradually from the initial stage, reaches a maximum level around $t = 55$, and then remains almost steady with a slight decrease towards the end of the interval. The MPIM and MADM solutions follow the RK4 profile closely throughout the simulation, indicating that the multistage methods are able to capture the increasing and stabilising behaviour of $x(t)$. In contrast, the classical PIM and ADM solutions only follow the RK4 trend for

a limited time before deviating noticeably. The ADM solution begins to decrease sharply after the middle of the interval, while the PIM solution also moves away from the RK4 curve at the later stage. This suggests that the classical non-multistage methods are less consistent for the long-time behaviour of this model. Overall, the graphical result suggests that MPIM and MADM provide more consistent numerical behaviour than the classical PIM and ADM for $x(t)$ in the two-species Lotka-Volterra model.

Table 6.22
Comparison of Absolute Errors for $x(t)$ in the Two-Species Lotka-Volterra Model Between RK4, 5-iteration PIM, 5-iteration MPIM, 5-iteration ADM, and 5-iteration MADM.

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
10	3.4440×10^{-4}	1.5109×10^{-3}	5.2068×10^{-4}	1.5987×10^{-14}
20	6.2301×10^{-3}	2.4601×10^{-3}	6.2861×10^{-2}	6.0396×10^{-14}
30	1.8500×10^{-2}	1.5170×10^{-3}	1.6534×10^{0}	1.5987×10^{-13}
40	3.8252×10^{-2}	2.6447×10^{-4}	1.2005×10^{1}	3.9080×10^{-14}
50	5.0106×10^{-1}	1.4472×10^{-4}	4.7966×10^{1}	3.9790×10^{-13}
60	3.6254×10^{0}	3.4269×10^{-5}	1.3713×10^{2}	2.2808×10^{-12}
70	1.3379×10^{1}	1.7350×10^{-4}	3.1904×10^{2}	9.6279×10^{-13}

Table 6.22 presents the comparison of absolute errors for $x(t)$ in the two-species Lotka-Volterra model between the RK4 reference solution and the 5-iteration PIM, 5-iteration MPIM, 5-iteration ADM, and 5-iteration MADM over the interval $0 \leq t \leq 70$. Based on the results, the ADM error increases noticeably as time progresses, from 5.2068×10^{-4} at $t = 10$ to 3.1904×10^{2} at $t = 70$. This is consistent with the graphical result, where the ADM solution begins to deviate from the RK4 curve after the middle of the interval. The PIM error remains relatively small at earlier times, but increases at later times, reaching 1.3379×10^{1} at $t = 70$. In comparison, MPIM produces smaller errors than PIM and ADM for most of the selected time points, with values remaining within the order of 10^{-5} to 10^{-3}. Meanwhile, MADM shows the smallest displayed errors, with values mostly within the order of 10^{-14} to 10^{-12}, suggesting very small discrepancy from RK4 under the selected computational setting. Overall, the absolute error results suggest that the multistage methods reduce the difference from RK4 compared with the corresponding classical methods, particularly for the long-time behaviour of $x(t)$.

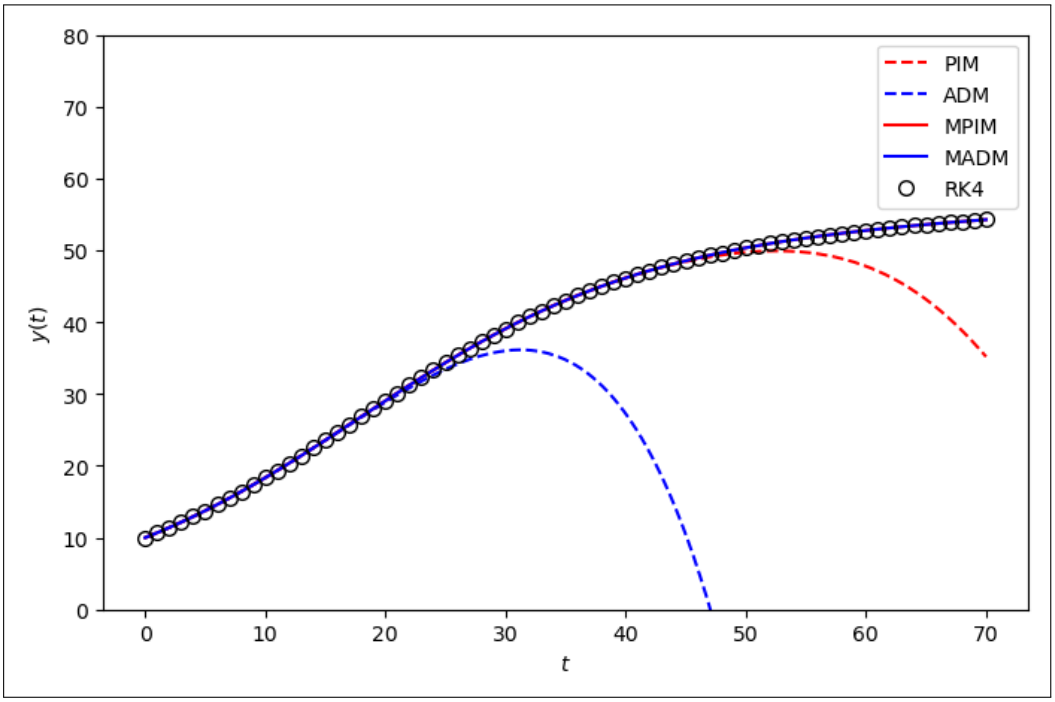

Figure 6.16 Solution of $y(t)$ using 5-iteration PIM, 5-iteration of ADM, 5-iteration MPIM ($\Delta t = 0.01$), 5-iteration of MADM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$) for the Two Species Lotka-Volterra Equations

Figure 6.16 illustrates the numerical behaviour of $y(t)$ for the two-species Lotka-Volterra model computed using the 5-iteration of PIM, the 5-iteration of MPIM, the 5-iteration of ADM, the 5-iteration of MADM, and RK4 over the interval $0 \leq t \leq 70$. From the figure, the RK4 solution shows that $y(t)$ increases gradually from the initial stage and approaches a higher population level towards the end of the interval. The MPIM and MADM solutions follow the RK4 profile closely throughout the simulation, indicating that the multistage methods are able to preserve the increasing behaviour of $y(t)$. In contrast, the classical PIM and ADM solutions follow the RK4 trend only for a limited time before deviating at later times. The ADM curve begins to decrease sharply after the middle of the interval, while the PIM curve also moves away from the RK4 solution near the later stage. This suggests that the classical non-multistage methods are less consistent for long-time simulation of $y(t)$. Overall, the graphical result suggests that MPIM and MADM provide more consistent numerical behaviour than PIM and ADM for $y(t)$ in the two-species Lotka-Volterra model.

Table 6.23
Comparison of Absolute Errors for $y(t)$ in the Two-Species Lotka–Volterra Model between RK4, 5-iteration PIM, 5-iteration MPIM, 5-iteration ADM, and 5-iteration MADM.

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
10	3.6923×10^{-5}	1.9598×10^{-3}	1.4113×10^{-3}	$< 1.0 \times 10^{-16}$
20	3.9928×10^{-3}	2.5571×10^{-3}	1.9554×10^{-1}	8.1712×10^{-14}
30	1.7950×10^{-2}	7.5304×10^{-4}	3.0728×10^{0}	3.4106×10^{-13}
40	1.5449×10^{-2}	1.0882×10^{-3}	1.8895×10^{1}	1.2079×10^{-13}
50	6.1894×10^{-1}	1.6644×10^{-3}	6.9892×10^{1}	1.5632×10^{-13}
60	4.9781×10^{0}	1.4957×10^{-3}	1.9140×10^{2}	1.2790×10^{-13}
70	1.9085×10^{1}	1.1689×10^{-3}	4.3324×10^{2}	1.1511×10^{-12}

Note: The value $< 1.0 \times 10^{-16}$ indicates that the displayed error is below the numerical precision used in Maple with `Digits := 16`. It should not be interpreted as an exact zero error.

Table 6.23 presents the comparison of absolute errors for $y(t)$ in the two-species Lotka-Volterra model between the RK4 reference solution and the 5-iteration PIM, 5-iteration MPIM, 5-iteration ADM, and 5-iteration MADM over the interval $0 \leq t \leq 70$. Based on the results, the ADM error increases noticeably as time progresses, from 1.4113×10^{-3} at $t = 10$ to 4.3324×10^{2} at $t = 70$, which is consistent with the graphical result where ADM deviates from the RK4 curve after the middle of the interval. The PIM error remains relatively small at the earlier time points, but increases at later times, reaching 1.9085×10^{1} at $t = 70$. In comparison, MPIM gives smaller errors than PIM and ADM for most of the selected time points, with values remaining within the order of 10^{-3} to 10^{-4}. Meanwhile, MADM shows the smallest displayed errors in the table, mostly within the order of 10^{-14} to 10^{-12}. The value reported as $< 1.0 \times 10^{-16}$ at $t = 10$ does not indicate an exact zero error, but represents an error below the displayed numerical precision used in the computation. Overall, the absolute error results suggest that the multistage methods reduce the discrepancy from RK4 compared with the corresponding classical methods for $y(t)$.

Three Species

Figure 6.17 - 6.19 shows the solutions for x and y, respectively, obtained by the 6-iteration of PIM, the 6-iteration of MPIM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$) for the case $\alpha = 0.1$, $\beta = 0.1$, and the initial condition in (6.91). It is observed that the

6-iteration of MPIM solutions agree very well with the RK4 solutions for t up to $t = 5$, while the 6-iteration of PIM solutions are only valid for $t \leq 4$.

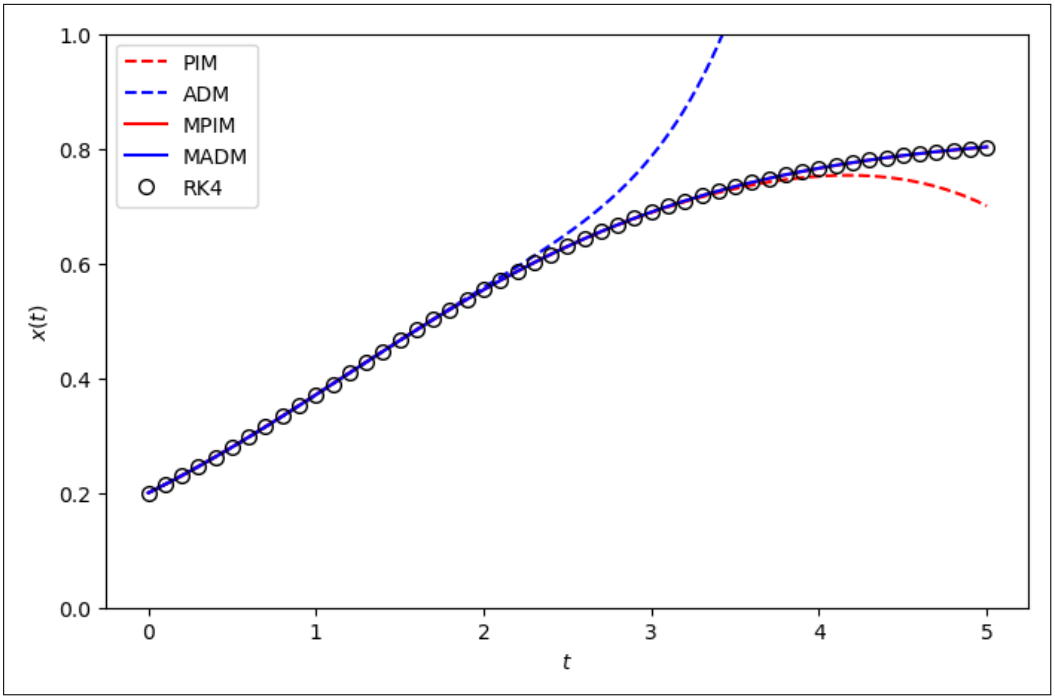

Figure 6.17 Solution of $x(t)$ using 6-iteration PIM, 6-iteration MPIM ($\Delta t = 0.01$), 6-iteration ADM, 6-iteration MADM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$).

Figure 6.17 illustrates the numerical behaviour of $x(t)$ for the three-species Lotka-Volterra model computed using the 6-iteration of PIM, the 6-iteration of MPIM, the 6-iteration of ADM, the 6-iteration of MADM, and RK4 over the interval $0 \leq t \leq 5$. Based on the figure, the RK4 solution shows that $x(t)$ increases gradually from the initial value and continues to rise until the end of the simulation interval. The MPIM and MADM solutions show good graphical agreement with the RK4 solution, as both methods follow the same increasing trend throughout the interval. In contrast, the classical PIM and ADM solutions begin to deviate from the RK4 reference solution at later times. The PIM solution remains close to the reference curve at the early stage but slightly departs near the end of the interval, while the ADM solution increases more rapidly after approximately $t = 2.5$ and moves away from the RK4 trend. This suggests that the classical non-multistage methods are less consistent for representing the long-time behaviour of $x(t)$ in this model. Overall, the graphical result suggests that the multistage approaches provide more consistent numerical behaviour than the classical PIM and ADM for $x(t)$ in the three-species Lotka-Volterra model.

Table 6.24
Comparison of Absolute Errors for $x(t)$ in the Three-Species Lotka-Volterra Model Between RK4, 6-iteration PIM, 6-iteration MPIM, 6-iteration ADM, and 6-iteration MADM.

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
1	3.1654×10^{-8}	2.7066×10^{-4}	1.1916×10^{-5}	1.9123×10^{-12}
2	7.8652×10^{-6}	1.6699×10^{-4}	3.7183×10^{-3}	2.9873×10^{-12}
3	7.5219×10^{-4}	9.9334×10^{-5}	9.6807×10^{-2}	3.6942×10^{-12}
4	1.4094×10^{-2}	2.0389×10^{-4}	8.8601×10^{-1}	3.3942×10^{-12}
5	1.0277×10^{-1}	1.7568×10^{-4}	4.6281×10^{0}	2.4025×10^{-12}

Table 6.24 presents the comparison of absolute errors for $x(t)$ in the three-species Lotka-Volterra model between the RK4 reference solution and the 6-iteration PIM, 6-iteration MPIM, 6-iteration ADM, and 6-iteration MADM over the interval $0 \leq t \leq 5$. Based on the results, the ADM error increases as time progresses, from 1.1916×10^{-5} at $t = 1$ to 4.6281×10^{0} at $t = 5$. This is consistent with the graphical result, where the ADM solution begins to move away from the RK4 curve at later times. The PIM error is very small at the early time points, but increases to 1.0277×10^{-1} at $t = 5$, showing some deviation near the end of the interval. In comparison, MPIM gives relatively small errors throughout the selected time points, with values remaining within the order of 10^{-4}. Meanwhile, MADM gives the smallest displayed errors, with values around the order of 10^{-12}, indicating a very small discrepancy from RK4 under the selected computational setting. Overall, the absolute error results suggest that the multistage methods reduce the difference from RK4 compared with the corresponding classical methods for $x(t)$.

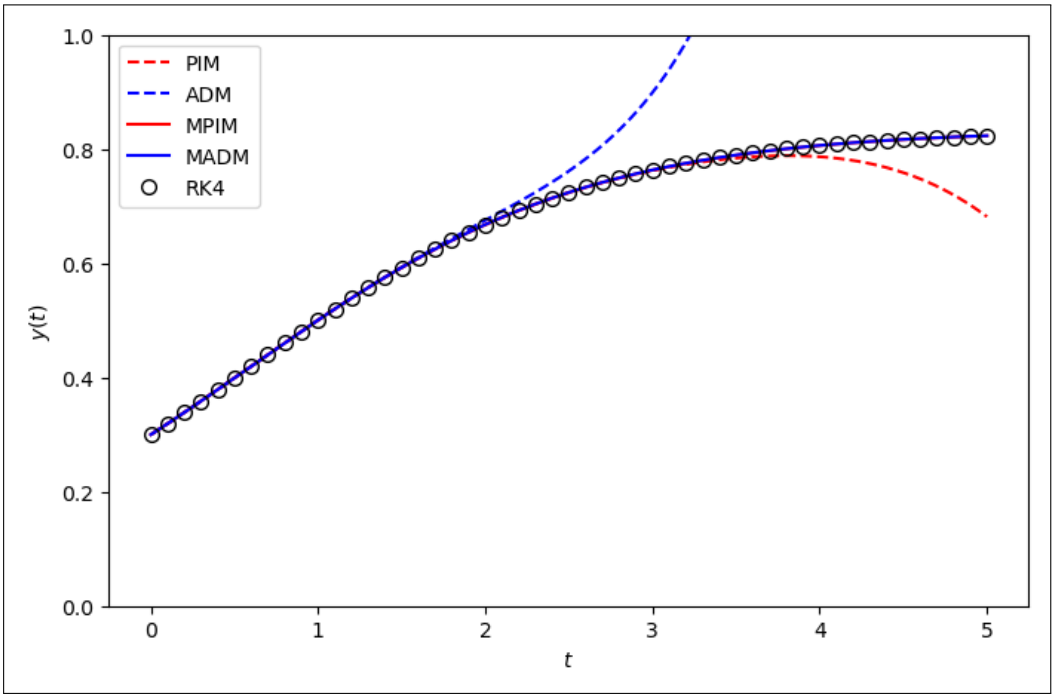

Figure 6.18 Solution of $y(t)$ using 6-iteration PIM, 6-iteration MPIM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$).

Figure 6.18 illustrates the numerical behaviour of $y(t)$ for the three-species Lotka-Volterra model computed using the 6-iteration of PIM, the 6-iteration of MPIM, the 6-iteration of ADM, the 6-iteration of MADM, and RK4 over the interval $0 \leq t \leq 5$. Based on the figure, the RK4 solution shows that $y(t)$ increases gradually from the initial value and approaches a higher level towards the end of the interval. The MPIM and MADM solutions show good graphical agreement with the RK4 solution, as both methods follow the increasing trend throughout most of the simulation interval. In contrast, the classical PIM and ADM solutions begin to deviate from the RK4 reference solution at later times. The PIM curve remains close to the RK4 solution at the early stage but slightly moves away near the end of the interval, while the ADM curve increases more rapidly after approximately $t = 2.5$ and does not maintain the same trend as RK4. This suggests that the classical non-multistage methods are less consistent in capturing the later behaviour of $y(t)$. Overall, the graphical result suggests that the multistage approaches provide more consistent numerical behaviour than the classical PIM and ADM for $y(t)$ in the three-species Lotka-Volterra model.

Table 6.25
Comparison of Absolute Errors for $y(t)$ in the Three-Species Lotka-Volterra Model Between RK4, 6-iteration PIM, 6-iteration MPIM, 6-iteration ADM, and 6-iteration MADM.

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
1	6.4290×10^{-10}	4.2537×10^{-5}	3.5437×10^{-5}	1.2377×10^{-12}
2	1.4125×10^{-5}	2.3477×10^{-4}	7.3144×10^{-3}	2.7849×10^{-12}
3	1.1435×10^{-3}	3.5225×10^{-4}	1.3554×10^{-1}	3.7086×10^{-12}
4	1.9853×10^{-2}	2.7200×10^{-4}	9.3805×10^{-1}	3.0935×10^{-12}
5	1.4181×10^{-1}	1.5909×10^{-4}	3.8600×10^{0}	1.9440×10^{-12}

Table 6.25 presents the comparison of absolute errors for $y(t)$ in the three-species Lotka-Volterra model between the RK4 reference solution and the 6-iteration PIM, 6-iteration MPIM, 6-iteration ADM, and 6-iteration MADM over the interval $0 \leq t \leq 5$. Based on the results, the ADM error increases as time progresses, from 3.5437×10^{-5} at $t = 1$ to 3.8600×10^{0} at $t = 5$, which is consistent with the graphical result where the ADM solution moves away from the RK4 curve at later times. The PIM error is very small at the early stage, but increases to 1.4181×10^{-1} at $t = 5$, showing some deviation near the end of the interval. In comparison, MPIM gives relatively small errors throughout the selected time points, with values remaining within the order of 10^{-5} to 10^{-4}. Meanwhile, MADM shows the smallest displayed errors, with values around the order of 10^{-12}, indicating a very small discrepancy from RK4 under the selected computational setting. Overall, the absolute error results suggest that the multistage methods reduce the difference from RK4 compared with the corresponding classical methods for $y(t)$.

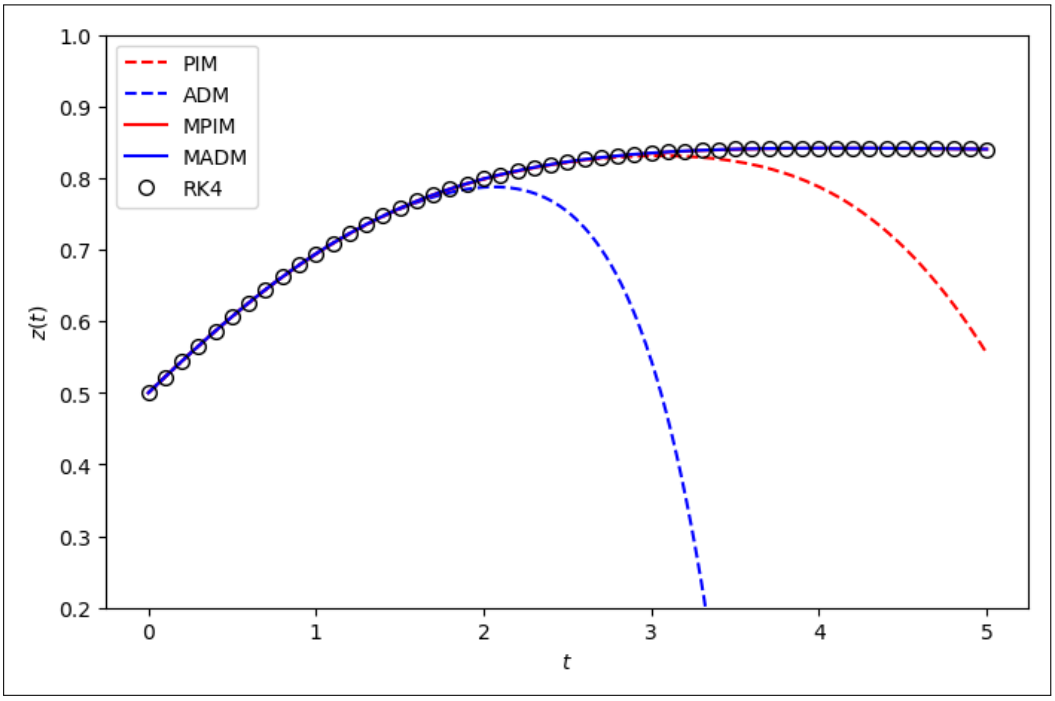

Figure 6.19 Solution of $z(t)$ using 6-iteration PIM, 6-iteration MPIM ($\Delta t = 0.01$) and RK4 ($\Delta t = 0.01$).

Figure 6.19 illustrates the numerical behaviour of $z(t)$ for the three-species Lotka-Volterra model computed using the 6-iteration of PIM, the 6-iteration of MPIM, the 6-iteration of ADM, the 6-iteration of MADM, and RK4 over the interval $0 \leq t \leq 5$. Based on the figure, the RK4 solution shows that $z(t)$ increases gradually from the initial value and approaches a nearly steady level after approximately $t = 3$. The MPIM and MADM solutions show good graphical agreement with the RK4 solution, as both methods follow the increasing and stabilising behaviour of $z(t)$ throughout most of the interval. In contrast, the classical PIM and ADM solutions begin to deviate from the RK4 reference solution at later times. The PIM solution remains close to the RK4 curve at the early stage but moves downward near the end of the interval, while the ADM solution decreases sharply after approximately $t = 2.5$ and fails to maintain the stabilising trend shown by RK4. This suggests that the classical non-multistage methods are less consistent in capturing the later behaviour of $z(t)$. Overall, the graphical result suggests that the multistage approaches provide more consistent numerical behaviour than the classical PIM and ADM for $z(t)$ in the three-species Lotka-Volterra model.

Table 6.26
Comparison of Absolute Errors for $z(t)$ in the Three-Species Lotka-Volterra Model Between RK4, 6-iteration PIM, 6-iteration MPIM, 6-iteration ADM, and 6-iteration MADM.

Time	|RK4-PIM|	|RK4-MPIM|	|RK4-ADM|	|RK4-MADM|
1	3.0435×10^{-8}	3.2778×10^{-4}	4.1285×10^{-5}	1.9590×10^{-12}
2	6.8578×10^{-5}	5.0296×10^{-4}	1.1818×10^{-2}	4.1440×10^{-12}
3	4.0904×10^{-3}	3.6507×10^{-4}	2.9039×10^{-1}	4.0819×10^{-12}
4	5.4238×10^{-2}	1.8002×10^{-4}	2.6173×10^{0}	2.6975×10^{-12}
5	2.8530×10^{-1}	6.4656×10^{-5}	1.3813×10^{1}	1.4007×10^{-12}

Table 6.26 presents the comparison of absolute errors for $z(t)$ in the three-species Lotka-Volterra model between the RK4 reference solution and the 6-iteration PIM, 6-iteration MPIM, 6-iteration ADM, and 6-iteration MADM over the interval $0 \leq t \leq 5$. Based on the results, the ADM error increases noticeably as time progresses, from 4.1285×10^{-5} at $t = 1$ to 1.3813×10^{1} at $t = 5$. This supports the graphical observation where the ADM solution deviates from the RK4 curve at later times. The PIM error is very small at the early time point, but increases to 2.8530×10^{-1} at $t = 5$, indicating some loss of agreement near the end of the interval. In comparison, MPIM gives relatively small errors across the selected time points, with values remaining within the order of 10^{-4}. Meanwhile, MADM shows the smallest displayed errors, with values around the order of 10^{-12}, suggesting a very small discrepancy from RK4 under the selected computational setting. Overall, the absolute error results suggest that the multistage methods reduce the difference from RK4 compared with the corresponding classical methods for $z(t)$.

6.6 Concluding Remarks

This chapter investigated the applicability of the classical PIM and the proposed MPIM for solving several nonlinear systems of ordinary differential equations. Particular emphasis was given to the Monkeypox Transmission Model, which involves strongly coupled human and rodent compartments and represents a challenging nonlinear epidemiological system. The results show that the classical PIM is only reliable over a limited time interval, as its approximation deteriorates and becomes unstable when the solution is extended over a long computational domain.

In contrast, the MPIM remains stable throughout the simulation interval by applying the Picard iteration successively over smaller subintervals.

For the Monkeypox Transmission Model, the numerical solutions obtained using PIM, MPIM, ADM, and MADM were compared with both the RK4 and the RK8. The inclusion of RK8 provides an additional higher-order numerical reference, whereas RK4 is retained because it is commonly used in related Monkeypox modelling studies. The comparisons with both RK4 and RK8 consistently show that the multistage methods, particularly MPIM and MADM, provide substantially more stable approximations than the corresponding classical PIM and ADM methods. Although the numerical differences relative to RK4 and RK8 vary across the compartments and time points, the overall results confirm that the multistage implementation effectively controls the error growth and instability observed in the standard iterative methods. In addition, the comparison with the available infected human data indicates that MPIM, MADM, RK4, and RK8 are able to capture the general trend of the infected population more reliably than PIM and ADM.

The performance of PIM and MPIM was further assessed using the COVID-19 model, the Michaelis–Menten biochemical reaction system, and one-, two-, and three-species Lotka–Volterra models. Across these nonlinear systems, the classical PIM generally provides acceptable approximations during the early stages of the simulation but shows increasing deviation as the time interval becomes longer. Conversely, MPIM consistently improves the long-term numerical behaviour of the standard PIM and remains close to the RK4 reference solution. Where available, ADM and MADM were also included as additional semi-analytical benchmarks. The results demonstrate that the multistage approaches provide more reliable solutions than their respective classical forms, with MADM often producing the smallest displayed errors relative to RK4 or to the exact solution in the one-species Lotka–Volterra problem.

Overall, the findings establish that the multistage strategy is effective in extending the applicability of the classical PIM to nonlinear systems with long computational intervals, strong coupling, and complex dynamical behaviour. While PIM remains useful as a simple semi-analytical approach for short-time approximation, MPIM provides a more stable and reliable framework for long-term

simulation. Therefore, this chapter supports the use of MPIM as an improved Picard-based method for solving nonlinear systems of ordinary differential equations and provides a basis for its further application to more complex mathematical models.

CHAPTER 7
CONCLUSION AND RECOMMENDATIONS

The primary aim of this thesis was to develop and evaluate three enhanced Picard-based numerical methods, namely the Multistage Picard Iterative Method (MPIM), the Picard Iterative Method for *n*th order problems (PIM*n*), and the Multistage Picard Iterative Method for *n*th order problems (MPIM*n*). The motivation for this work stemmed from the limited long-time reliability of the classical Picard Iterative Method (PIM) and its direct formulation for first-order differential equations. To address these limitations, this thesis proposed MPIM by incorporating a multistage time-marching structure to improve long-time numerical behaviour. In addition, PIM*n* and MPIM*n* were developed to extend the classical Picard-based framework for the direct solution of higher-order differential equations without reducing them into equivalent first-order systems.

The first major application of the PIM and MPIM in this thesis involved the Monkeypox transmission model. The standard PIM produced reliable approximations only over short computational intervals but showed increasing deviations over longer simulations due to error accumulation. In contrast, MPIM improved the long-time approximation by updating the solution across successive subintervals. To provide a broader comparison, ADM and MADM were also included as supplementary semi-analytical methods. The numerical results of PIM, MPIM, ADM, and MADM were compared with RK4 and RK8, where RK8 was additionally employed as a higher-order numerical benchmark because the Monkeypox model involved an extended simulation interval and a large number of computational stages.

To further validate the PIM and MPIM framework, the methods were applied to additional nonlinear systems, including the COVID-19 model, the biochemical reaction model (Michaelis–Menten system), and the multispecies Lotka–Volterra equations. For each application, ADM and MADM were included as supplementary semi-analytical comparison methods, while RK4 was used as the numerical benchmark. The numerical findings showed that MPIM generally produced more reliable long-time approximations than the classical PIM. The inclusion of ADM and

MADM provided an additional comparative basis for evaluating the behaviour of the proposed Picard-based approach across biological, epidemiological, and ecological systems.

In chapter 1, the introduction to the study was presented, outlining the research motivation, problem statement, research objectives, research questions, and the overall structure of the thesis.

In Chapter 2, a systematic review of the literature was conducted, covering the background of Monkeypox disease, mathematical modelling of infectious diseases, and the development of the Picard Iterative Method and its modifications. The chapter critically examined existing numerical approaches and identified their limitations in terms of stability, convergence, and applicability to higher order and long-term simulations. These findings established the research gap addressed in this thesis, particularly the limited application of Picard-based methods in epidemiological modelling and direct higher order differential equations.

In chapter 3, the mathematical development of the proposed MPIM, PIM*n* and MPIM*n* methods was detailed. This chapter presented the derivations, iterative formulations, convergence analysis, and implementation strategies that formed the foundation for applying these methods to nonlinear ODEs and PDEs. The chapter established how MPIM enhances global accuracy and how the nth-order formulation allows PIM to be applied directly to higher order systems.

In Chapter 4, the discussion began with the formulation of the PIM*n* method for solving nth-order initial value problems directly. This formulation provided the basis for extending the classical Picard Iterative Method to higher order differential equations without first reducing them into systems of first order equations. The next section presented the application of PIM and MPIM to the Riccati equation, where the numerical results showed that MPIM gave more reliable approximations than PIM over the considered interval. The following section then examined the nonlinear second-order Duffing equation using PIM*n* and MPIM*n*. The results indicated that PIM*n* could produce acceptable approximations over shorter intervals, but its accuracy became less consistent as the simulation interval increased. Finally, the Pohlhausen flow equation was considered to further evaluate the performance of PIM*n* and MPIM*n* for a higher order nonlinear problem. Overall, the findings suggest that the multistage

modification helps improve the stability and accuracy of the Picard-based approach, particularly for longer simulation intervals, while still depending on the problem type, time step size, and number of iterations used.

In Chapter 5, the PIM and MPIM were applied to nonlinear partial differential equations through the Newell Whitehead Segel equation. The application demonstrated how the Picard-based iterative framework can be implemented for a nonlinear PDE problem while preserving the iterative structure of the original method. The numerical results showed that both PIM and MPIM were able to approximate the solution over the considered interval. However, MPIM generally produced closer agreement with the benchmark solution and smaller absolute errors compared to the standard PIM. This suggests that the multistage modification improves the accuracy and stability of the Picard-based approach for the nonlinear PDE problem considered in this study, particularly when the solution is evaluated over a longer simulation interval.

In Chapter 6, the PIM and MPIM were applied to nonlinear systems of ODEs, including the Monkeypox transmission model, COVID-19 model, biochemical reaction model, and multispecies Lotka-Volterra equations. To provide a broader semi-analytical comparison, ADM and MADM were also applied to all four systems. For the COVID-19, biochemical reaction, and multispecies Lotka-Volterra models, the numerical approximations were compared with the RK4 benchmark. For the Monkeypox transmission model, both RK4 and RK8 were included, with RK8 serving as an additional higher-order numerical benchmark for the extended simulation interval. The results indicated that MPIM generally produced more stable and accurate approximations than the classical PIM over the considered time intervals. In several cases, the standard PIM showed increasing deviations as the simulation time increased, whereas MPIM maintained closer agreement with the numerical benchmarks. The ADM and MADM results further provided supplementary evidence for assessing the relative numerical behaviour of the proposed multistage Picard-based approach.

Overall, the results of this thesis indicate that MPIM and MPIM*n* provide useful extensions of the classical Picard-based approach for solving the selected linear and nonlinear differential equation models considered in this study. Both

methods preserve the iterative structure of the classical PIM and can be implemented directly in Maple without requiring linearisation of the governing equations. Across the tested ODE and PDE models, including epidemiological systems, chemical kinetics, ecological interactions, nonlinear oscillators, and boundary-layer flow problems, MPIM generally produced more accurate and stable approximations than the classical PIM over the considered simulation intervals. Meanwhile, PIM*n* extended the applicability of the classical Picard approach to higher order differential equations and showed satisfactory performance over shorter time intervals, although its accuracy became less consistent in longer simulations. In comparison, MPIM*n* maintained smaller absolute errors and closer agreement with the exact or RK4 benchmark solutions for the higher order problems investigated. These findings suggest that the multistage Picard-based methods have potential as practical numerical approaches for solving selected differential equation models in applied sciences, particularly when longer simulation intervals are required.

7.1 Recommendations

The results of this thesis suggest several promising avenues for further research. First, although the MPIM has demonstrated strong accuracy for nonlinear ODEs and PDEs, future studies could extend the method to more complex classes of differential equations such as fractional differential equations, delay differential equations, and stochastic differential equations, which frequently arise in epidemiological modelling, biological systems, and financial dynamics.

Second, the PIM*n* and MPIM*n* developed in this thesis can be further enhanced. Future research may also extend the PIM*n* and MPIM*n* framework to fractional-order differential equations and delay differential systems. Many modern applications in physics, biology, and finance involve memory-dependent or hereditary processes that cannot be adequately described by classical integer-order models. Extending the PIM*n* and MPIM*n* to such generalized systems would broaden its applicability and contribute to the advancement of iterative-based numerical solvers for complex dynamical systems.

Third, the PIM*n* and MPIM*n* may also be generalised to handle variable-order or mixed-order differential equations, and it may be coupled with adaptive step-size

techniques or embedded error estimators to improve robustness when applied to highly nonlinear or rapidly oscillatory systems.

Fourth, beyond Maple, future work could focus on implementing MPIM, PIM*n* and MPIM*n* in Python, MATLAB, or Julia, providing wider accessibility and enabling large-scale simulations or integration into existing computational platforms.

Fifth, the methods introduced in this thesis can be extended to multi-dimensional PDEs, such as higher-dimensional reaction–diffusion equations, Navier–Stokes-type fluid models, or coupled multiphysics systems. Investigating how the methods behave in 2D and 3D settings would deepen understanding of their scalability and stability.

Finally, future research should include a more rigorous theoretical convergence and stability analysis for the MPIM, PIM*n* and MPIM*n*. While the numerical experiments conducted here strongly support their effectiveness, a formal mathematical theory would further strengthen the foundation and reliability of these methods for broader scientific applications.

REFERENCES

Abbass, G., Chen, H., Abdullahi, M., Sani Halilu, A., & Ahmed Memon, I. (2025). Hybrid accelerated double step length approach for convex constrained nonlinear monotone equations with applications. *IMA Journal of Applied Mathematics*, *90*(1), 75–98.

Abdel-Gawad, H., El-Shrae, A., & Saad, K. (2017). Criterion of existence of realistic permanent travelling wave solutions in reaction diffusion systems. *Communications in Numerical Analysis*, *2017*(1), 50–79.

Abed, S. M., & AL-Jawary, M. A. (2021). Efficient iterative methods for solving the sir epidemic model. *Iraqi Journal of Science*, 613–622.

Adomian, G. (1994). Modified decomposition. In *Solving frontier problems of physics: The decomposition method* (pp. 115–153). Springer.

Adom-Konadu, A., Bonyah, E., Sackitey, A. L., Anokye, M., & Asamoah, J. K. K. (2023). A fractional order monkeypox model with protected travelers using the fixed point theorem and newton polynomial interpolation. *Healthcare Analytics*, *3*, 100191.

Agbata, B. C., Dervishi, R., Agbebaku, D. F., Cenaj, E., Collins, O. C., Ezeafulukwe, A. U., Shior, M. M.-A., & Mbah, G. C. E. (2025). A comprehensive analysis of fractional-order model of tuberculosis with treatment intervention. *BMC Infectious Diseases*, *25*(1), 1070.

Agom, E., Ogunfiditimi, F., & Bassey, E. V. (2017). Multistage adomian decomposition method for nonlinear 4th order multi-point boundary value problems. *Global Journal of Mathematics Vol*, *10*(2).

Ahmad, J., Ullah, K., Arshad, M., & de la Sen, M. (2021). Approximation of fixed points for mean nonexpansive mappings in banach spaces. *Journal of Function Spaces*, *2021*(1), 1934274.

Akkilic, A. N., Sabir, Z., Bhat, S. A., & Bulut, H. (2024). A radial basis deep neural network process using the bayesian regularization optimization for the monkeypox transmission model. *Expert Systems with Applications*, *235*, 121257.

Alaje, A. I., & Olayiwola, M. O. (2023). A fractional-order mathematical model for examining the spatiotemporal spread of covid-19 in the presence of vaccine distribution. *Healthcare Analytics*, *4*, 100230.

Alalhareth, F. K., Alharbi, M. H., Laksaci, N., Boudaoui, A., & Medjoudja, M. (2024). Investigating a fractal–fractional mathematical model of the third wave of covid-19 with vaccination in saudi arabia. *Fractal and Fractional*, *8*(2), 95.

Alharbi, R., Jan, R., Alyobi, S., Altayeb, Y., & Khan, Z. (2022). Mathematical modeling and stability analysis of the dynamics of monkeypox via fractional-calculus. *Fractals*, *30*(10), 2240266.

Almeida, R., Malinowska, A. B., & Monteiro, M. T. T. (2018). Fractional differential equations with a caputo derivative with respect to a kernel function and their applications. *Mathematical Methods in the Applied Sciences*, *41*(1), 336–352.

Ann, A. Y. A. B. D., & Samarji, J. A. M. A. (2009). A multistage adomian decomposition method for solving the autonomous van der pol system. *Australian Journal of Basic and Applied Sciences*, *3*(4), 4397–4407.

Ansari, S., & Akrami, M. H. (2024). An effective approach for solving nonlinear fractional initial value problems: The fractional legendre-picard iteration method. *Journal of Mathematical Extension*, *18*.

Ariffin, M. R. K., Gopal, K., Krishnarajah, I., Che Ilias, I. S., Adam, M. B., Arasan, J., Abd Rahman, N. H., Mohd Dom, N. S., & Mohammad Sham, N. (2021). Mathematical epidemiologic and simulation modelling of first wave covid-19 in malaysia. *Scientific Reports*, *11*(1), 20739.

Asaithambi, A. (2021). On solving the nonlinear falkner–skan boundary-value problem: A review. *Fluids*, *6*(4), 153.

Atay, M. T., & Kilic, O. (2013). The semianalytical solutions for stiff systems of ordinary differential equations by using variational iteration method and modified variational iteration method with comparison to exact solutions. *Mathematical Problems in Engineering*, *2013*(1), 143915.

Atkinson, K. E. (2008). *An introduction to numerical analysis*. John wiley & sons.

Azevedo, J., Oliveira, S., & Rocha, A. (2020). Spectral element approximation of functional integral equations. *Electron. J. Math. Anal. Appl*, *8*(2), 172–187.

Azhari, N. A. I., Abd Rasid, N., Ijam, H. M., Arsad, R., Kamal, M. H. A., Salih, N. M., & Aksah, S. J. (2025). Designing an implicit block backward differentiation formula (bbdf) for stiff ordinary differential equations. *Data Analytics and Applied Mathematics (DAAM)*, 16–30.

Balaji, S. (2014). A new approach for solving duffing equations involving both integral and non-integral forcing terms. *Ain shams engineering journal*, *5*(3), 985–990.

Bansal, J., Kumar, A., Kumar, A., Khan, A., & Abdeljawad, T. (2025). Investigation of monkeypox disease transmission with vaccination effects using fractional order mathematical model under atangana-baleanu caputo derivative. *Modeling Earth Systems and Environment*, *11*(1), 40.

Basor, E., & Morrison, R. (2024). Analytic solutions to nonlinear odes via spectral power series. *Linear Algebra and its Applications*, *697*, 561–582.

Batiha, B., Noorani, M. M., & Hashim, I. (2007). Variational iteration method for solving multispecies lotka–volterra equations. *Computers & Mathematics with Applications*, *54*(7-8), 903–909.

Bender, C. M., & Orszag, S. A. (2013). *Advanced mathematical methods for scientists and engineers i: Asymptotic methods and perturbation theory*. Springer Science & Business Media.

Biazar, J. (2006). Solution of the epidemic model by adomian decomposition method. *Applied Mathematics and Computation*, *173*(2), 1101–1106.

Boussange, V., Becker, S., Jentzen, A., Kuckuck, B., & Pellissier, L. (2023). Deep learning approximations for non-local nonlinear pdes with neumann boundary conditions. *Partial Differential Equations and Applications*, *4*(6), 51.

Branga, A. N. (2022). Some conditions for the existence and uniqueness of monotonic and positive solutions for nonlinear systems of ordinary differential equations. *Electronic Research Archive*, *30*(6), 1999–2017.

Bülbül, B., & Sezer, M. (2013). Numerical solution of duffing equation by using an improved taylor matrix method. *Journal of Applied Mathematics*, *2013*(1), 691614.

Butcher, J. C. (2016). *Numerical methods for ordinary differential equations*. John Wiley & Sons.

Cascante-Vega, J., Torres-Florez, S., Cordovez, J., & Santos-Vega, M. (2022). How disease risk awareness modulates transmission: Coupling infectious disease models with behavioural dynamics. *Royal Society open science*, *9*(1), 210803.

Chen, J., Chen, S., Huang, J., Liu, F., Wang, S., Wang, N., Li, M., Zhang, Z., Huang, C., Du, W., et al. (2025). Mpox virus: Virology, molecular epidemiology, and global public health challenges. *Frontiers in Microbiology*, *16*, 1624110.

Chen, X., Mao, Z., & Karniadakis, G. E. (2022). Efficient and accurate numerical methods using the accelerated spectral deferred correction for solving fractional differential equations. *Numerical Mathematics: Theory, Methods and Applications*, *15*(4), 876–902.

Chowdhury, M., Hashim, I., & Hosen, M. A. (2015). Solving linear and non-linear stiffsystem of ordinary differential equations by multistage adomian decomposition method. *Proc. The Third Intl. Conf. Adv. Appl. Sci. Environ. Technol*, *125*, 79–82.

Di Giulio, D. B., & Eckburg, P. B. (2004). Human monkeypox: An emerging zoonosis. *The Lancet infectious diseases*, *4*(1), 15–25.

Dong, Z., & Xie, Y. (2025). Misuse and correction of the multistage adomian decomposition method for fractional ordinary differential equations. *Physical Review E*, *112*(6), 064215.

Duan, N. (2023). Power system simulation using multistage adomian decomposition methods. *Power System Simulation Using Semi-Analytical Methods*, 155–195.

Faires, J. D., & Burden, R. L. (2015). *Numerical methods*. Thomson/Brooks/Cole.

Feng, Z., Chen, G., & Hsu, S. (2006). A qualitative study of the damped duffing equation and applications. *DISCRETE AND CONTINUOUS DYNAMICAL SYSTEMS SERIES B*, *6*(5), 1097.

Filobello-Nino, U., Vazquez-Leal, H., Huerta-Chua, J., Callejas-Molina, R. A., Trigos, Á., & Salinas-Castro, A. (2024). A handy analytical approximate

solution for the magnetohydrodynamic flow of blood in a porous channel. *Acta universitaria*, *34*.

Forsythe, G. E., Malcolm, M. A., & Moler, C. B. (1977). *Computer methods for mathematical computations*. Prentice Hall Professional Technical Reference.

Gao, J., Zhou, C., Liang, H., Jiao, R., Wheelock, Å. M., Jiao, K., Ma, J., Zhang, C., Guo, Y., Luo, S., et al. (2023). Monkeypox outbreaks in the context of the covid-19 pandemic: Network and clustering analyses of global risks and modified seir prediction of epidemic trends. *Frontiers in public health*, *11*, 1052946.

Ghosh, S. K., & Ghosh, S. (2023). A mathematical model for covid-19 considering waning immunity, vaccination and control measures. *Scientific Reports*, *13*(1), 3610.

Gronwall, T. H. (1919). Note on the derivatives with respect to a parameter of the solutions of a system of differential equations. *Annals of Mathematics*, *20*(4), 292–296.

Gunasekar, T., Manikandan, S., Haque, S., Suba, M., & Mlaiki, N. (2025). Fractal-fractional mathematical modeling of monkeypox disease and analysis of its ulam–hyers stability. *Boundary Value Problems*, *2025*(1), 20.

Hairer, E., Wanner, G., & Nørsett, S. P. (1993). *Solving ordinary differential equations i: Nonstiff problems*. Springer.

Hamid, M., Usman, M., Haq, R. U., Tian, Z., & Wang, W. (2022). Linearized stable spectral method to analyze two-dimensional nonlinear evolutionary and reaction-diffusion models. *Numerical Methods for Partial Differential Equations*, *38*(2), 243–261.

Han, J., Hu, W., Long, J., & Zhao, Y. (2024). Deep picard iteration for high-dimensional nonlinear pdes. *arXiv preprint arXiv:2409.08526*.

Han, J., Jentzen, A., & E, W. (2018). Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences*, *115*(34), 8505–8510.

Harris, E. (2022). What to know about monkeypox. *Jama*, *327*(23), 2278–2279.

Hasegawa, C., & Duffull, S. B. (2018). Exploring inductive linearization for pharmacokinetic–pharmacodynamic systems of nonlinear ordinary

differential equations. *Journal of pharmacokinetics and pharmacodynamics*, *45*(1), 35–47.

Hermann, M., & Saravi, M. (2016). *Nonlinear ordinary differential equations*. Springer.

Higham, D. J. (2008). Modeling and simulating chemical reactions. *SIAM review*, *50*(2), 347–368.

Hoang, M. T., & Ehrhardt, M. (2025). Differential equation models for infectious diseases: Mathematical modeling, qualitative analysis, numerical methods and applications: Mt hoang, m. ehrhardt. *SeMA Journal*, 1–36.

Hofbauer, J., & Sigmund, K. (1988). Dynamical systems and the theory of evolution.

Huang, Y., & Seinfeld, J. H. (2022). A neural network-assisted euler integrator for stiff kinetics in atmospheric chemistry. *Environmental Science & Technology*, *56*(7), 4676–4685.

Hunaiber, M., et al. (2023). Application of adomian decomposition method for solving helmholtz equation. *Albaydha University Journal*, *5*(4).

Hunter, E., & Kelleher, J. D. (2022). Understanding the assumptions of an seir compartmental model using agentization and a complexity hierarchy. *Journal of Computational Mathematics and Data Science*, *4*, 100056.

Jalili, M. (2025). A hybrid double sumudu transform and optimized multistage adm framework for solving nonlinear time-fractional pdes with mixed boundary conditions. *Zagros Journal of Applied Mathematics & Data Analytics*, *1*(2), 49–70.

Ji, W., Qiu, W., Shi, Z., Pan, S., & Deng, S. (2021). Stiff-pinn: Physics-informed neural network for stiff chemical kinetics. *The Journal of Physical Chemistry A*, *125*(36), 8098–8106.

Kaur, M., Bhatti, J., Kumar, S., & Thota, S. (2024). Explicit runge-kutta method for evaluating ordinary differential equations of type vvi= f (u, v, v′). *WSEAS Transactions on Mathematics*, *23*, 167–175.

Khairuddin, A. B., Selamat, M. S., & Yacob, N. A. (2025). Solution of monkeypox transmission model using picard iterative method. *Mathematical Sciences and Informatics Journal (MIJ)*, *6*(2), 43–59.

Khan, M. A., DarAssi, M. H., Ahmad, I., Seyam, N. M., & Alzahrani, E. (2024). The transmission dynamics of an infectious disease model in fractional derivative with vaccination under real data. *Computers in Biology and Medicine*, *181*, 109069.

Khan, Y., Vázquez-Leal, H., & Faraz, N. (2012). An efficient new iterative method for oscillator differential equation. *Scientia Iranica*, *19*(6), 1473–1477.

Kingsland, S. E. (1995). *Modeling nature*. University of Chicago Press.

Krishna, S., Kurrey, C., Yadav, M., Mahilkar, S., Sonkar, S. C., Vishvakarma, N. K., Sonkar, A., Chandra, L., & Koner, B. C. (2024). Insights into the emergence and evolution of monkeypox virus: Historical perspectives, epidemiology, genetic diversity, transmission, and preventative measures. *Infectious medicine*, *3*(2), 100105.

Kumar, P., Vellappandi, M., Khan, Z. A., SM, S., Kaziboni, A., & Govindaraj, V. (2023). A case study of monkeypox disease in the united states using mathematical modeling with real data. *Mathematics and Computers in Simulation*, *213*, 444–465.

Latif, B., Selamat, M. S., Rosli, A. N., Yusoff, A. I., & Hasan, N. M. (2020). The semi analytics iterative method for solving newell-whitehead-segel equation. *Mathematics and Statistics*, *8*(2), 87–94.

Letafati, A., & Sakhavarz, T. (2023). Monkeypox virus: A review. *Microbial pathogenesis*, *176*, 106027.

Leung, A. W. (2013). *Systems of nonlinear partial differential equations: Applications to biology and engineering* (Vol. 49). Springer Science & Business Media.

Li, W., & Pang, Y. (2020). Application of adomian decomposition method to nonlinear systems. *Advances in difference Equations*, *2020*(1), 67.

Liao, L.-C., Hsu, C.-Y., Chen, H.-H., & Lai, C.-C. (2023). Estimating the global spread of epidemic human monkeypox with bayesian directed acyclic graphic model. *Vaccines*, *11*(2), 468.

Lin, M., Xin, Y., Wang, J., Nie, P., Yan, Q., Wang, L., & Wang, L. (2024). Analysing monkeypox epidemic drivers: Policy simulation and multi-index modelling across 39 nations. *Journal of Global Health*, *14*, 04037.

Lindelöf, E. (1894). On the application of the method of successive approximations to first-order ordinary differential equations. *Weekly reports of the sessions of the Academy of Sciences*, *116*(3), 454–457.

Liu, B., Farid, S., Ullah, S., Altanji, M., Nawaz, R., & Wondimagegnhu Teklu, S. (2023a). Mathematical assessment of monkeypox disease with the impact of vaccination using a fractional epidemiological modeling approach. *Scientific Reports*, *13*(1), 13550.

Liu, C.-S., Li, B., & Kuo, C.-L. (2025). Variational iteration and linearized liapunov methods for seeking the analytic solutions of nonlinear boundary value problems. *Mathematics*, *13*(3), 354.

Liu, T., Yin, X., Chen, Y., & Hou, M. (2023b). A second-order accurate numerical approximation for a two-sided space-fractional diffusion equation. *Mathematics*, *11*(8), 1838.

Lyons, R., Vatsala, A. S., & Chiquet, R. A. (2017). Picard's iterative method for caputo fractional differential equations with numerical results. *Mathematics*, *5*(4), 65.

Ma, L., & Liu, Y. (2024). Stochastic chebyshev-picard iteration method for nonlinear differential equations with random inputs. *Communications in Mathematical Research*, *40*(3), 275–312.

Madubueze, C. E., Onwubuya, I. O., Nkem, G. N., & Chazuka, Z. (2022). The transmission dynamics of the monkeypox virus in the presence of environmental transmission. *Frontiers in Applied Mathematics and Statistics*, *8*, 1061546.

Majee, S., Jana, S., Barman, S., & Kar, T. (2023). Transmission dynamics of monkeypox virus with treatment and vaccination controls: A fractional order mathematical approach. *Physica Scripta*, *98*(2), 024002.

Manivel, M., Venkatesh, A., & Kumawat, S. (2025). Numerical simulation for the co-infection of monkeypox and hiv model using fractal-fractional operator. *Modeling Earth Systems and Environment*, *11*(3), 157.

Movaheedi, Z., & Rahmani, A. R. (2026). Comparative analysis of runge–kutta and backward euler methods for modeling monkeypox transmission dynamics. *Afghanistan Journal of Infectious Diseases*, *4*(1), 94–113.

Mungkasi, S. (2020). Successive approximation, variational iteration, and multistage-analytical methods for a seir model of infectious disease involving vaccination strategy. *Communication in Biomathematical Sciences*, *3*(2), 114–126.

Murugesh, V., Priyadharshini, M., Sharma, Y. K., Lilhore, U. K., Alroobaea, R., Alsufyani, H., Baqasah, A. M., & Simaiya, S. (2025). A novel hybrid framework for efficient higher order ode solvers using neural networks and block methods. *Scientific Reports*, *15*(1), 8456.

Musafir, R. R., Suryanto, A., Darti, I., et al. (2024). Stability analysis of a fractional-order monkeypox epidemic model with quarantine and hospitalization. *Journal of Biosafety and Biosecurity*, *6*(1), 34–50.

Newell, A. C., & Whitehead, J. A. (1969). Finite bandwidth, finite amplitude convection. *Journal of Fluid Mechanics*, *38*(2), 279–303.

Ngungu, M., Addai, E., Adeniji, A., Adam, U. M., & Oshinubi, K. (2023). Mathematical epidemiological modeling and analysis of monkeypox dynamism with non-pharmaceutical intervention using real data from united kingdom. *Frontiers in Public Health*, *11*, 1101436.

Nguyen, N. V., Nguyen, H. X., Lee, S., & Nguyen-Xuan, H. (2018). Geometrically nonlinear polygonal finite element analysis of functionally graded porous plates. *Advances in Engineering software*, *126*, 110–126.

Nourazar, S., & Mirzabeigy, A. (2013). Approximate solution for nonlinear duffing oscillator with damping effect using the modified differential transform method. *Scientia Iranica*, *20*(2), 364–368.

Nwaigwe, C., & Mishra, V. N. (2024). Accelerating fixed point algorithms for nonlinear fredholm integral equations.

Odibat, Z. (2020). An improved optimal homotopy analysis algorithm for nonlinear differential equations. *Journal of Mathematical Analysis and Applications*, *488*(2), 124089.

Okyere, S., & Ackora-Prah, J. (2023). Modeling and analysis of monkeypox disease using fractional derivatives. *Results in engineering*, *17*, 100786.

Olek, S. (1994). An accurate solution to the multispecies lotka–volterra equations. *Siam Review*, *36*(3), 480–488.

Onifade, A. A., Akindele, O. A., Ahmad, I., Khan, M. A., Mat Isa, N., & Alzahrani, E. (2025). Modeling monkeypox transmission with a compartmental framework to evaluate testing, isolation and public awareness strategies. *Scientific Reports*, *15*(1), 27236.

Peter, O. J., Kumar, S., Kumari, N., Oguntolu, F. A., Oshinubi, K., & Musa, R. (2022a). Transmission dynamics of monkeypox virus: A mathematical modelling approach. *Modeling Earth Systems and Environment*, *8*(3), 3423–3434.

Peter, O. J., Oguntolu, F. A., Ojo, M. M., Olayinka Oyeniyi, A., Jan, R., & Khan, I. (2022b). Fractional order mathematical model of monkeypox transmission dynamics. *Physica Scripta*, *97*(8), 084005.

Picard, E. (1893). *Traité d'analyse: Fonctions harmoniques et fonctions analytiques. introduction à la théorie des équations différentielles, intégrales abéliennes et surfaces de riemann* (Vol. 2). Gauthier-Villars.

Piovella, N. (2020). Analytical solution of seir model describing the free spread of the covid-19 pandemic. *Chaos, Solitons & Fractals*, *140*, 110243.

Pohlhausen, K. (1921). On the approximate integration of the differential equation of the internal boundary layer. *ZAMM-Journal of Applied Mathematics and Mechanics/Journal of Applied Mathematics and Mechanics*, *1*(4), 252–290.

Pollock, S., Rebholz, L. G., & Xiao, M. (2019). Anderson-accelerated convergence of picard iterations for incompressible navier–stokes equations. *SIAM Journal on Numerical Analysis*, *57*(2), 615–637.

Prabowo, P. S., & Mungkasi, S. (2021). A multistage successive approximation method for riccati differential equations. *Bulletin of Electrical Engineering and Informatics*, *10*(3), 1589–1597.

Priyadarsini, D., Sahu, P., & Routaray, M. (2025). Novel combined technique for solving chen–lee–liu equation modeling quantum particle dynamics. *International Journal of Computational Methods*, 2550048.

Qian, M., Li, D., Hao, Z., Hu, S., & Li, W. (2025). An epidemiological model of monkeypox: Model prediction and control application. *BMC Infectious Diseases*, *25*(1), 485.

Rafei, M., Daniali, H., & Ganji, D. (2007). Variational iteration method for solving the epidemic model and the prey and predator problem. *Applied Mathematics and Computation*, *186*(2), 1701–1709.

Razali, N. I., Chowdhury, M., & Asrar, W. (2013). The multistage adomian decomposition method for solving chaotic lü system. *Middle-east J. Scientific Research*, *13*, 43–49.

Reynolds, M. G., Doty, J. B., McCollum, A. M., Olson, V. A., & Nakazawa, Y. (2019). Monkeypox re-emergence in africa: A call to expand the concept and practice of one health. *Expert review of anti-infective therapy*, *17*(2), 129–139.

Sahani, S. K., & Sah, B. K. (2024). An in-depth stability and convergence analysis of the runge-kutta 4th order method for nonlinear ordinary differential equations. *Panamerican Mathematical Journal ISSN*, *34*(2), 300–312.

Schnell, S., & Mendoza, C. (1997). Closed form solution for time-dependent enzyme kinetics. *Journal of theoretical Biology*, *187*(2), 207–212.

Segel, L. A. (1969). Distant side-walls cause slow amplitude modulation of cellular convection. *Journal of Fluid Mechanics*, *38*(1), 203–224.

Sen, A. K. (1988). An application of the adomian decomposition method to the transient behavior of a model biochemical reaction. *Journal of mathematical analysis and applications*, *131*(1), 232–245.

Sheu, L.-J., Chen, H.-K., Chen, J.-H., & Tam, L.-M. (2007). Chaotic dynamics of the fractionally damped duffing equation. *Chaos, Solitons & Fractals*, *32*(4), 1459–1468.

Suantai, S., Sabir, Z., Umar, M., & Cholamjiak, W. (2023). Scaled conjugate gradient for the numerical simulations of the mathematical model-based monkeypox transmission. *Fractal and Fractional*, *7*(1), 63.

Sudsutad, W., Thaiprayoon, C., Aphithana, A., Kongson, J., & Sae-dan, W. (2024). Qualitative results and numerical approximations of the (k, ψ)-caputo proportional fractional differential equations and applications to blood alcohol levels model. *AIMS Math*, *9*(12), 34013–34041.

Sweilam, N. H., Zaghrout, A. S., & Ali, N. Y. (2024). Variable-order hybrid covid-19 mathematical model with time delay; numerical treatments.

Sweilam, N., Al-Mekhlafi, S., Kareem, W. A., & Alqurishi, G. (2025). A new crossover dynamics mathematical model of monkeypox disease based on fractional differential equations and the ψ-caputo derivative: Numerical treatments. *Alexandria Engineering Journal*, *111*, 181–193.

Tafakkori-Bafghi, M., Barid Loghmani, G., Heydari, M., & Bai, X. (2020). Jacobi-picard iteration method for the numerical solution of nonlinear initial value problems. *Mathematical methods in the applied sciences*, *43*(3), 1084–1111.

Tajudeen, Y. A., Oladipo, H. J., Muili, A. O., & Ikebuaso, J. G. (2023). Monkeypox: A review of a zoonotic disease of global public health concern. *Health Promotion Perspectives*, *13*(1), 1.

Tamakloe, M. D., Kuuwill, A., Osumanu, I., & Siripi, H. (2025). Mathematical modelling and time series clustering of mpox outbreak: A comparative study of the top 10 affected countries and implications for future outbreak management. *Global Epidemiology*, 100214.

Teschl, G. (2012). *Ordinary differential equations and dynamical systems* (Vol. 140). American Mathematical Soc.

Tolles, J., & Luong, T. (2020). Modeling epidemics with compartmental models. *Jama*, *323*(24), 2515–2516.

Turkyilmazoglu, M. (2012). An effective approach for approximate analytical solutions of the damped duffing equation. *Physica Scripta*, *86*(1), 015301.

Wanner, G., & Hairer, E. (1996). *Solving ordinary differential equations ii* (Vol. 375). Springer Berlin Heidelberg New York.

Wong, S. W., Yang, S., & Kou, S. (2023). Estimating and assessing differential equation models with time-course data. *The Journal of Physical Chemistry B*, *127*(11), 2362–2374.

Wu, S., Deng, J., Du, M., Liu, M., & Liu, J. (2025). Global transmission characteristics of mpox outbreaks: A systematic review and meta-analysis. *Eclinicalmedicine*, *89*.

Wu, Z., Wang, H., He, C., Zhang, B., Xu, T., & Chen, Q. (2023). The application of physics-informed machine learning in multiphysics modeling in chemical

engineering. *Industrial & Engineering Chemistry Research*, *62*(44), 18178–18204.

Yadav, R., Chaudhary, A. A., Srivastava, U., Gupta, S., Rustagi, S., Rudayni, H. A., Kashyap, V. K., & Kumar, S. (2025). Mpox 2022 to 2025 update: A comprehensive review on its complications, transmission, diagnosis, and treatment. *Viruses*, *17*(6), 753.

Yahaya, F., Hashim, I., Ismail, E., & Zulkifle, A. (2007). Direct solutions of n-th order initial value problems in decomposition series. *International Journal of Nonlinear Sciences and Numerical Simulation*, *8*(3), 385–392.

Yao, Y., Miao, S., & Lv, G. (2021). An efficient iterative method for radiation heat conduction problems. *International Journal for Numerical Methods in Fluids*, *93*(7), 2362–2379.

Yapışkan, D., Yurtoğlu, M., Avcı, D., İskender Eroğlu, B. B., & Bonyah, E. (2023). A novel model for monkeypox disease: System analysis and optimal preventive strategies. *Iranian Journal of Science*, *47*(5), 1665–1677.

Yasui, K. (2022). Merits and demerits of ode modeling of physicochemical systems for numerical simulations. *Molecules*, *27*(18), 5860.

Yeshwanth, R., & Kumbinarasaiah, S. (2025). Modified hermite wavelet method for the transmission dynamics of monkeypox virus model. *New Mathematics and Natural Computation*, 1–28.

Youssef, I., Saad, M. K., Dumlu, A., & Asklany, S. A. (2024). Practical aspects for applying picard iterations to the sir model using actual data. *International Journal of Analysis and Applications*, *22*, 32–32.

Yusufoğlu, E. (2006). Numerical solution of duffing equation by the laplace decomposition algorithm. *Applied Mathematics and Computation*, *177*(2), 572–580.

Zeng, M.-L., & Zhang, G.-F. (2020). On c-to-r-based iteration methods for a class of complex symmetric weakly nonlinear equations. *Mathematics*, *8*(2), 208.

Zhao, F., Sheng, Z., & Yuan, G. (2025). A finite volume scheme preserving strong extremum principle for three-dimensional diffusion equations and its anderson acceleration. *Journal of Computational Mathematics*.

Zhou, Q., Wang, Y., & Liu, Y. (2024). Chebyshev–picard iteration methods for solving delay differential equations. *Mathematics and Computers in Simulation*, *217*, 1–20.

Ziada, E. (2022). Analytical solution of a system of ordinary differential equations. *Nile Journal of Basic Science*, *2*(1), 11–20.

APPENDICES

APPENDIX A

Maple code for Initial Value Problems (IVPS)

Picard Iterative Method for *n*th Order (PIM*n*) for Example 1

This appendix presents the Maple code for applying the PIM*n* to solve the given initial value problem (IVP) in Example 1. The implementation includes the initial condition and the iterative procedure used to obtain the approximate solution directly in its higher-order form.

```
> ###nth-order Initial Value Problems
> restart;
> Digits:=16:
> ##Initial Condition
> y[0](t):=1+2*t:
> ##Apply Picard nth-order to IVPs
> for n from 0 to 10 do
> y[n+1](t):= y[0](t) + int((t-s)*(subs(t=s, -diff(y[n](t),
t$1)^2)), s=0..t):
> epsilon:=evalf(subs(t=0.1,abs(y[n+1](t)-y[n](t))));
> if epsilon <=10^(-5) then break else print(epsilon) end if
> end;
```

Picard Iterative Method for *n*th Order (PIM*n*) for Example 2

This appendix presents the Maple code for applying the PIM*n* to solve the given initial value problem (IVP) in Example 2. The implementation includes the initial condition and the iterative procedure used to obtain the approximate solution directly in its higher-order form.

```
> ###nth-order Initial Value Problems
> restart;
> Digits:=16:
> ##Initial Condition
> y[0](t):=1+2*t:
```

```
> ##Apply Picard nth-order to IVPs
> for n from 0 to 10 do
> y[n+1](t):= y[0](t) + int((t-s)*(subs(t=s, -diff(y[n](t),
t$1)^2)), s=0..t):
> epsilon:=evalf(subs(t=0.1,abs(y[n+1](t)-y[n](t))));
> if epsilon <=10^(-5) then break else print(epsilon) end if
> end;
```

APPENDIX B

Maple code for Riccati Equation

Picard Iterative Method

This appendix presents the Maple code for applying the Picard Iterative Method (PIM) to solve the Riccati equation. The implementation includes the initial condition and the iterative procedure used to obtain the approximate solution.

```
> restart:
> with(plots):
>######Initial Value
> u[0](t) := -0.5:
> ######Apply PIM
> for n from 0 to 10 do;
> u[n+1](t) := u[0](t) + int(1 - u[n](t)^2, t);
> epsilon:=evalf(subs(t=0.1,abs(u[n+1](t)-u[n](t))));
> if epsilon <=10^(-5) then break else print(epsilon) end if
> end;
```

Multistage Picard Iterative Method

This appendix presents the Maple code for implementing the Multistage Picard Iterative Method (MPIM) to solve the Riccati equation. The implementation includes the initial value and the multistage iterative procedure applied over partitioned time intervals to obtain improved approximate solutions.

```
> restart; Digits := 10:
> ######Initial Value
> u[0](t) := -0.5:
> ######Apply MPIM
> for m from 0 to 6 by 0.01 do
> for n from 0 to 3 do
> u[n+1](t) := u[0](t) + int(1 - u[n](t)^2, s = m .. t);
> end do:
```

```
> a := simplify(eval(u[k](t), k=4)):
> masa:=m+0.01:
> uu := evalf(subs(t = masa, u[n](t))):
> u[0](t) := uu:
> print(u[0](t));
> end do;
```

APPENDIX C

Maple code for Duffing Equation

Picard Iterative Method for *n*th Order (PIM*n*) for Damping Duffing Equation

This appendix presents the Maple code for applying the PIM*n* to solve the damping Duffing equation. The implementation includes the initial condition and the iterative procedure used to obtain the approximate solution directly in its higher-order form.

```
> ##Solution of Damping Duffing Equation using Picard nth-order
> restart;
> Digits:=16:
> ##Initial Condition
> y[0](t):=1+ t:
> ##Apply Picard nth-order
> for n from 0 to 10 do
> y[n+1](t):= y[0](t) + int((t-s)*(-subs(t=s, diff(y[n](t), t))
> epsilon:=evalf(subs(t=0.1, abs(y[n+1](t)-y[n](t))));
> if epsilon <= 10^(-5) then break else print(epsilon) end if
> end;
```

Multistage Picard Iterative Method for *n*th Order (MPIM*n*) for Damping Duffing Equation

This appendix presents the Maple code for implementing the MPIM*n* to solve the damping Duffing equation. The implementation includes the parameter settings, initial conditions, and the multistage iterative procedure applied directly to the higher-order equation to obtain improved approximate solutions.

```
> #### Solution of Damping Duffing Equation using MPIMn
> restart:
> Digits := 16:
> h := 0.01:
> # Damped Duffing: u'' + u' + u^3 = 0
```

```
> # let y=u, v=u' -> v' = -v - y^3
> F := (s, y, v) -> -v - y^3:
> # Initial conditions: u(0)=1, u'(0)=1
> y0 := 1.0:
> v0 := 1.0:
> for k from 0 to 600 do
> m := k*h:
> masa := m + h:
> # initial guess (expressions in t)
> yphi0 := y0 + v0*(t - m):
> vphi0 := v0:
> # 1st Picard correction
> vphi1 := v0 + int(F(s, subs(t=s, yphi0), subs(t=s, vphi0)),
           s = m .. t):
> yphi1 := y0 + int(subs(t=s, vphi1), s = m .. t):
> # 2nd Picard correction
> vphi2 := v0 + int(F(s, subs(t=s, yphi1), subs(t=s, vphi1)),
           s = m .. t):
> yphi2 := y0 + int(subs(t=s, vphi2), s = m .. t):
> # ===== PRINT EQUATIONS EACH ITERATION =====
    printf("\n================ Stage k=%d: [%g, %g]
    =================\n", k, m, masa):
> print("phi0:\n"):
> print(vphi0):
> print(yphi0):

> printf("phi1:\n"):
> print(simplify(vphi1)):
> print(simplify(yphi1)):

> printf("phi2:\n"):
> print(simplify(vphi2)):
```

```
> print(simplify(yphi2)):

> # update at end of stage (numeric)
> y0 := evalf(subs(t=masa, yphi2)):
> v0 := evalf(subs(t=masa, vphi2)):

> printf("UPDATE at t=%.2f: u ~ %.16f, u' ~ %.12f\n", masa, y0,
          v0):
> end do:
```

Picard Iterative Method for *n*th Order (PIM*n*) for Undamping Duffing Equation

This appendix presents the Maple code for applying the PIM*n* to solve the undamped Duffing equation. The implementation includes the initial condition and the iterative procedure used to obtain the approximate solution directly in its higher-order form.

```
> ###Solution of Undamping Duffing Equation using Picard nth
     order
> restart;
> Digits:=16:
> ##Initial Condition
> y[0](t):=t:
> ##Apply Picard nth-order
> for n from 0 to 10 do
> y[n+1](t):= y[0](t) + int((t-s)*(cos(s)*sin(2*s)- 3*subs(t=s,
              y[n](t))+ 2*subs(t=s, (y[n](t))^3)), s=0..t):
> epsilon:=evalf(subs(t=0.1, abs(y[n+1](t)-y[n](t))));
> if epsilon <=10^(-5) then break else print(epsilon) end if
> end;
```

Multistage Picard Iterative Method for *n*th Order (MPIM*n*) for Undamping Duffing Equation

This appendix provides the Maple implementation of the MPIM*n* for the undamped Duffing equation, including the initial value and the multistage updating process without transforming the equation into a first-order system.

```
> restart:
> Digits := 16:
> h := 0.01:
> f := (s, y) -> cos(s)*sin(2*s) - 3*y + 2*y^3:
> y0 := 0.0:
> v0 := 1.0:
> for k from 0 to 600 do
> m := k*h:
> masa := m + h:
> phi0 := y0 + v0*(t - m):
> phi1 := y0 + v0*(t - m) + int((t-s)*f(s, subs(t=s, phi0)),
          s = m .. t):
> phi2 := y0 + v0*(t - m) + int((t-s)*f(s, subs(t=s, phi1)),
          s = m .. t):
> printf("\n========== Stage k=%d: [%g, %g] ==========\n",
          k, m, masa):
> printf("phi0(t) =\n"); print(simplify(phi0));
> printf("phi1(t) =\n"); print(simplify(phi1));
> printf("phi2(t) =\n"); print(simplify(phi2));
> # update y
> y0 := evalf(subs(t=masa, phi2)):
> # update v
> v0 := evalf(v0 + int(f(s, subs(t=s, phi2)), s = m .. masa)):
> printf("UPDATE at t=%.2f: y = %.10f, v = %.10f\n", masa, y0,
         v0):
end do:
```

APPENDIX D

Maple code for Pohlhausent Flow Equation

Picard Iterative Method for *n*th Order (PIM*n*)

This appendix presents the Maple code for applying the PIM*n* to solve the Pohlhausen flow equation. The implementation includes the initial conditions and the iterative procedure used to obtain the approximate solution directly in its higher-order form.

```
> restart:
> Digits := 18:
> # Initial conditions
> alpha := 1.154701:
> y[0](t) := 0:          # y(0)
> y1[0](t) := 0:         # y'(0)
> y2[0](t) := alpha:     # y''(0)
> # Picard iterations for y''' = (y')^2 - 1
> for n from 0 to 10 do
> y[n+1](t) := y[0](t) + y1[0](t)*t + (y2[0](t)/2)*t^2
               + int(((t-s)^2)/2 * ((subs(t=s, diff(y[n](t),
               t))^2 - 1)), s=0..t):
> epsilon := evalf(subs(t=0.1, abs(y[n+1](t) - y[n](t))));
> if epsilon <= 10^(-10) then print("Converged at n=", n+1,
  "epsilon =", epsilon); break; else print("n =", n+1,
  "epsilon =", epsilon); end if;
> end do:
```

Multistage Picard Iterative Method for *n*th Order (MPIM*n*)

This appendix presents the Maple code for implementing the MPIM*n* to solve the Pohlhausen flow equation. The implementation includes the parameter settings, initial conditions, and the multistage iterative procedure applied directly to the higher-order equation to obtain improved approximate solutions.

```
> restart:
> Digits := 16:
> h := 0.01:
> T := 6:
> kmax := trunc(T/h):
> # ODE: f''' = (f')^2 - 1
> g := (s, fp) -> fp^2 - 1:
> # Initial conditions
> F0 := 0.0:
> Fp0 := 0.0:
> Fpp0 := -1.154701:
> printf("Initial: f=%.6f, f'=%.6f, f''=%.6f\n", F0, Fp0, Fpp0);
> for k from 0 to kmax-1 do
> m := k*h:
> tn := m + h:
> printf("\n========== Stage k=%d: [%g, %g] ==========\n", k+1,
  m, tn);
> # Initial guess (3rd order Taylor)
> phi0 := F0 + Fp0*(t-m) + (Fpp0/2)*(t-m)^2:
> printf("phi0(t) =\n");
> print(simplify(phi0));
> # 1st correction
> phi1 := F0 + Fp0*(t-m) + (Fpp0/2)*(t-m)^2 + int(((t-s)^2)/2 *
          g(s, subs(t=s, diff(phi0,t))), s = m .. t):
> printf("phi1(t) =\n");
> print(simplify(phi1));
```

```
> # 2nd correction
> phi2 := F0 + Fp0*(t-m) + (Fpp0/2)*(t-m)^2 + int(((t-s)^2)/2 *
          g(s, subs(t=s, diff(phi1,t))),s = m .. t):
> printf("phi2(t) =\n");
> print(simplify(phi2));
> # Update at tn
> F0   := evalf(subs(t=tn, phi2)):
> Fp0  := evalf(subs(t=tn, diff(phi2,t))):
> Fpp0 := evalf(subs(t=tn, diff(phi2,t$2))):
> printf("UPDATE at t=%.2f:\n", tn);
> printf("f  = %.16f\n", F0);
> printf("f' = %.10f\n", Fp0);
> printf("f'' = %.10f\n", Fpp0);
> end do:
```

APPENDIX E

Maple code for Newel Whitehead Segel Equation

Picard Iterative Method (PIM)

This appendix presents the Maple code for applying the PIM to solve the Newell Whitehead Segel equation. The implementation includes the initial condition and the iterative procedure used to obtain the approximate solution of the nonlinear partial differential equation.

```
> restart:
> with(plots):
> u[0](x,t) := exp(2*x):
> ######Apply Picard Method
> for n from 0 to 6 do
> u[n+1](x,t) := u[0](x,t) + int(subs(t=s, diff(u[n](x,t), x$2)
                - 3*u[n](x,t)), s = 0 .. t):
> epsilon := evalf(subs(t=0.1, x=0, abs(u[n+1](x,t) - u[n]
            (x,t)))):
> if epsilon <= 10^(-5) then print("Converged at n=", n,
"epsilon=", epsilon); break; else print(n, epsilon); end if;
> end do:
```

Multistage Picard Iterative Method (MPIM)

This appendix presents the Maple code for implementing the MPIM to solve the Newell Whitehead Segel equation. The implementation includes the initial condition and the multistage iterative procedure applied over partitioned time intervals to obtain improved approximate solutions of the nonlinear partial differential equation.

```
> restart; Digits := 16:
> u[0](x,t) := exp(2*x);
> for m from 0 to 6 by 0.01 do
> for n from 0 to 3 do
```

```
> u[n+1](x,t) := u[0](x,t) + int(subs(t=s, diff(u[n](x,t), x$2)
                - 3*u[n](x,t)), s = m .. t):
> end do:
> a := simplify(eval(u[k](x,t), k=4)):
> masa := m + h:
> uu := simplify(subs(t = masa, u[4](x,t))):
> u[0](x,t) := uu:
> print(masa, evalf(subs(x=0, uu)));
> end do:
```

APPENDIX F

Maple code for Monkeypox Transmission Model

Picard Iterative Method (PIM)

This appendix presents the Maple code used for the implementation of the PIM in solving the Monkeypox transmission model. The code includes the parameter setup, initial conditions, and the iterative procedure applied to obtain the approximate numerical solutions of the model.

```
> restart;with(Plot);
```

[*AnalyzeData*, *AnimatePoints*, *ColorRange*, *ColorScheme*, *ContourPlot*, *CoordinateSystems*, *DataPlot*, *DualAxisPlot*, *Export*, *Extrude*, *FormatText*, *GetAdaptiveOptions*, *GetLocalOptions*, *Import*, *Inequality*, *IsCoordinateSystem*, *IsOption*, *LogPlot2D*, *MakeColorBox*, *MergeAXIS*, *ModuleApply*, *Plot2D*, *Plot3D*, *PlotArray*, *PointPlot*, *PolarPlot*, *PolygonPlot*, *Preprocess*, *ProcessLocalOptions*, *ProcessUnits*, *Redraw*, *ResetView*, *SetColors*, *SetColours*, *ShadeBetween*, *SpaceCurve*, *StackedPlot*, *Structure*, *TernaryPlot*, *TextPlot*, *TranslateOptions*, *Utilities*, *VerifyAdaptiveOptions*] (1)

```
> ###############Parameter Setup
> Digits:=10;
```

$$Digits := 10 \quad (2)$$

```
> alpha1:= 0.423890; #transmission rate of humans from exposed to
  infectious class
```

$$\alpha 1 := 0.423890 \quad (3)$$

```
> alpha2:= 1.797575;  #rate of identification of suspected case
```

$$\alpha 2 := 1.797575 \quad (4)$$

```
> alpha3:= 0.025289;  #transmission rate of rodents from exposed to
  infectious class
```

$$\alpha 3 := 0.025289 \quad (5)$$

```
> beta1:= 0.011503;  #contact rate between rodents and humans
```

$$\beta 1 := 0.011503 \quad (6)$$

```
> beta2:= 0.747322;  #contact rate between humans
```

$$\beta 2 := 0.747322 \quad (7)$$

```
> beta3:= 0.321102;  #contact rate between rodents
```

$$\beta 3 := 0.321102 \quad (8)$$

```
> Lambdah:=0.34857;  #recruitment rate of humans
```

$$Lambdah := 0.34857 \quad (9)$$

```
> Lambdar:= 0.60822;  #recruitment rate of rodents
```

$$Lambdar := 0.60822 \quad (10)$$

```
> phi := 1.575454;  #proportion of humans not detected after
  diagnosis
```

$$\phi := 1.575454 \quad (11)$$

```
> rho := 0.067689; #recovery rate
```

$$\rho := 0.067689 \tag{12}$$

```
> muh:= 1/(79*365); #natural death rates of humans
```

$$muh := \frac{1}{28835} \tag{13}$$

```
> mur:=1/(5*365);  #natural death rates of rodents
```

$$mur := \frac{1}{1825} \tag{14}$$

```
> deltah := 0.015016;  #disease induced death rates of humans
```

$$deltah := 0.015016 \tag{15}$$

```
> deltar := 0.000025;  #disease induced death rates of rodents
```

$$deltar := 0.000025 \tag{16}$$

```
> tau := 0.999763;  #rate of transmission from isolation to
  recovered class
```

$$\tau := 0.999763 \tag{17}$$

```
> ###################Initial Condition
> NH := 10000+50+0.57+0.43+0
```

$$NH := 10051.00 \tag{18}$$

```
> SH[0] (t):= 10000;  #susceptible class human
```

$$SH_0(t) := 10000 \tag{19}$$

```
> EH[0] (t):= 50;  #exposed class human
```

$$EH_0(t) := 50 \tag{20}$$

```
> UH[0] (t) := 0.57;  #infected class human
```

$$UH_0(t) := 0.57 \tag{21}$$

```
> QH[0] (t):= 0.43;  #isolation class
```

$$QH_0(t) := 0.43 \tag{22}$$

```
> RH[0] (t):= 0;  #recovery class
```

$$RH_0(t) := 0 \tag{23}$$

```
> NR := 1000+100+10
```

$$NR := 1110 \tag{24}$$

```
> SR[0] (t):= 1000;  #susceptible class rodent
```

$$SR_0(t) := 1000 \tag{25}$$

```
> ER[0] (t):= 100;  #exposed class rodent
```

$$ER_0(t) := 100 \tag{26}$$

```
> VR[0](t):= 10;  #infected class rodent
```

$$VR_0(t) := 10 \tag{27}$$

```
> ########################Apply Picard Method
> for n from 0 to 15 do;
> SH[n+1](t) := SH[0](t) + int(Lambdah - (beta1*VR[n](t) + beta2*UH
  [n](t))*SH[n](t)/NH - muh*SH[n](t) + phi*QH[n](t), t);
> EH[n+1](t) := EH[0](t) + int((beta1*VR[n](t) + beta2*UH[n](t))*SH
  [n](t)/NH - (alpha1 + alpha2 + muh)*EH[n](t), t);
> UH[n+1](t) := UH[0](t) + int(alpha1*EH[n](t) - (muh + deltah +
  rho)*UH[n](t), t);
> QH[n+1](t) := QH[0](t) + int(alpha2*EH[n](t) - (phi + tau + muh +
  deltah)*QH[n](t), t);
> RH[n+1](t) := RH[0](t) + int(rho*UH[n](t) + tau*QH[n](t) - muh*RH
  [n](t), t);
> SR[n+1](t) := SR[0](t) + int(Lambdar - beta3*SR[n](t)*VR[n](t)/NR
  - mur*SR[n](t), t);
> ER[n+1](t) := ER[0](t) + int(beta3*SR[n](t)*VR[n](t)/NR - (mur +
  alpha3)*ER[n](t), t);
> VR[n+1](t) := VR[0](t) + int(alpha3*ER[n](t) - (mur + deltar)*VR
  [n](t), t);
> epsilon:=evalf(subs(t=0.1,abs(SH[n+1](t)-SH[n](t))));
> if epsilon <=10^(-5) then break else print (epsilon) end if
> end;
```

$$SH_1(t) := 10000 + 0.1409560350\,t$$

$$EH_1(t) := 50 - 110.5367256\,t$$

$$UH_1(t) := 0.57 + 21.14733838\,t$$

$$QH_1(t) := 0.43 + 88.76493490\,t$$

$$RH_1(t) := 0.4684808200\,t$$

$$SR_1(t) := 1000 - 2.832536016\,t$$

$$ER_1(t) := 100 + 0.3091162900\,t$$

$$VR_1(t) := 10 + 2.523170548\,t$$

$$\epsilon := 0.01409560350$$

$$0.01409560350$$

$$SH_2(t) := 10000 + 0.1409560350\,t + 62.04625101\,t^2 - 0.00007401393346\,t^3$$

$$EH_2(t) := 50 - 110.5367256\,t + 130.6549326\,t^2 + 0.00007401393346\,t^3$$

$$UH_2(t) := 0.57 + 21.14733838\,t - 24.30256831\,t^2$$

$$QH_2(t) := 0.43 + 88.76493490\,t - 214.3114982\,t^2$$

$$RH_2(t) := 0.4684808200\,t + 45.08766178\,t^2$$

$$SR_2(t) := 1000 - 2.832536016\,t - 0.3600797193\,t^2 + 0.0006891612094\,t^3$$

$$ER_2(t) := 100 + 0.3091162900\,t + 0.3568624462\,t^2 - 0.0006891612094\,t^3$$

$$VR_2(t) := 10 + 2.523170548\,t + 0.003185801695\,t^2$$

$$\epsilon := 0.6204624361$$

$$0.6204624361$$

$$SH_3(t) := 10000 + 0.1409560350\,t + 62.04625101\,t^2 - 106.5246562\,t^3 - 0.02437100065\,t^4 + 0.02242310668\,t^5 - 2.229010044\,10^{-8}\,t^6$$

$$EH_3(t) := 50 - 110.5367256\,t + 130.6549326\,t^2 - 102.7719938\,t^3 + 0.02432989581\,t^4 - 0.02242310668\,t^5 + 2.229010044\,10^{-8}\,t^6$$

$$UH_3(t) := 0.57 + 21.14733838\,t - 24.30256831\,t^2 + 19.13136870\,t^3 + 0.000007843441562\,t^4$$

$$QH_3(t) := 0.43 + 88.76493490\,t - 214.3114982\,t^2 + 263.3287292\,t^3 + 0.00003326139910\,t^4$$

$$RH_3(t) := 0.4684808200\,t + 45.08766178\,t^2 - 71.96909553\,t^3$$

$$SR_3(t) := 1000 - 2.832536016\,t - 0.3600797193\,t^2 + 0.0007949459768\,t^3 + 0.00006576584409\,t^4 - 3.423518282\,10^{-8}\,t^5 - 1.058542611\,10^{-10}\,t^6$$

$$ER_3(t) := 100 + 0.3091162900\,t + 0.3568624462\,t^2 - 0.003802589815\,t^3 - 0.00006140879464\,t^4 + 3.423518282\,10^{-8}\,t^5 + 1.058542611\,10^{-10}\,t^6$$

$$VR_3(t) := 10 + 2.523170548\,t + 0.003185801695\,t^2 + 0.003007623037\,t^3 - 0.000004357049455\,t^4$$

$$\epsilon := 0.1065267951$$

$$0.1065267951$$

$$SH_4(t) := 10000 + 0.1409560350\,t + 1.874220825\,10^{-12}\,t^{10} + 1.171642503\,10^{-18}\,t^{11} + 0.000009405826937\,t^8 - 0.000003544044915\,t^9 + 0.02163565633\,t^7 + 0.05595345215\,t^5 - 0.04678489574\,t^6 + 100.1373657\,t^4 - 106.5246562\,t^3 + 62.04625101\,t^2$$

$$EH_4(t) := 50 - 1.874220825\,10^{-12}\,t^{10} - 1.171642503\,10^{-18}\,t^{11} - 0.000009405826937\,t^8 + 0.000003544044915\,t^9 - 0.02163566341\,t^7 - 0.06675257390\,t^5 + 0.05508692019\,t^6 + 60.65612062\,t^4 - 110.5367256\,t - 102.7719938\,t^3 + 130.6549326\,t^2$$

$$UH_4(t) := 0.57 + 21.14733838\,t - 24.30256831\,t^2 + 19.13136870\,t^3 - 11.28673594\,t^4$$

$$+ 0.002062510114\, t^5 - 0.001584155115\, t^6 + 1.349792954\, 10^{-9}\, t^7$$

$$QH_4(t) := 0.43 + 88.76493490\, t - 214.3114982\, t^2 + 263.3287292\, t^3 - 216.7080658\, t^4$$
$$+ 0.008729731306\, t^5 - 0.006717869332\, t^6 + 5.724018186\, 10^{-9}\, t^7$$

$$RH_4(t) := 0.4684808200\, t + 45.08766178\, t^2 - 71.96909553\, t^3 + 66.14094985\, t^4$$
$$+ 0.000006756886174\, t^5$$

$$SR_4(t) := 1000 - 2.832536016\, t + 4.894790067\, 10^{-18}\, t^{10} - 1.212908895\, 10^{-20}\, t^{11}$$
$$- 7.013587048\, 10^{-12}\, t^8 + 1.253065328\, 10^{-14}\, t^9 - 1.686781314\, 10^{-10}\, t^7 + 6.500365015\, 10^{-7}\, t^5$$
$$+ 4.351649407\, 10^{-8}\, t^6 - 0.0001518372617\, t^4 + 0.0007949459768\, t^3 - 0.3600797193\, t^2$$

$$ER_4(t) := 100 + 0.3091162900\, t - 4.894790067\, 10^{-18}\, t^{10} + 1.212908895\, 10^{-20}\, t^{11}$$
$$+ 7.013587048\, 10^{-12}\, t^8 - 1.253065328\, 10^{-14}\, t^9 + 1.682957102\, 10^{-10}\, t^7 - 3.399205847\, 10^{-7}\, t^5$$
$$- 4.366078966\, 10^{-8}\, t^6 + 0.0001762901912\, t^4 - 0.003802589815\, t^3 + 0.3568624462\, t^2$$

$$VR_4(t) := 10 + 2.523170548\, t + 0.003185801695\, t^2 + 0.003007623037\, t^3$$
$$- 0.00002447172426\, t^4 - 3.100941314\, 10^{-7}\, t^5 + 1.442955897\, 10^{-10}\, t^6$$
$$+ 3.824212013\, 10^{-13}\, t^7$$

$$\epsilon := 0.01001646435$$

$$0.01001646435$$

$$SH_5(t) := 10000 + 1.100516119\, 10^{-13}\, t^{15} - 2.609015819\, 10^{-14}\, t^{16} + 3.390847471\, 10^{-20}\, t^{17}$$
$$- 2.783105624\, 10^{-27}\, t^{18} - 6.188837251\, 10^{-33}\, t^{19} + 3.158599162\, 10^{-10}\, t^{13}$$
$$- 3.051573048\, 10^{-11}\, t^{14} + 0.000001512556070\, t^{12} + 0.00001246835491\, t^{10}$$
$$- 0.000005293581905\, t^{11} - 0.02895944869\, t^8 + 0.009317886471\, t^9 + 0.05357903686\, t^7$$
$$- 66.55014128\, t^5 - 0.07102577407\, t^6 + 100.1373657\, t^4 + 0.1409560350\, t$$
$$- 106.5246562\, t^3 + 62.04625101\, t^2$$

$$EH_5(t) := 50 - 1.100516119\, 10^{-13}\, t^{15} + 2.609015819\, 10^{-14}\, t^{16} - 3.390847471\, 10^{-20}\, t^{17}$$
$$+ 2.783105624\, 10^{-27}\, t^{18} + 6.188837251\, 10^{-33}\, t^{19} - 3.158599162\, 10^{-10}\, t^{13}$$
$$+ 3.051573048\, 10^{-11}\, t^{14} - 0.000001512556070\, t^{12} - 0.00001325565209\, t^{10}$$
$$+ 0.000005293582284\, t^{11} + 0.03496730844\, t^8 - 0.009315564836\, t^9 - 0.07257298646\, t^7$$
$$- 28.68278160\, t^5 + 0.09803280261\, t^6 - 110.5367256\, t + 60.65612062\, t^4 - 102.7719938\, t^3$$
$$+ 130.6549326\, t^2$$

$$UH_5(t) := 0.57 + 21.14733838\, t - 7.222395141\, 10^{-14}\, t^{11} - 4.138729505\, 10^{-20}\, t^{12}$$
$$- 4.430039978\, 10^{-7}\, t^9 + 1.502285199\, 10^{-7}\, t^{10} - 0.001146392684\, t^8 - 0.004744399997\, t^6$$
$$+ 0.003354552441\, t^7 + 5.329076778\, t^5 - 24.30256831\, t^2 - 11.28673594\, t^4$$

$$+ 19.13136870\, t^3$$

$$QH_5(t) := 0.43 + 88.76493490\, t - 3.062774999\, 10^{-13}\, t^{11} - 1.755096060\, 10^{-19}\, t^{12}$$
$$- 0.000001878631040\, t^9 + 6.370686538\, 10^{-7}\, t^{10} - 0.004861467810\, t^8 - 0.02376751648\, t^6$$
$$+ 0.01663199291\, t^7 + 134.0731650\, t^5 - 214.3114982\, t^2 - 216.7080658\, t^4 + 263.3287292\, t^3$$

$$RH_5(t) := 0.4684808200\, t + 45.08766178\, t^2 - 71.96909553\, t^3 + 66.14094985\, t^4$$
$$- 43.48459752\, t^5 + 0.001477878562\, t^6 - 0.0009747867247\, t^7 + 7.267534661\, 10^{-10}\, t^8$$

$$SR_5(t) := 1000 + 9.871199075\, 10^{-26}\, t^{15} + 3.787868776\, 10^{-29}\, t^{16} - 1.575636352\, 10^{-31}\, t^{17}$$
$$- 1.955820630\, 10^{-36}\, t^{18} + 7.062144074\, 10^{-38}\, t^{19} - 5.966777544\, 10^{-21}\, t^{13}$$
$$- 3.874735466\, 10^{-23}\, t^{14} + 7.321880811\, 10^{-19}\, t^{12} - 4.164193501\, 10^{-15}\, t^{10}$$
$$+ 4.697110898\, 10^{-17}\, t^{11} + 9.207328642\, 10^{-12}\, t^8 - 1.607388921\, 10^{-13}\, t^9$$
$$- 5.744022836\, 10^{-10}\, t^7 + 0.000001963539315\, t^5 + 8.179944610\, 10^{-8}\, t^6 - 2.832536016\, t$$
$$- 0.0001518372617\, t^4 + 0.0007949459768\, t^3 - 0.3600797193\, t^2$$

$$ER_5(t) := 100 - 9.871199075\, 10^{-26}\, t^{15} - 3.787868776\, 10^{-29}\, t^{16} + 1.575636352\, 10^{-31}\, t^{17}$$
$$+ 1.955820630\, 10^{-36}\, t^{18} - 7.062144074\, 10^{-38}\, t^{19} + 5.966777544\, 10^{-21}\, t^{13}$$
$$+ 3.874735466\, 10^{-23}\, t^{14} - 7.322136421\, 10^{-19}\, t^{12} + 4.195882270\, 10^{-15}\, t^{10}$$
$$- 4.695985586\, 10^{-17}\, t^{11} - 9.739306226\, 10^{-12}\, t^8 + 1.410314918\, 10^{-13}\, t^9$$
$$+ 7.321475373\, 10^{-10}\, t^7 - 0.000002857859618\, t^5 - 8.039505858\, 10^{-8}\, t^6$$
$$+ 0.0001762901912\, t^4 + 0.3091162900\, t - 0.003802589815\, t^3 + 0.3568624462\, t^2$$

$$VR_5(t) := 10 + 2.523170548\, t + 0.003185801695\, t^2 - 1.125312236\, 10^{-20}\, t^{11}$$
$$+ 2.556104421\, 10^{-23}\, t^{12} + 1.970740032\, 10^{-14}\, t^9 - 3.168876908\, 10^{-17}\, t^{10}$$
$$+ 5.319763886\, 10^{-13}\, t^8 - 1.403097453\, 10^{-9}\, t^6 - 1.577457690\, 10^{-10}\, t^7 + 8.944447204\, 10^{-7}\, t^5$$
$$- 0.00002447172426\, t^4 + 0.003007623037\, t^3$$

$$\epsilon := 0.0006660822742$$

$$0.0006660822742$$

$$SH_6(t) := 10000 + 1.787422463\, 10^{-13}\, t^{19} + 9.737940806\, 10^{-7}\, t^{14} - 2.469513805\, 10^{-7}\, t^{15}$$
$$+ 2.677709819\, 10^{-10}\, t^{16} - 8.348417685\, 10^{-11}\, t^{17} + 1.048279241\, 10^{-11}\, t^{18}$$
$$+ 0.000006132335486\, t^{12} - 0.000002588549684\, t^{13} + 0.002384963506\, t^{11}$$
$$+ 0.02376836284\, t^9 - 0.009533980139\, t^{10} + 0.06505857250\, t^7 - 0.04410770250\, t^8$$
$$+ 34.47188272\, t^6 + 100.1373657\, t^4 - 66.55014128\, t^5 - 106.5246562\, t^3 + 62.04625101\, t^2$$
$$+ 0.1409560350\, t - 5.951418528\, 10^{-58}\, t^{32} + 1.079360139\, 10^{-26}\, t^{27} - 1.853076171\, 10^{-32}\, t^{28}$$
$$+ 4.582483883\, 10^{-39}\, t^{29} + 5.284266896\, 10^{-45}\, t^{30} - 1.348347039\, 10^{-51}\, t^{31}$$

$$
\begin{aligned}
&- 7.517555510\ 10^{-23}\ t^{25} - 8.022764923\ 10^{-26}\ t^{26} + 4.734514363\ 10^{-22}\ t^{24} \\
&+ 6.993715287\ 10^{-18}\ t^{22} - 8.488062851\ 10^{-19}\ t^{23} - 4.599068576\ 10^{-14}\ t^{20} \\
&+ 6.117058147\ 10^{-15}\ t^{21}
\end{aligned}
$$

$$
\begin{aligned}
EH_6(t) := 50 &- 1.787422463\ 10^{-13}\ t^{19} - 9.737439612\ 10^{-7}\ t^{14} + 2.469468612\ 10^{-7}\ t^{15} \\
&- 2.677557022\ 10^{-10}\ t^{16} + 8.348076753\ 10^{-11}\ t^{17} - 1.048279240\ 10^{-11}\ t^{18} \\
&- 0.000007112294507\ t^{12} + 0.000002847018174\ t^{13} - 0.002382195263\ t^{11} \\
&- 0.03325034941\ t^{9} + 0.01160310428\ t^{10} - 0.1015188590\ t^{7} + 0.06753544597\ t^{8} \\
&+ 11.35265099\ t^{6} + 60.65612062\ t^{4} - 28.68278160\ t^{5} - 102.7719938\ t^{3} + 130.6549326\ t^{2} \\
&- 110.5367256\ t + 5.951418528\ 10^{-58}\ t^{32} - 1.079360139\ 10^{-26}\ t^{27} + 1.853076171\ 10^{-32}\ t^{28} \\
&- 4.582483883\ 10^{-39}\ t^{29} - 5.284266896\ 10^{-45}\ t^{30} + 1.348347039\ 10^{-51}\ t^{31} \\
&+ 7.517555510\ 10^{-23}\ t^{25} + 8.022764923\ 10^{-26}\ t^{26} - 4.734514363\ 10^{-22}\ t^{24} \\
&- 6.993715287\ 10^{-18}\ t^{22} + 8.488062851\ 10^{-19}\ t^{23} + 4.599068576\ 10^{-14}\ t^{20} \\
&- 6.117058147\ 10^{-15}\ t^{21}
\end{aligned}
$$

$$
\begin{aligned}
UH_6(t) := 0.57 &+ 21.14733838\ t + 1.311693111\ 10^{-34}\ t^{20} + 8.623541993\ 10^{-13}\ t^{15} \\
&- 2.915611111\ 10^{-15}\ t^{16} + 6.505504212\ 10^{-16}\ t^{17} - 7.985257411\ 10^{-22}\ t^{18} \\
&+ 6.209108647\ 10^{-29}\ t^{19} - 4.931979942\ 10^{-8}\ t^{13} - 9.563561421\ 10^{-12}\ t^{14} \\
&+ 1.869913833\ 10^{-7}\ t^{12} - 0.0003948738124\ t^{10} - 5.119425658\ 10^{-7}\ t^{11} - 0.003880064729\ t^{8} \\
&+ 0.001657460503\ t^{9} + 0.005992524977\ t^{7} + 5.329076778\ t^{5} - 2.099878400\ t^{6} \\
&- 11.28673594\ t^{4} + 19.13136870\ t^{3} - 24.30256831\ t^{2}
\end{aligned}
$$

$$
\begin{aligned}
QH_6(t) := 0.43 &+ 88.76493490\ t + 5.562449560\ 10^{-34}\ t^{20} + 3.656954281\ 10^{-12}\ t^{15} \\
&- 1.236412664\ 10^{-14}\ t^{16} + 2.758765654\ 10^{-15}\ t^{17} - 3.386279246\ 10^{-21}\ t^{18} \\
&+ 2.633074259\ 10^{-28}\ t^{19} - 2.091486906\ 10^{-7}\ t^{13} - 4.055584921\ 10^{-11}\ t^{14} \\
&+ 7.929676639\ 10^{-7}\ t^{12} - 0.001674056030\ t^{10} - 0.000002316200650\ t^{11} \\
&- 0.02169208748\ t^{8} + 0.008383206936\ t^{9} + 0.03396936357\ t^{7} + 134.0731650\ t^{5} \\
&- 66.47413953\ t^{6} - 216.7080658\ t^{4} + 263.3287292\ t^{3} - 214.3114982\ t^{2}
\end{aligned}
$$

$$
\begin{aligned}
RH_6(t) := 0.4684808200\ t &- 1.371303652\ 10^{-20}\ t^{13} - 2.592447326\ 10^{-14}\ t^{12} \\
&- 1.908172302\ 10^{-7}\ t^{10} + 5.882604425\ 10^{-8}\ t^{11} + 0.002106893905\ t^{8} - 0.0005486570907\ t^{9} \\
&- 0.003440439789\ t^{7} - 43.48459754\ t^{5} + 22.40060293\ t^{6} + 66.14094985\ t^{4} \\
&- 71.96909550\ t^{3} + 45.08766178\ t^{2}
\end{aligned}
$$

$$
\begin{aligned}
SR_6(t) := 1000 &+ 2.770271956\ 10^{-31}\ t^{19} + 2.581990745\ 10^{-22}\ t^{14} - 4.839029099\ 10^{-24}\ t^{15} \\
&+ 5.005380368\ 10^{-26}\ t^{16} - 4.838493012\ 10^{-28}\ t^{17} - 2.232355923\ 10^{-29}\ t^{18}
\end{aligned}
$$

$$
\begin{aligned}
&- 2.870037458\ 10^{-18}\ t^{12} + 3.530844533\ 10^{-20}\ t^{13} + 6.811968944\ 10^{-17}\ t^{11} \\
&- 3.451174775\ 10^{-13}\ t^{9} - 4.499384399\ 10^{-15}\ t^{10} - 5.252345222\ 10^{-10}\ t^{7} \\
&+ 2.698082797\ 10^{-11}\ t^{8} + 2.297115544\ 10^{-8}\ t^{6} - 0.0001518372617\ t^{4} \\
&+ 0.000001963539315\ t^{5} + 0.0007949459768\ t^{3} - 0.3600797193\ t^{2} - 2.832536016\ t \\
&- 1.631868722\ 10^{-65}\ t^{32} + 6.877304420\ 10^{-50}\ t^{27} - 7.323925946\ 10^{-53}\ t^{28} \\
&- 4.184644588\ 10^{-56}\ t^{29} + 6.020313901\ 10^{-59}\ t^{30} + 7.882485164\ 10^{-63}\ t^{31} \\
&- 3.823306016\ 10^{-44}\ t^{25} + 2.427368743\ 10^{-47}\ t^{26} + 6.446312132\ 10^{-42}\ t^{24} \\
&- 2.080337684\ 10^{-37}\ t^{22} + 2.236327914\ 10^{-39}\ t^{23} + 2.837699662\ 10^{-33}\ t^{20} \\
&- 3.487461847\ 10^{-35}\ t^{21}
\end{aligned}
$$

$$
\begin{aligned}
ER_6(t) \coloneqq\ & 100 - 2.770271982\ 10^{-31}\ t^{19} - 2.689772057\ 10^{-22}\ t^{14} + 4.773703642\ 10^{-24}\ t^{15} \\
&- 4.989778321\ 10^{-26}\ t^{16} + 4.839056491\ 10^{-28}\ t^{17} + 2.232333786\ 10^{-29}\ t^{18} \\
&+ 2.969000927\ 10^{-18}\ t^{12} - 3.388406342\ 10^{-20}\ t^{13} - 7.776760129\ 10^{-17}\ t^{11} \\
&+ 3.725162342\ 10^{-13}\ t^{9} + 4.143809717\ 10^{-15}\ t^{10} + 8.155689664\ 10^{-10}\ t^{7} \\
&- 2.930604232\ 10^{-11}\ t^{8} - 1.084408037\ 10^{-8}\ t^{6} + 0.0001762901912\ t^{4} \\
&- 0.000002857859618\ t^{5} - 0.003802589815\ t^{3} + 0.3568624462\ t^{2} + 0.3091162900\ t \\
&+ 1.631868722\ 10^{-65}\ t^{32} - 6.877304420\ 10^{-50}\ t^{27} + 7.323925946\ 10^{-53}\ t^{28} \\
&+ 4.184644588\ 10^{-56}\ t^{29} - 6.020313901\ 10^{-59}\ t^{30} - 7.882485164\ 10^{-63}\ t^{31} \\
&+ 3.823306016\ 10^{-44}\ t^{25} - 2.427368743\ 10^{-47}\ t^{26} - 6.446312132\ 10^{-42}\ t^{24} \\
&+ 2.080337684\ 10^{-37}\ t^{22} - 2.236327914\ 10^{-39}\ t^{23} - 2.837699573\ 10^{-33}\ t^{20} \\
&+ 3.487461847\ 10^{-35}\ t^{21}
\end{aligned}
$$

$$
\begin{aligned}
VR_6(t) \coloneqq\ & 10 - 8.929728075\ 10^{-41}\ t^{20} + 6.532545680\ 10^{-26}\ t^{15} - 1.560204709\ 10^{-28}\ t^{16} \\
&- 5.634789028\ 10^{-32}\ t^{17} + 2.213681539\ 10^{-34}\ t^{18} + 2.603197258\ 10^{-39}\ t^{19} \\
&- 1.424381958\ 10^{-21}\ t^{13} + 1.077813124\ 10^{-23}\ t^{14} - 9.896344567\ 10^{-20}\ t^{12} \\
&+ 3.555254135\ 10^{-16}\ t^{10} + 9.647983873\ 10^{-18}\ t^{11} + 2.325707344\ 10^{-12}\ t^{8} \\
&- 2.740023427\ 10^{-14}\ t^{9} - 2.903295340\ 10^{-10}\ t^{7} + 8.944447204\ 10^{-7}\ t^{5} - 1.213081328\ 10^{-8}\ t^{6} \\
&- 0.00002447172426\ t^{4} + 2.523170548\ t + 0.003007623037\ t^{3} + 0.003185801695\ t^{2}
\end{aligned}
$$

$$\epsilon \coloneqq 0.00003454391848$$

$$0.00003454391848$$

$$
\begin{aligned}
SH_7(t) \coloneqq\ & 10000 - 2.950438652\ 10^{-10}\ t^{19} + 0.000001080409095\ t^{14} - 4.420263215\ 10^{-7}\ t^{15} \\
&+ 2.073158730\ 10^{-7}\ t^{16} - 8.506695327\ 10^{-8}\ t^{17} + 2.149950849\ 10^{-8}\ t^{18} \\
&- 0.001996723361\ t^{12} + 0.0004107065586\ t^{13} + 0.006222329754\ t^{11} + 0.02905539591\ t^{9} \\
&- 0.01490618885\ t^{10} - 14.66847952\ t^{7} - 0.04773772173\ t^{8} + 34.47188272\ t^{6}
\end{aligned}
$$

$$+ 100.1373657\, t^4 - 66.55014128\, t^5 - 106.5246562\, t^3 + 62.04625101\, t^2 + 0.1409560350\, t$$
$$+ 1.095154395\, 10^{-97}\, t^{53} + 3.057263151\, 10^{-91}\, t^{52} - 1.581312477\, 10^{-84}\, t^{51}$$
$$- 2.407114654\, 10^{-78}\, t^{50} + 1.099047341\, 10^{-71}\, t^{49} - 6.761284324\, 10^{-68}\, t^{48}$$
$$- 2.918527287\, 10^{-59}\, t^{47} + 3.341711800\, 10^{-53}\, t^{46} - 1.160220734\, 10^{-47}\, t^{45}$$
$$+ 1.413011599\, 10^{-46}\, t^{44} + 6.806562367\, 10^{-44}\, t^{43} - 6.308825885\, 10^{-43}\, t^{42}$$
$$+ 2.085474126\, 10^{-39}\, t^{41} - 2.625038084\, 10^{-38}\, t^{40} - 1.317330019\, 10^{-35}\, t^{39}$$
$$+ 1.472081286\, 10^{-34}\, t^{38} - 9.544934079\, 10^{-32}\, t^{37} + 1.169496445\, 10^{-30}\, t^{36}$$
$$+ 6.367842741\, 10^{-28}\, t^{35} - 8.206124281\, 10^{-27}\, t^{34} + 5.644342851\, 10^{-26}\, t^{33}$$
$$+ 6.640836367\, 10^{-24}\, t^{32} - 7.809591187\, 10^{-19}\, t^{27} + 1.533587013\, 10^{-19}\, t^{28}$$
$$- 2.208836967\, 10^{-20}\, t^{29} + 4.949086469\, 10^{-22}\, t^{30} - 8.162460024\, 10^{-23}\, t^{31}$$
$$+ 2.768440307\, 10^{-15}\, t^{25} - 2.755057353\, 10^{-16}\, t^{26} - 1.467508626\, 10^{-14}\, t^{24}$$
$$+ 1.294600230\, 10^{-12}\, t^{22} + 5.581856480\, 10^{-14}\, t^{23} + 8.420376546\, 10^{-11}\, t^{20}$$
$$- 1.518444646\, 10^{-11}\, t^{21}$$

$$EH_7(t) := 50 + 2.962695050\, 10^{-10}\, t^{19} - 0.000001555699404\, t^{14} + 5.862312704\, 10^{-7}\, t^{15}$$
$$- 2.416020010\, 10^{-7}\, t^{16} + 8.510194094\, 10^{-8}\, t^{17} - 2.150981101\, 10^{-8}\, t^{18}$$
$$+ 0.002437416214\, t^{12} - 0.0004093950947\, t^{13} - 0.008805362489\, t^{11} - 0.04952243234\, t^9$$
$$+ 0.02361340617\, t^{10} - 3.895528341\, t^7 + 0.08261760002\, t^8 + 11.35265099\, t^6$$
$$+ 60.65612062\, t^4 - 28.68278160\, t^5 - 102.7719938\, t^3 + 130.6549326\, t^2 - 110.5367256\, t$$
$$- 1.095154395\, 10^{-97}\, t^{53} - 3.057263151\, 10^{-91}\, t^{52} + 1.581312477\, 10^{-84}\, t^{51}$$
$$+ 2.407114654\, 10^{-78}\, t^{50} - 1.099047341\, 10^{-71}\, t^{49} + 6.761284324\, 10^{-68}\, t^{48}$$
$$+ 2.918527287\, 10^{-59}\, t^{47} - 3.341711800\, 10^{-53}\, t^{46} + 1.160220734\, 10^{-47}\, t^{45}$$
$$- 1.413011599\, 10^{-46}\, t^{44} - 6.806562367\, 10^{-44}\, t^{43} + 6.308825885\, 10^{-43}\, t^{42}$$
$$- 2.085474126\, 10^{-39}\, t^{41} + 2.625038084\, 10^{-38}\, t^{40} + 1.317330019\, 10^{-35}\, t^{39}$$
$$- 1.472081286\, 10^{-34}\, t^{38} + 9.544934079\, 10^{-32}\, t^{37} - 1.169496445\, 10^{-30}\, t^{36}$$
$$- 6.367842741\, 10^{-28}\, t^{35} + 8.206124281\, 10^{-27}\, t^{34} - 5.644342851\, 10^{-26}\, t^{33}$$
$$- 6.640836367\, 10^{-24}\, t^{32} + 7.809591121\, 10^{-19}\, t^{27} - 1.533587004\, 10^{-19}\, t^{28}$$
$$+ 2.208836967\, 10^{-20}\, t^{29} - 4.949086469\, 10^{-22}\, t^{30} + 8.162460024\, 10^{-23}\, t^{31}$$
$$- 2.768440265\, 10^{-15}\, t^{25} + 2.755057289\, 10^{-16}\, t^{26} + 1.467500769\, 10^{-14}\, t^{24}$$
$$- 1.293982556\, 10^{-12}\, t^{22} - 5.581788931\, 10^{-14}\, t^{23} - 8.418391198\, 10^{-11}\, t^{20}$$
$$+ 1.517958138\, 10^{-11}\, t^{21}$$

$$UH_7(t) := -2.338710984\, 10^{-13}\, t^{19} + 8.649308843\, 10^{-8}\, t^{14} - 2.751730243\, 10^{-8}\, t^{15}$$
$$+ 6.542389600\, 10^{-9}\, t^{16} - 6.676395494\, 10^{-12}\, t^{17} + 1.965922707\, 10^{-12}\, t^{18}$$

$$
\begin{aligned}
&- 0.00008414553267\, t^{12} - 2.331001635\, 10^{-7}\, t^{13} + 0.0004501010551\, t^{11} \\
&+ 0.003216515056\, t^{9} - 0.001423162836\, t^{10} + 0.7122883564\, t^{7} - 0.005441081092\, t^{8} \\
&- 2.099878400\, t^{6} - 11.28673594\, t^{4} + 5.329076778\, t^{5} + 19.13136870\, t^{3} - 24.30256831\, t^{2} \\
&+ 0.57 + 21.14733838\, t + 7.644687273\, 10^{-60}\, t^{33} + 1.786096332\, 10^{-53}\, t^{32} \\
&+ 1.259544379\, 10^{-27}\, t^{27} - 1.634035605\, 10^{-28}\, t^{28} + 2.708622269\, 10^{-34}\, t^{29} \\
&- 6.474896977\, 10^{-41}\, t^{30} - 7.225638371\, 10^{-47}\, t^{31} - 8.027653172\, 10^{-24}\, t^{25} \\
&+ 1.225621771\, 10^{-24}\, t^{26} + 1.499168734\, 10^{-20}\, t^{24} - 1.178618081\, 10^{-16}\, t^{22} \\
&- 1.288941727\, 10^{-19}\, t^{23} - 3.788352539\, 10^{-15}\, t^{20} + 9.283329424\, 10^{-16}\, t^{21}
\end{aligned}
$$

$$
\begin{aligned}
QH_7(t) \coloneqq\ & -9.917687126\, 10^{-13}\, t^{19} + 4.042485563\, 10^{-7}\, t^{14} - 1.166848500\, 10^{-7}\, t^{15} \\
&+ 2.774350197\, 10^{-8}\, t^{16} - 2.831052529\, 10^{-11}\, t^{17} + 8.336433044\, 10^{-12}\, t^{18} \\
&- 0.0003563479225\, t^{12} - 0.000001141452408\, t^{13} + 0.002290336674\, t^{11} \\
&+ 0.01973203804\, t^{9} - 0.008148474682\, t^{10} + 27.51329383\, t^{7} - 0.03380968845\, t^{8} \\
&- 66.47413953\, t^{6} - 216.7080658\, t^{4} + 134.0731650\, t^{5} + 263.3287292\, t^{3} - 214.3114982\, t^{2} \\
&+ 0.43 + 88.76493490\, t + 3.241854897\, 10^{-59}\, t^{33} + 7.574234153\, 10^{-53}\, t^{32} \\
&+ 5.341304319\, 10^{-27}\, t^{27} - 6.929395721\, 10^{-28}\, t^{28} + 1.148635654\, 10^{-33}\, t^{29} \\
&- 2.745786155\, 10^{-40}\, t^{30} - 3.064150344\, 10^{-46}\, t^{31} - 3.404257862\, 10^{-23}\, t^{25} \\
&+ 5.197449942\, 10^{-24}\, t^{26} + 6.357470658\, 10^{-20}\, t^{24} - 4.998123091\, 10^{-16}\, t^{22} \\
&- 5.465968591\, 10^{-19}\, t^{23} - 1.606512967\, 10^{-14}\, t^{20} + 3.936747950\, 10^{-15}\, t^{21}
\end{aligned}
$$

$$
\begin{aligned}
RH_7(t) \coloneqq\ & 0.4684808200\, t + 1.337239528\, 10^{-29}\, t^{20} - 2.746239025\, 10^{-12}\, t^{15} \\
&+ 2.321537172\, 10^{-13}\, t^{16} - 7.387383024\, 10^{-16}\, t^{17} + 1.556748297\, 10^{-16}\, t^{18} \\
&- 1.810277951\, 10^{-22}\, t^{19} + 6.195669156\, 10^{-8}\, t^{13} - 1.517410931\, 10^{-8}\, t^{14} \\
&- 1.958588858\, 10^{-7}\, t^{12} + 0.0008493430987\, t^{10} - 0.0001545807168\, t^{11} \\
&+ 0.004295882520\, t^{8} - 0.002438850803\, t^{9} - 9.514471526\, t^{7} - 43.48459754\, t^{5} \\
&+ 22.40060293\, t^{6} + 66.14094985\, t^{4} - 71.96909550\, t^{3} + 45.08766178\, t^{2} \\
&+ 2.690437359\, 10^{-35}\, t^{21}
\end{aligned}
$$

$$
\begin{aligned}
SR_7(t) \coloneqq\ & 1000 - 3.249012829\, 10^{-31}\, t^{19} - 3.416665800\, 10^{-22}\, t^{14} + 1.576311760\, 10^{-24}\, t^{15} \\
&+ 3.509809218\, 10^{-25}\, t^{16} - 8.427776152\, 10^{-27}\, t^{17} + 5.736039014\, 10^{-29}\, t^{18} \\
&- 3.459935710\, 10^{-18}\, t^{12} + 7.838603475\, 10^{-20}\, t^{13} + 4.500755770\, 10^{-19}\, t^{11} \\
&- 5.437621805\, 10^{-13}\, t^{9} - 2.440650777\, 10^{-16}\, t^{10} - 5.298608972\, 10^{-11}\, t^{7} \\
&+ 3.602252962\, 10^{-11}\, t^{8} + 2.297115544\, 10^{-8}\, t^{6} - 0.0001518372617\, t^{4} \\
&+ 0.000001963539315\, t^{5} + 0.0007949459768\, t^{3} - 0.3600797193\, t^{2} - 2.832536016\, t \\
&- 7.953670855\, 10^{-111}\, t^{53} + 4.152102830\, 10^{-108}\, t^{52} + 5.086749504\, 10^{-104}\, t^{51}
\end{aligned}
$$

$$
\begin{aligned}
&- 3.794181303\ 10^{-101}\ t^{50} - 1.290552434\ 10^{-97}\ t^{49} + 1.282693236\ 10^{-94}\ t^{48} \\
&+ 1.532408382\ 10^{-91}\ t^{47} - 2.100321475\ 10^{-88}\ t^{46} - 6.528424248\ 10^{-86}\ t^{45} \\
&+ 1.713752215\ 10^{-82}\ t^{44} - 2.405427787\ 10^{-80}\ t^{43} - 5.878955443\ 10^{-77}\ t^{42} \\
&+ 2.473159946\ 10^{-74}\ t^{41} + 2.733741952\ 10^{-72}\ t^{40} - 2.249842145\ 10^{-69}\ t^{39} \\
&- 8.159477512\ 10^{-68}\ t^{38} + 7.168957468\ 10^{-65}\ t^{37} + 8.795524999\ 10^{-64}\ t^{36} \\
&- 1.218440331\ 10^{-60}\ t^{35} + 4.058878123\ 10^{-60}\ t^{34} + 1.228094670\ 10^{-56}\ t^{33} \\
&- 2.502035819\ 10^{-55}\ t^{32} + 3.251284547\ 10^{-45}\ t^{27} - 1.279349119\ 10^{-47}\ t^{28} \\
&- 8.231876979\ 10^{-50}\ t^{29} + 2.890280506\ 10^{-51}\ t^{30} - 6.037751522\ 10^{-53}\ t^{31} \\
&- 1.484620441\ 10^{-41}\ t^{25} - 5.863143348\ 10^{-44}\ t^{26} + 7.612713183\ 10^{-40}\ t^{24} \\
&- 1.494616367\ 10^{-36}\ t^{22} - 2.669565582\ 10^{-38}\ t^{23} - 1.880317856\ 10^{-32}\ t^{20} \\
&+ 4.783230820\ 10^{-34}\ t^{21}
\end{aligned}
$$

$$
\begin{aligned}
ER_7(t) &:= 100 + 2.951889266\ 10^{-31}\ t^{19} + 4.028175512\ 10^{-22}\ t^{14} - 1.122440401\ 10^{-24}\ t^{15} \\
&- 3.585238215\ 10^{-25}\ t^{16} + 8.501998478\ 10^{-27}\ t^{17} - 5.804025241\ 10^{-29}\ t^{18} \\
&+ 3.624264993\ 10^{-18}\ t^{12} - 8.416582636\ 10^{-20}\ t^{13} - 9.958981803\ 10^{-18}\ t^{11} \\
&+ 6.262504687\ 10^{-13}\ t^{9} - 6.994925288\ 10^{-16}\ t^{10} + 9.121337200\ 10^{-11}\ t^{7} \\
&- 3.862053099\ 10^{-11}\ t^{8} - 1.084408037\ 10^{-8}\ t^{6} + 0.0001762901912\ t^{4} \\
&- 0.000002857859618\ t^{5} - 0.003802589815\ t^{3} + 0.3568624462\ t^{2} + 0.3091162900\ t \\
&+ 7.953670855\ 10^{-111}\ t^{53} - 4.152102830\ 10^{-108}\ t^{52} - 5.086749504\ 10^{-104}\ t^{51} \\
&+ 3.794181303\ 10^{-101}\ t^{50} + 1.290552434\ 10^{-97}\ t^{49} - 1.282693236\ 10^{-94}\ t^{48} \\
&- 1.532408382\ 10^{-91}\ t^{47} + 2.100321475\ 10^{-88}\ t^{46} + 6.528424248\ 10^{-86}\ t^{45} \\
&- 1.713752215\ 10^{-82}\ t^{44} + 2.405427787\ 10^{-80}\ t^{43} + 5.878955443\ 10^{-77}\ t^{42} \\
&- 2.473159946\ 10^{-74}\ t^{41} - 2.733741952\ 10^{-72}\ t^{40} + 2.249842145\ 10^{-69}\ t^{39} \\
&+ 8.159477512\ 10^{-68}\ t^{38} - 7.168957468\ 10^{-65}\ t^{37} - 8.795524999\ 10^{-64}\ t^{36} \\
&+ 1.218440331\ 10^{-60}\ t^{35} - 4.058878123\ 10^{-60}\ t^{34} - 1.228094670\ 10^{-56}\ t^{33} \\
&+ 2.502035819\ 10^{-55}\ t^{32} - 3.251261811\ 10^{-45}\ t^{27} + 1.279355331\ 10^{-47}\ t^{28} \\
&+ 8.231870592\ 10^{-50}\ t^{29} - 2.890280541\ 10^{-51}\ t^{30} + 6.037751527\ 10^{-53}\ t^{31} \\
&+ 1.485272524\ 10^{-41}\ t^{25} + 5.859424594\ 10^{-44}\ t^{26} - 7.589148809\ 10^{-40}\ t^{24} \\
&+ 1.454527993\ 10^{-36}\ t^{22} + 2.646691817\ 10^{-38}\ t^{23} + 1.915346560\ 10^{-32}\ t^{20} \\
&- 4.749058160\ 10^{-34}\ t^{21}
\end{aligned}
$$

$$
\begin{aligned}
VR_7(t) &:= 10 + 2.971235602\ 10^{-32}\ t^{19} - 6.114842764\ 10^{-23}\ t^{14} - 4.538893223\ 10^{-25}\ t^{15} \\
&+ 7.542797719\ 10^{-27}\ t^{16} - 7.422209700\ 10^{-29}\ t^{17} + 6.798623467\ 10^{-31}\ t^{18} \\
&- 1.643493862\ 10^{-19}\ t^{12} + 5.779981928\ 10^{-21}\ t^{13} + 9.508100664\ 10^{-18}\ t^{11}
\end{aligned}
$$

$$
\begin{aligned}
&- 8.249477857\ 10^{-14}\ t^9 + 9.436261880\ 10^{-16}\ t^{10} - 3.818366531\ 10^{-11}\ t^7 \\
&+ 2.598908312\ 10^{-12}\ t^8 - 1.213081328\ 10^{-8}\ t^6 - 0.00002447172426\ t^4 + 8.944447204\ 10^{-7}\ t^5 \\
&+ 0.003007623037\ t^3 + 0.003185801695\ t^2 + 2.523170548\ t + 1.250555397\ 10^{-68}\ t^{33} \\
&- 6.229380228\ 10^{-66}\ t^{32} - 2.273545487\ 10^{-50}\ t^{27} - 6.211433982\ 10^{-53}\ t^{28} \\
&+ 6.386715972\ 10^{-56}\ t^{29} + 3.527515900\ 10^{-59}\ t^{30} - 4.911216716\ 10^{-62}\ t^{31} \\
&- 6.520831500\ 10^{-45}\ t^{25} + 3.718753302\ 10^{-47}\ t^{26} - 2.356437359\ 10^{-42}\ t^{24} \\
&+ 4.008837393\ 10^{-38}\ t^{22} + 2.287376508\ 10^{-40}\ t^{23} - 3.502870408\ 10^{-34}\ t^{20} \\
&- 3.417265926\ 10^{-36}\ t^{21}
\end{aligned}
$$

$$\epsilon := 0.000001473385323 \qquad \textbf{(28)}$$

Multistage Picard Iterative Method (MPIM)

This appendix presents the Maple code used for the implementation of the MPIM in solving the Monkeypox transmission model. The implementation includes the parameter setup, initial conditions, and the multistage iterative procedure applied to obtain improved approximate numerical solutions of the model.

```
> restart; Digits := 16;
```

$$Digits := 16 \quad (1)$$

```
> ############## Parameter Setup
> alpha1 := 0.423890;  # transmission rate of humans from exposed
  to infectious class
```

$$\alpha 1 := 0.423890 \quad (2)$$

```
> alpha2 := 1.797575;  # rate of identification of suspected case
```

$$\alpha 2 := 1.797575 \quad (3)$$

```
> alpha3 := 0.025289;  # transmission rate of rodents from exposed
  to infectious class
```

$$\alpha 3 := 0.025289 \quad (4)$$

```
> beta1 := 0.011503;   # contact rate between rodents and humans
```

$$\beta 1 := 0.011503 \quad (5)$$

```
> beta2 := 0.747322;   # contact rate between humans
```

$$\beta 2 := 0.747322 \quad (6)$$

```
> beta3 := 0.321102;   # contact rate between rodents
```

$$\beta 3 := 0.321102 \quad (7)$$

```
> Lambdah := 0.34857;  # recruitment rate of humans
```

$$Lambdah := 0.34857 \quad (8)$$

```
> Lambdar := 0.60822;  # recruitment rate of rodents
```

$$Lambdar := 0.60822 \quad (9)$$

```
> phi := 1.575454;     # proportion of humans not detected after
  diagnosis
```

$$\phi := 1.575454 \quad (10)$$

```
> rho := 0.067689;     # recovery rate
```

$$\rho := 0.067689 \quad (11)$$

```
> muh := 1/(79*365);   # natural death rates of humans
```

$$muh := \frac{1}{28835} \quad (12)$$

```
> mur := 1/(5*365);    # natural death rates of rodents
```

(13)

$$mur := \frac{1}{1825} \tag{13}$$

```
> deltah := 0.015016;  # disease induced death rates of humans
```
$$deltah := 0.015016 \tag{14}$$

```
> deltar := 0.000025;  # disease induced death rates of rodents
```
$$deltar := 0.000025 \tag{15}$$

```
> tau := 0.999763;     # rate of transmission from isolation to
  recovered class
```
$$\tau := 0.999763 \tag{16}$$

```
> ################## Initial Condition
> NH := 10000 + 50 + 0.57 + 0.43 + 0;
```
$$NH := 10051.00 \tag{17}$$

```
> SH[0] := 10000;  # susceptible class human
```
$$SH_0 := 10000 \tag{18}$$

```
> EH[0] := 50;     # exposed class human
```
$$EH_0 := 50 \tag{19}$$

```
> UH[0] := 0.57;   # infected class human
```
$$UH_0 := 0.57 \tag{20}$$

```
> QH[0] := 0.43;   # isolation class
```
$$QH_0 := 0.43 \tag{21}$$

```
> RH[0] := 0;      # recovery class
```
$$RH_0 := 0 \tag{22}$$

```
> NR := 1000 + 100 + 10;
```
$$NR := 1110 \tag{23}$$

```
> SR[0] := 1000;   # susceptible class rodent
```
$$SR_0 := 1000 \tag{24}$$

```
> ER[0] := 100;    # exposed class rodent
```
$$ER_0 := 100 \tag{25}$$

```
> VR[0] := 10;     # infected class rodent
```
(26)

$$VR_0 := 10 \tag{26}$$

```
> ###Apply Multistage Picard Iterative Method to Monkeypox
  Transmission Model
> for m from 0 to 120 by 0.01 do
> for n from 0 to 3 do
> SH[n+1] := SH[0] + int(Lambdah - (beta1*VR[n] + beta2*UH[n])*SH
  [n]/NH - muh*SH[n] + phi*QH[n], t=m..t);
> EH[n+1] := EH[0] + int((beta1*VR[n] + beta2*UH[n])*SH[n]/NH -
  (alpha1 + alpha2 + muh)*EH[n], t=m..t);
> UH[n+1] := UH[0] + int(alpha1*EH[n] - (muh + deltah + rho)*UH[n],
  t=m..t);
> QH[n+1] := QH[0] + int(alpha2*EH[n] - (phi + tau + muh + deltah)*
  QH[n], t=m..t);
> RH[n+1] := RH[0] + int(rho*UH[n] + tau*QH[n] - muh*RH[n], t=m..t)
  ;
> SR[n+1] := SR[0] + int(Lambdar - beta3*SR[n]*VR[n]/NR - mur*SR
  [n], t=m..t);
> ER[n+1] := ER[0] + int(beta3*SR[n]*VR[n]/NR - (mur + alpha3)*ER
  [n], t=m..t);
> VR[n+1] := VR[0] + int(alpha3*ER[n] - (mur + deltar)*VR[n], t=m..
  t);
> end do;
> a := simplify( eval(SH[k], k=4) );
> b := simplify( eval(EH[k], k=4) );
> c := simplify( eval(UH[k], k=4) );
> d := simplify( eval(QH[k], k=4) );
> e := simplify( eval(RH[k], k=4) );
> f := simplify( eval(SR[k], k=4) );
> g := simplify( eval(ER[k], k=4) );
> h := simplify( eval(VR[k], k=4) );
> masa:=m+0.01:
> SSH := evalf(subs(t=m+0.01, SH[n]));
> EEH := evalf(subs(t=m+0.01, EH[n]));
> UUH := evalf(subs(t=m+0.01, UH[n]));
> QQH := evalf(subs(t=m+0.01, QH[n]));
> RRH := evalf(subs(t=m+0.01, RH[n]));
> SSR := evalf(subs(t=m+0.01, SR[n]));
> EER := evalf(subs(t=m+0.01, ER[n]));
> VVR := evalf(subs(t=m+0.01, VR[n]));
> SH[0] := SSH:
> EH[0] := EEH:
> UH[0] := UUH:
> QH[0] := QQH:
> RH[0] := RRH:
> SR[0] := SSR:
> ER[0] := EER:
> VR[0] := VVR:
> end do;
```

$$a := 10000. + 0.1409560349907640\, t + 1.171642504605577\, 10^{-18} t^{11} + 1.874220825616563\, 10^{-12} t^{10} - 0.00000354404491894998\, t^{9} + 0.00000940582694875254\, t^{8} + 0.02163565633684853\, t^{7} - 0.04678489576108552\, t^{6} + 0.05595345217575974\, t^{5} + 100.1373657534783\, t^{4} - 106.5246561885549\, t^{3} + 62.04625103250735\, t^{2}$$

$$b := 50. - 110.5367255817672\, t - 1.171642504605577\, 10^{-18}\, t^{11} - 1.874220825616563\, 10^{-12}\, t^{10} + 0.000003544044918949989\, t^{9} - 0.000009405826948752541\, t^{8} - 0.02163566341065968\, t^{7} + 0.05508692021588728\, t^{6} - 0.06675257393575918\, t^{5} + 60.65612063683533\, t^{4} - 102.7719938130963\, t^{3} + 130.6549326536542\, t^{2}$$

$$c := 0.57 + 21.14733838235651\, t - 24.30256830953833\, t^{2} + 1.349792954475336\, 10^{-9}\, t^{7} - 0.001584155116621652\, t^{6} + 0.002062510115517784\, t^{5} - 11.28673594597681\, t^{4} + 19.13136870984034\, t^{3}$$

$$d := 0.43 + 88.76493489756719\, t - 214.3114982384450\, t^{2} + 5.724018189013666\, 10^{-9}\, t^{7} - 0.006717869338180110\, t^{6} + 0.008729731312247464\, t^{5} - 216.7080658844422\, t^{4} + 263.3287293068882\, t^{3}$$

$$e := 0.000006756886176651958\, t\, \left(t + 9.788674176267888\, 10^{6}\right)\, \left(t + 0.01022208797018407\right)\, \left(t^{2} - 1.098339090736330\, t + 0.6929178115852673\right)$$

$$f := 1000. - 2.832536016290263\, t - 1.212908896480071\, 10^{-20}\, t^{11} + 4.894790083146175\, 10^{-18}\, t^{10} + 1.253065327646832\, 10^{-14}\, t^{9} - 7.013587047505857\, 10^{-12}\, t^{8} - 1.686781313858093\, 10^{-10}\, t^{7} + 4.351649406914840\, 10^{-8}\, t^{6} + 6.500365017317256\, 10^{-7}\, t^{5} - 0.0001518372617462733\, t^{4} + 0.0007949459764006202\, t^{3} - 0.3600797192610615\, t^{2}$$

$$g := 100. + 0.3091162902628660\, t + 1.212908896480071\, 10^{-20}\, t^{11} - 4.894790083146175\, 10^{-18}\, t^{10} - 1.253065327646832\, 10^{-14}\, t^{9} + 7.013587047505857\, 10^{-12}\, t^{8} + 1.682957101842196\, 10^{-10}\, t^{7} - 4.366078965858888\, 10^{-8}\, t^{6} - 3.399205851498398\, 10^{-7}\, t^{5} + 0.0001762901912025089\, t^{4} - 0.003802589813926236\, t^{3} + 0.3568624461989772\, t^{2}$$

$$h := 10. + 2.523170547945205\, t + 0.003185801698202726\, t^{2} + 3.824212015897696\, 10^{-13}\, t^{7} + 1.442955894404778\, 10^{-10}\, t^{6} - 3.100941313346056\, 10^{-7}\, t^{5} - 0.00002447172425088317\, t^{4} + 0.003007623037372447\, t^{3}$$

$$masa := 0.01$$

$$SSH := 10000.00750866217$$

$$EEH := 48.90759607200848$$

$$UUH := 0.7790621454941663$$

$$QQH := 1.296479360801342$$

$$RRH := 0.009122266691405502$$

$$SSR := 999.9716386326586$$

$$EER := 100.0031268453464$$

$$VVR := 10.02523202706700$$

$$SH_{0} := 10000.00750866217$$

APPENDIX G

Maple code for COVID-19 Model

Picard Iterative Method (PIM)

This appendix presents the Maple code for applying the PIM to the COVID-19 model. The implementation includes the parameter definitions, initial values, and the iterative procedure used to compute the approximate solutions of the system.

```
> restart; with(Plot):


> ############## Parameter Setup
> Digits := 10:
> beta := 0.22:
> local_gamma := 0.09:


> ############## Initial Value
> s[0](t) := 0.999:
> u[0](t) := 0.001:
> r[0](t) := 0.0:


> ############## Apply PIM to COVID-19 Model
> for n from 0 to 15 do;
> s[n+1](t) := s[0](t) - int(beta*s[n](t)*u[n](t), t);
> u[n+1](t) := u[0](t) + int(beta*s[n](t)*u[n](t) - local_gamma
                *u[n](t), t);
> r[n+1](t) := r[0](t) + int(local_gamma*u[n](t), t);
> epsilon := evalf(subs(t=0.1, abs(s[n+1](t)-s[n](t))));
> if epsilon <= 10^(-10) then break else print(epsilon) end if;
> end;
```

Multistgae Picard Iterative Method (MPIM)

This appendix presents the Maple code for implementing the MPIM in solving the COVID-19 model. The code includes the parameter setup, initial values, and the multistage iterative process applied over partitioned time intervals to obtain improved approximate solutions.

```
> restart;with(Plot):
> #############Parameter Setup
> Digits:=10:
> beta := 0.22:
> local_gamma := 0.09:
> #############Initial Value
> s[0](t) := 0.999:
> u[0](t) := 0.001:
> r[0](t) := 0.0:
> #############Apply MPIM to COVID-19 Model
> for m from 0 to 70 by 0.01 do:
> for n from 0 to 3 do:
> s[n+1](t):= s[0](t) - int(beta*s[n](t)*u[n](t), t=m..t):
> u[n+1](t):= u[0](t) + int( beta*s[n](t)*u[n](t) - local_gamma
              *u[n](t), t=m..t):
> r[n+1](t):= r[0](t) + int( local_gamma*u[n](t), t=m..t):
> end do:masa:=m+0.01:
> a := simplify( eval(s[k](t), k=4) );
> b := simplify( eval(u[k](t), k=4) );
> c := simplify( eval(r[k](t), k=4) );
> ss:=evalf(subs(t=m+0.01,s[n](t))):
> uu:=evalf(subs(t=m+0.01,u[n](t))):
> rr:=evalf(subs(t=m+0.01,r[n](t))):
> s[0](t):=ss:u[0](t):=uu:r[0](t):=rr:
> print(s[0](t),u[0](t),r[0](t));
> end do;
```

APPENDIX H

Maple code for Michaelis-Menten System

Picard Iterative Method (PIM)

This appendix presents the Maple code for applying the PIM to the Michaelis-Menten system. The implementation includes the parameter definitions, initial conditions, and the iterative procedure used to obtain the approximate solutions of the system.

```
> restart;Digits:=16:

> #############Parameter Setup
> alpha:= 0.375:
> beta:= 1:
> epsilon:= 0.1:

> ###################Initial Condition
> x[0](t):= 1:
> y[0](t):= 0:

> ##Apply Picard Iterative Method to Michaelis Menten Equation
> for n from 0 to 15 do;
> x[n+1](t) := x[0](t) + int(-x[n](t) + (beta - alpha)*y[n](t)
               + x[n](t)*y[n](t), t):
> y[n+1](t) := y[0](t) + int(((1)/(epsilon))*(x[n](t) - beta
               *y[n](t) - x[n](t)*y[n](t)), t):
> epsilon:=evalf(subs(t=0.1,abs(x[n+1](t)-x[n](t))));
> if epsilon <=10^(-2) then break else print(epsilon) end if
> end;
```

Multistage Picard Iterative Method (MPIM)

This appendix provides the Maple implementation of the MPIM for the Michaelis-Menten system, including the parameter initialization and the multistage updating process.

```
> restart;Digits:=16:
> #############Parameter Setup
> alpha:= 0.375:
> beta:= 1:
> epsilon:= 0.1:
> ###################Initial Condition
> x[0](t):= 1:
> y[0](t):= 0:
> ###Apply MPIM to Michaelis Menten Equation
> for m from 0 to 1 by 0.01 do:
> for n from 0 to 4 do:
> x[n+1](t) := x[0](t) + int(-x[n](t) + (beta - alpha)*y[n](t)
               + x[n](t)*y[n](t), t=m..t):
> y[n+1](t) := y[0](t) +int(((1)/(epsilon))*(x[n](t) - beta
               *y[n](t) - x[n](t)*y[n](t)), t=m..t):
> end do:
> a:=eval(x[k](t),k=5):
> b:=eval(y[k](t),k=5):
> masa:=m+0.01:
> xx:=evalf(subs(t=m+0.01,x[n](t))):
> yy:=evalf(subs(t=m+0.01,y[n](t))):
> x[0](t):=xx:
> y[0](t):=yy:
> print(x[0](t),y[0](t));
> end do;
```

APPENDIX I

Maple code for Multispecies Lotka-Volterra

Picard Iterative Method (PIM) for Solving One Species

This appendix provides the Maple implementation of the PIM for solving one species in the multispecies Lotka–Volterra model, including the parameter definitions and iterative computation procedure.

```
> restart;Digits := 16:
> ###Parameters
> beta:=1:
> alpha:=-3:
> x[0](t):=0.1:
> for n from 0 to 10 do;
> x[n+1](t):= x[0](t) + int(beta*x[n](t) + alpha*x[n](t)^2, t);
> epsilon:=evalf(subs(t=0.1,abs(x[n+1](t)-x[n](t))));
> if epsilon <=10^(-10) then break else print(epsilon) end if
> end do;
```

Multistage Picard Iterative Method (MPIM) for Solving One Species

This appendix provides the Maple implementation of the MPIM for solving one species in the multispecies Lotka–Volterra model, including the parameter initialization and multistage updating procedure.

```
> restart;Digits := 16:
> ###Parameters
> beta:=1:
> alpha:=-3:
> x[0](t):=0.1:
> for m from 0 by 0.01 to 7 do
> for n from 0 to 5 do
> x[n+1](t):= x[0](t) + int(beta*x[n](t)+ alpha*x[n](t)^2,
              s=m..t);
```

```
> end do:
> a := simplify(eval(x[k](t), k=6)):
> masa := m + 0.01:
> xx := evalf(subs(t = masa, x[n](t))):
> x[0](t) := xx:
> print(x[0](t));
> end do;
```

Picard Iterative Method for Solving Two Species

This appendix provides the Maple implementation of the PIM for solving two species in the multispecies Lotka–Volterra system, including the parameter setup and iterative computation process.

```
> restart;Digits := 16:
> ###Parameters
> b[1]:=0.1: a[11]:=-0.0014: a[12]:=-0.0012:
> b[2]:=0.08: a[21]:=-0.0009: a[22]:=-0.001:
> ###Intial Value
> x[0](t):=4:
> y[0](t):=10:
> for n from 0 to 10 do;
> x[n+1](t):= x[0](t) + int(x[n](t)*(b[1] + a[11]*x[n](t)
              + a[12]*y[n](t)), t);
> y[n+1](t):= y[0](t) + int(y[n](t)*(b[2] + a[21]*x[n](t)
              + a[22]*y[n](t)), t);
> epsilon:=evalf(subs(t=0.1, abs(x[n+1](t)-x[n](t))));
> if epsilon <=10^(-10) then break else print(epsilon) end if
> end do;
```

Multistage Picard Iterative Method for Solving Two Species

This appendix provides the Maple implementation of the MPIM for solving two species in the multispecies Lotka–Volterra system, including the parameter initialization and multistage updating process.

```
> restart;Digits := 16:
> ###Parameters
> b[1]:=0.1: a[11]:=-0.0014: a[12]:=-0.0012:
> b[2]:=0.08: a[21]:=-0.0009: a[22]:=-0.001:
> ###Intial Value
> x[0](t):=4:
> y[0](t):=10:
> ###Multistage PIM
> for m from 0 to 70 by 0.01 do
> for n from 0 to 4 do
> x[n+1](t):= x[0](t) + int(x[n](t)*(b[1] + a[11]*x[n](t)
              + a[12]*y[n](t)), s=m..t);
> y[n+1](t):= y[0](t) + int(y[n](t)*(b[2] + a[21]*x[n](t)
              + a[22]*y[n](t)), s=m..t);
> end do:
> a := simplify(eval(x[k](t), k=5));
> b := simplify(eval(y[k](t), k=5));
> masa:=m+0.01:
> xx:=evalf(subs(t=m+0.01, x[n](t))):
> yy:=evalf(subs(t=m+0.01, y[n](t))):
> x[0](t):=xx:
> y[0](t):=yy:
> print(x[0](t),y[0](t));
> end do;
```

Picard Iterative Method for Solving Three Species

This appendix presents the Maple code for applying the PIM to solve three interacting species in the multispecies Lotka–Volterra model. The implementation includes the parameter definitions, initial values for all three species, and the iterative procedure used to obtain the approximate solutions.

```
> restart;Digits := 16:
> ###Parameters
> beta:=0.1: alpha:=0.1:
> ###Initial Value
> x[0](t):=0.2: y[0](t):=0.3: z[0](t):=0.5:
> ###Apply PIM to Three Species
> for n from 0 to 10 do;
> x[n+1](t):= x[0](t) + int(x[n](t)*(1 - x[n](t) - alpha*y[n](t)
              - beta*z[n](t)), t=0..t);
> y[n+1](t):= y[0](t) + int(y[n](t)*(1 - beta*x[n](t) - y[n](t)
> z[n+1](t):= z[0](t) + int(z[n](t)*(1 - alpha*x[n](t) - beta
> epsilon:=evalf(subs(t=0.1,abs(x[n+1](t)-x[n](t))));
> if epsilon <=10^(-10) then break else print(epsilon) end if
> end do;
```

Multistage Picard Iterative Method for Solving Three Species

This appendix presents the Maple code for implementing the MPIM to solve three interacting species in the multispecies Lotka–Volterra model. The implementation includes the parameter setup, initial values for all three species, and the multistage iterative process applied over partitioned time intervals to obtain improved approximate solutions.

```
> restart;Digits := 16:
> ###Parameters
> beta:=0.1: alpha:=0.1:
> ###Initial Value
> x[0](t):=0.2: y[0](t):=0.3: z[0](t):=0.5:
```

```
> ###Apply MPIM to Three Species
> for m from 0 to 7 by 0.01 do
> for n from 0 to 5 do
> x[n+1](t):= x[0](t) + int(x[n](t)*(1 - x[n](t) - alpha*y[n](t)
> y[n+1](t):= y[0](t) + int(y[n](t)*(1 - beta*x[n](t) - y[n](t)
> z[n+1](t):= z[0](t) + int(z[n](t)*(1 - alpha*x[n](t) - beta
> end do:
> masa:=m+0.01:
> a := simplify(eval(x[k](t), k=6));
> b := simplify(eval(y[k](t), k=6));
> c := simplify(eval(z[k](t), k=6));
> xx:=evalf(subs(t=m+0.01, x[n](t))):
> yy:=evalf(subs(t=m+0.01, y[n](t))):
> zz:=evalf(subs(t=m+0.01, z[n](t))):
> x[0](t):=xx:
> y[0](t):=yy:
> z[0](t):=zz:
> print(x[0](t),y[0](t),z[0](t));
> end do;
```

AUTHOR'S PROFILE

Ahmad Bazli Khairuddin graduated from Universiti Teknologi MARA (UiTM) Seremban in 2024 with a Bachelor of Science (Hons.) in Mathematics, majoring in Mathematics with a minor in Big Data Analytics, and is currently pursuing a Master of Science in Mathematics (By Research) at the same institution. His postgraduate research focuses on Numerical Methods, specifically the development and modification of the Picard Iterative Method for solving nonlinear differential equations. In his current work, he applies this modified Picard Iterative Method to analyze Monkeypox disease dynamics using the SEIR epidemiological model, while conducting comparative performance studies against traditional Runge–Kutta methods.

LIST OF PUBLICATIONS

Khairuddin, A. B., Selamat, M. S., & Yacob, N. A. (2025). Solution of Monkeypox Transmission Model using Picard Iterative Method. *Mathematical Sciences and Informatics Journal*, 6(2), 43–59.

Khairuddin, A. B., Selamat, M. S., Yacob, N. A. M., & Saleh, S. H. M. (2025). A Novel Approach to Monkeypox Transmission Modeling via the Modified Picard Iterative Method. *Proceedings of the 1st International Conference and Innovation on Education, Science & Technology (ICIEST 2025)*, Universiti Sains Islam Malaysia, 93–101. eISBN: 978-629-95846-0-5.

Khairuddin, A. B., Selamat, M. S., & Yacob, N. A. (2026). Simulation of Time-Dependent Enzyme Kinetics by Multistage Picard Iterative Method. *Journal of Quality Measurement and Analysis*, 22(2), 241–256.