A STUDY ON SOLUTION OF MATRIX RICCATI DIFFERENTIAL EQUATIONS USING ANT COLONY PROGRAMMING AND SIMULINK

MOHD ZAHURIN MOHAMED KAMALI

THESIS SUBMITTED IN FULFILMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

INSTITUTE OF MATHEMATICAL SCIENCES , FACULTY OF SCIENCE,

UNIVERSITY OF MALAYA

KUALA LUMPUR

2015

ABSTRACT

Swarm intelligence is a modern artificial intelligence discipline that is concerned with the design and optimization of multiagent systems with applications in robotics. This non-traditional approach is fundamentally different from the traditional approaches. Instead of a sophisticated controller that governs the global behavior of the system, the swarm intelligence principle is based on many unsophisticated entities (for example such as ants, termites, bees etc.) that cooperate and interact in order to exhibit a desired behavior. In this thesis, we implement the modified ant colony programming (ACP) algorithm for solving the matrix Riccati differential equation (MRDE). Solving MRDE, especially nonlinear MRDE is the central issue in optimal control theory. It has been found that by implementing the ACP algorithm, the solution predicted is approximately close or similar to the exact solution. Besides that, we compared our present work with numerical solution obtained by Runge-Kutta fourth order (RK4) and a non-traditional method such as the genetic programming (GP). Furthermore, in this work, we also showed the implementation of the Simulink, for solving the MRDE in order to get the optimal solutions. This add-on Simulink package in the Matlab software can be used to create a block of diagrams which can be translated into a system of ordinary differential equations.

Illustrative examples are shown to prove the effectiveness of the proposed algorithm. Moreover, the proposed method have been well applied to biological and engineering problems such as linear and nonlinear singular systems, human immunodeficiency virus (HIV) models, microbial growth model and ethanol fermentation process.

ABSTRAK

Kepintaran berkumpulan adalah satu disiplin kepintaran buatan yang prihatin di dalam pembinaan dan pengoptimuman sistem-sistem multi-agen dengan aplikasi di dalam robotik. Secara asasnya kaedah bukan tradisional ini adalah berbeza dari kaedah tradisional. Di sebalik pengawalan yang rumit bagi mentadbir sifat global sesuatu sistem itu, prinsip kepintaran berkumpulan adalah berdasarkan gabungan banyak entiti-entiti yang tidak sofistikated (sebagai contoh seperti semut, anai-anai, lebah dan sebagainya) di mana entiti-entiti ini bekerjasama bagi memberikan sifat-sifat yang dikehendaki. Di dalam tesis ini, kami menggunakan kaedah modifikasi pengaturcaraan koloni semut (ACP) bagi menyelesaikan persamaan pembeza matriks Riccati (MRDE). Menyelesaikan MRDE, khususnya bukan linear MRDE telah menjadi fokus utama di dalam teori pengawalan optima. Di dapati dengan algoritma ACP ini, penyelesaian diperolehi adalah menghampiri atau sama dengan penyelesaian tepat. Kami juga membandingkan keputusan kami bersama penyelesaian berangka yang diperolehi dengan kaedah Runge-Kutta peringkat ke 4 (RK4) dan juga kaedah bukan tradisional pengaturcaraan genetik (GP). Selain itu, kami juga melaporkan penggunaan Simulink bagi menyelesaikan MRDE bagi mendapatkan penyelesaian optima. Pakej tambahan di dalam Matlab ini boleh digunakan untuk membina blok-blok diagram yang boleh diterjemahkan kepada sistem persamaan pembeza.

Contoh illustratif bagi membuktikan keberkesanan algoritma yang dicadangkan ada ditunjukkan di dalam tesis ini. Malah, kaedah yang dicadangkan telah diaplikasikan di dalam permasalahan biologi dan juga kejuruteraan seperti sistem singular linear dan bukan linear, human immunodeficiency virus model (HIV), model pertumbuhan mikrobial dan proses penapaian etanol.

ACKNOWLEDGEMENT

I would like to express my deep and sincere gratitude to both of my research supervisor, **Dr. N. Kumaresan** and **Prof. Kurunathan Ratnavelu**, from Institute of Mathematical Sciences, Faculty of Science, University of Malaya for their patience and energizing guidance. Without their encouragements and invaluable support, this research work and thesis wouldn't be possible. I also thank them for providing me with a good research environment and the learning for life. I gained much from their wide knowledge and vast research experience.

I also express my sincere thanks to Dr S. Jeeva Sathya Theesar, from Institute of Mathematical Sciences, Faculty of Science, University of Malaya for his invaluable suggestions during many occasions of the research work and for the completion of this thesis.

I wish to take this opportunity to thank the Head of Institute of Mathematical Sciences, University of Malaya and Centre for Foundation Studies in Science, University of Malaya for providing all the necessary facilities for doing this research work.

Also I want to express my sincere gratitude to my big family of research colleagues Prof. Koshy Philip, Mr Hilmi Jaafar, Dr. Ng Siow Yee, Ephrance Abu Ujum and Chin Jia Hou for their wonderful support during the whole research period and for successful completion of the thesis.

Last but not the least, I would like to express my deep gratitude to my parents: Mr. Mohamed Kamali Kormin, Mrs Entiah Saian and my wife, Siti Zawiah Hatibin who have been good moral support for me in every way of life and consecrate to them.

Place : Kuala Lumpur

Date : 05-03-2015 **MOHD ZAHURIN MOHAMED KAMALI**

LIST OF PUBLICATIONS

1. M. Z. M. Kamali, N. Kumaresan, and Kuru Ratnavelu, Solving Differential Equations with Ant Colony Programming, Applied Mathematical Modelling 39(2015), 3150-3163. (ISI publication)

2. M. Z. M. Kamali, N. Kumaresan, and Kuru Ratnavelu, Optimal Control For Stochastic Bilinear Quadratic Neuro Takagi-Sugeno Fuzzy Singular System Using Simulink, American Institute of Physics, accepted 2013. (ISI publication)

3. M. Z. M. Kamali, N. Kumaresan, Koshy Philip and Kuru Ratnavelu, Fuzzy Modelling of S-Type Microbial Growth Model for Ethanol Fermentation Process and the Optimal Control Using Simulink, Communications in Computer and Information Science, Volume 283, Part 1, 325-332, Springer-Verlag Berlin Heidelberg, 2012. (ISI publication)

LOCAL/INTERNATIONAL CONFERENCE PROCEEDINGS & PAPERS PRESENTED AT CONFERENCES

1. M. Z. M. Kamali, N. Kumaresan, and Kuru Ratnavelu, Optimal Control For Stochastic Bilinear Quadratic Neuro Takagi-Sugeno Fuzzy Singular System Using Simulink, Paper presented at The 2013 International Conference on Mathematics and Its Applications (ICMA2013), 18-21 Aug 2013, Kuala Lumpur, Malaysia.

2. M. Z. M. Kamali, N. Kumaresan, Koshy Philip and Kuru Ratnavelu, Fuzzy Modelling of S-Type Microbial Growth Model for Ethanol Fermentation Process and the Optimal Control Using Simulink, Paper presented at the International Conference on Mathematical Modelling and Scientific Computation 2012, 16-18 March 2012, Tamil Nadu, India.

3. M. Z. M. Kamali, N. Kumaresan, Koshy Philip and Kuru Ratnavelu, Fuzzy Modelling of P-Type Microbial Growth Model for Ethanol Fermentation Process and the Optimal Control Using Simulink, International Conference on Technology and Management 2012 , Lecture Notes in Information Technology,Vol 21, p81, 2012.

4. M. Z. M. Kamali, N. Kumaresan, and Kuru Ratnavelu, Solution of Matrix Riccati Differential Equation of Optimal Fuzzy Controller Design for Nonlinear Singular System with cross term using Simulink, Lecture Notes in Information Technology, Vol 10, p 304, 2012, International Conference on Affective Computing and Intelligent Interaction (ICACII2012), Information Engineering Research Institute, USA.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Name	Description
MRDE	Matrix Riccati Differential Equation
ACP	Ant Colony Programming
ODE	Ordinary Differential Equation
NLODE	Nonlinear Ordinary Differential Equation
FDE	Fuzzy Differential Equation
PDE	Partial Differential Equation
T-S	Takagi-Sugeno
LQ	Linear Quadratic
LQR	Linear Quadratic Regulator
TSP	Travelling Salesman Problem
ACO	Ant Colony Optimization
MMAS	Max-Min Ant System

LIST OF NOTATIONS AND SYMBOLS

Symbol	Meaning
$\mathbb{R}$	the set of real number
$\mathbb{R}^+$	the set of positive real numbers
$\mathbb{R}^{m\times n}$	the set of m x n real matrices
A (Capital Letters)	system matrix with constant entries, subscripts also used
A^T	transpose of a matrix A
$\|x\|$	norm of an element x in a normed space
$*$	symmetric block in one symmetric matrix
I	identity matrix with appropriate dimension
$\mathbb{E}(\cdot)$	the mathematical Expectation
$S = \{1, 2, \cdots, s\}$	finite state space
$x(t)$ and $y(t)$	state vectors
$\tau(t)$	time-varying delay

CHAPTER 1

INTRODUCTION

This work reports the theoretical investigation of solving the Matrix Riccati Differential Equation (MRDE) by using a modified Ant Colony Programming (ACP) algorithm and we also studied the MRDE using the Simulink. This modified ACP is unique and different from the other proposed ant colony methods. The characteristics of these modified ACP will be described further in this thesis.

1.0.1 General Introduction

Differential equations are widely used to derive and model physical phenomena. Information describing these phenomena is retrieved or extracted from the differential equations either analytically, numerically, or by using graphical tools and software. One of the most intensely studied nonlinear differential equations is the MRDE, which is very significant in optimal control problems, multivariable and large scale systems, scattering theory, estimation, detection, transportation, and radiative transfer (Jamshidi, 1980). A MRDE is a quadratic Ordinary Differential Equation (ODE) of the form

$$X' = A_{21} - XA_{11} + A_{22}X - XA_{12}X, \qquad X(0) = X_0,$$

with X is an $m \times n$ matrix-valued and A_{ij} are continuous, matrix-valued where both are functions of time t with matrix sizes to respect the size of X. The term "Riccati equation" refers to the matrix equations with an analogous quadratic term, which occurs in both continuous and discrete-time linear-quadratic-Gaussian control. Essentially, solving MRDE for state space representation of a dynamical system is a central issue in optimal control theory. The difficulty to get the solution from this equation can be viewed from two points: the nonlinear and the matrix form.

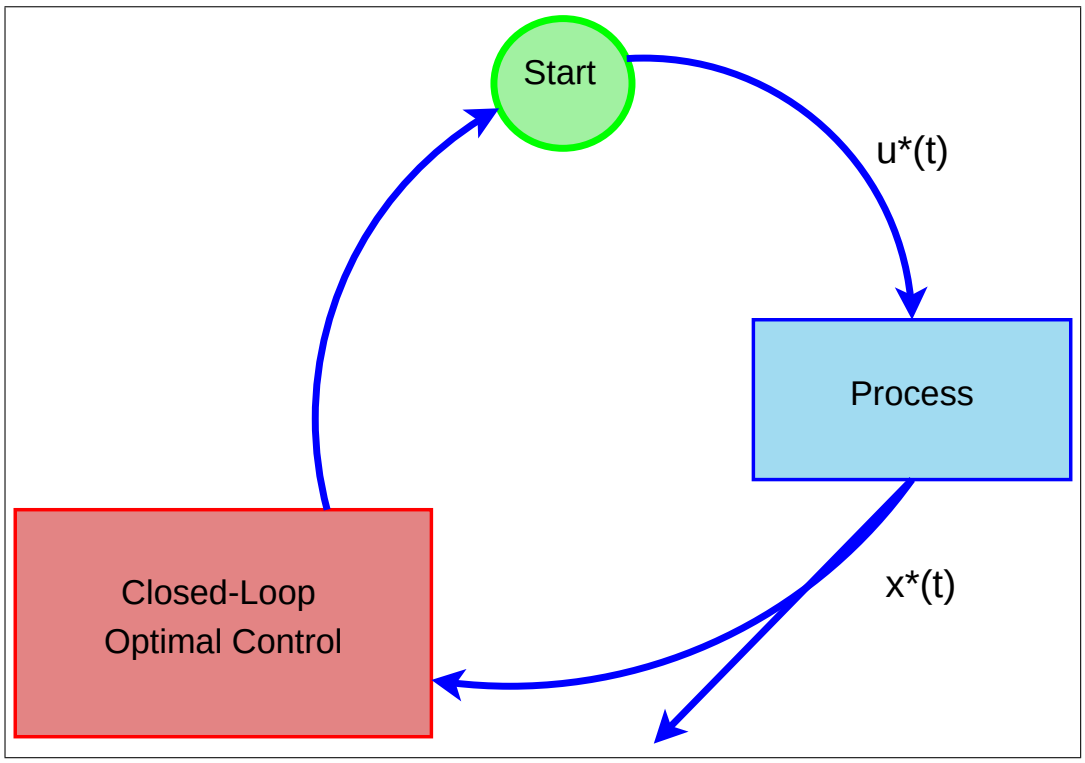

Figure 1.1: Closed-Loop Optimal Control System

1.1 Optimal control theory

The optimal control theory has been widely used in the field of science, engineering, finance and economics. Its objective is to design a control system that can determine the best control function for a dynamical system to minimize the performance index. An optimal control system consists of a set of differential equations which describe the paths of the control variables that minimize the cost function. There are two main approaches used in the optimal control problem: the dynamic programming and the variational approach. The dynamic programming approach is based on the principle of optimality that gives a closed-loop solution, which is depicted in Figure 1.1, resulting in a global search of the optimal controls. This approach has introduced a vital reduction in the computational time. Furthermore, a continuous approach of the principle of optimality may be presented, which results in the solution of the partial differential Hamilton-Jacobi-Bellman equation (Bellman, 1957).

The variational approach uses the Pontryagins minimum principle (Boltyanskii et al., 1956), which is a generalization of the Euler-Lagrange approach. However, the variational approach is an open-loop optimal control, which is depicted in Figure 1.2, and gives the optimal values for specific initial conditions.

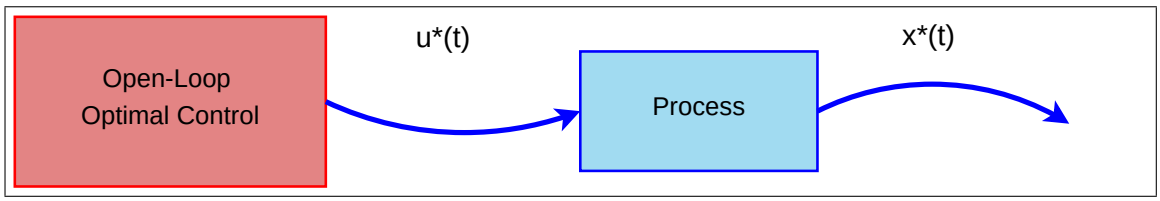

Figure 1.2: Open-Loop Optimal Control System

In the following, the optimal control problem is illustrated. Consider a system,

$$\dot{x} = f(t, x(t), u(t)), \qquad x(t_0) = x_0,$$

where $x \in \mathbb{R}^n$ is the state, $u(t)$ is the control function and form the set of admissible controls $u(t) \in U$, for all $t \in T$. The optimal control problem is to find a control function $u(t)$ that steers the system from an initial state $x(0) = x_0$ to a target state and minimizes the performance criterion,

$$J \quad = \phi(x(t_f)) + \int_0^{t_f} \quad L(t, x(t), u(t))dt, \tag{1.1}$$

where $\phi(x(t_f))$ is the terminal cost, $L(t, x(t), u(t))$ is the running cost and t_f refers as the terminal time which is either fixed or free. If the solution for the above problem can be found in the form

$$u(t) = u(t, x(t)),$$

then the control exists and it is called the optimal control law. In eq.(1.1), the terminal cost function is associated with error in the terminal state time t_f and L penalizes for transient state errors and control effort.

1.2 Pontryagin's minimum principle

The Pontryagin's minimum principle (PMP) was formulated by Pontryagin and his co-workers (Boltyanskii et al., 1956). This approach can be implemented only to deterministic problems and gives similar solutions as dynamic programming. The PMP approach also has some advantages and disadvantages. It can be used in cases where the dynamic programming approach fails due to lack of smoothness of the optimal performance criterion. It gives optimality conditions that in general are easier to verify than solving the

partial differential equation as in the dynamic programming approach. The optimal control can be derived by using the PMP (necessary conditions) or by solving the Hamilton Jacobi Bellman equation (sufficient condition) .

1.3 Linear optimal control

The linear optimal control is a special sort of optimal control where the plant is assumed linear while the controller that generates the optimal control, is constrained to be linear too. Linear controllers are obtained by working with quadratic performance indices. These approach are called as Linear-Quadratic (LQ) methods.

Advantages of linear optimal control:

1. Finding solutions for very difficult optimal control problems.

2. The linear optimal control approach can be applied into small fractions or signal operation of nonlinear systems.

3. The computational procedures required for linear optimal design may often be implemented to nonlinear optimal problems.

4. Linear optimal control provides a framework for the unified treatment of the control problems studied via classical methods. At the same time, it vastly extends the class of systems for which control designs may be achieved.

1.3.1 Linear quadratic regulator control

Linear quadratic regulator/control (LQR) is a basic method frequently used for designing controllers for linear (and often nonlinear) dynamical systems. It always refer to a problem where a dynamical system, which is described by a set of linear differential equations, is to be controlled by the quadratic cost function. The quadratic performance index to be minimized is,

$$J = \frac{1}{2}x^T(t_f)Sx(t_f) + \frac{1}{2}\int_{t_0}^{t_f}[x^T(t)Qx(t) + u^T(t)Ru(t)]dt,$$

and the running cost and the terminal cost functions can be expressed as quadratic equations:

$$L = \frac{1}{2}(x^T(t)Qx(t) + u^T(t)Ru(t)) = \frac{1}{2}\begin{bmatrix}x^T(t) & u^T(t)\end{bmatrix}\begin{bmatrix}Q & 0\\ 0 & R\end{bmatrix}\begin{bmatrix}x(t)\\ u(t)\end{bmatrix},$$

$$\phi(x(t_f)) = \tfrac{1}{2}x^T(t_f)Sx\phi(t_f).$$

The three weighting matrices Q, R and S are symmetric, with Q and S positive semidefinite and R positive definite. Then, the LQR problem is to minimize the quadratic continuous-time cost function subject to the linear first-order dynamic constraints:

$$\dot{x} = A(t)x(t) + B(t)u(t), \qquad x(t_0) = x_0.$$

In the finite-horizon case, the matrices are restricted in that Q and R are positive semidefinite and positive definite, respectively. In the infinite-horizon case, however, the matrices Q and R are not only positive-semidefinite and positive-definite, respectively, but are also constant. These additional restrictions on Q and R in the infinite-horizon case are enforced to ensure that the cost functional remains positive. Furthermore, in order to ensure that the cost function is bounded, the additional restriction is imposed such that the pair (A,B) is controllable. Note that the LQ or LQR cost functional can be thought of physically as attempting to minimize the control energy (measured as a quadratic form).

1.3.2 Infinite-horizon continuous-time linear quadratic regulator

The infinite horizon continuous time LQR is a specific LQR problem, where all the matrices (A, B, Q and R) are positive definite. To be more specific, the matrices Q and R, are positive-semidefinite and positive definite, respectively. Furthermore they are also constant. This is to ensure that the cost functional remains positive. Since this is the infinite time horizon case, the terminal cost is negligible. The initial time is set from zero and the terminal time t_f is taken as $t_f \to \infty$. Therefore the infinite horizon continuous time LQR problem is to minimize the cost function given as:

$$J = \tfrac{1}{2}\int_0^\infty [x^T(t)Qx(t) + u^T(t)Ru(t)]dt,$$

with subject to the linear time invariant first-order dynamic constraints:

$$\dot{x} = A(t)x(t) + B(t)u(t), \qquad x(t_0) = x_0.$$

In order to ensure that the cost function is bounded, the matrices A and B must be controllable. In the optimal control theory, the feedback control law is given as

$$u(t) = -K(t)x(t),$$

and the control gain $K(t)$ is obtained as

$$K(t) = R^{-1}B^T P(t),$$

where $P(t)$ is obtained by solving the continuous time algebraic Riccati equation (ARE)

$$A^T P + PA - PBR^{-1}B^T P + Q = 0.$$

1.3.3 Finite-horizon continuous-time linear quadratic regulator

For the finite horizon problem, the system is described on $t \in [t_0, t_1]$. The matrices Q and R are strictly positive definite with a quadratic cost function given as

$$J = \frac{1}{2}x^T(t_f)Sx(t_f) + \frac{1}{2}\int_{t_0}^{t_f}[x^T(t)Qx(t) + u^T(t)Ru(t)]dt. \tag{1.2}$$

The formula for the feedback control law that minimizes the cost function is similar to the infinite-horizon case except that P can be obtained by solving the continuous-time Riccati differential equation:

$$\dot{P}(t) = -A^T P - PA + PBR^{-1}B^T P - Q.$$

There are a few first order conditions that have to be followed for J_{min} which are given below. The state equation:

$$\dot{x}(t) = A(t)x(t) + B(t)u(t);$$

Co-state equation:

$$-\dot{\lambda} = Qx(t) + A^T(t)\lambda;$$

Stationary equation:

$$0 = Ru(t) + B^T(t)\lambda.$$

1.4 Fuzzy systems

Fuzzy Logic was introduced by Lofti A. Zadeh (Zadeh, 1965) and has been used for human knowledge based on decision making and dealing with reasoning that is approximate rather than fixed. Fuzzy logic may produce truth values ranging between 0 and 1 compared to the traditional binary numbers. It has emerged as a profitable tool for the controlling and steering of systems with uncertainties and complex industrial processes, as well as for household and entertainment electronics.

The complexity of the biological and engineering world which are inherently filled with uncertainties and nonlinear systems, has opened its door to the world of fuzzy logic. Studies have shown that fuzzy logic to be the most suitable tools to represent complicated system (Leite et al., 2011; Cao et al., 2011). It all starts with a fuzzy logic which is a form of many-valued logic and deals with reasoning that is approximate rather than fixed and exact. Normally, a fuzzy system consists of linguistic IF-THEN rules that have fuzzy antecedent and consequent parts. It is a static nonlinear mapping from the input to the output space. These inputs and outputs data are crisp real numbers and not fuzzy sets. Based on these IF-THEN rules, a fuzzy inference system has been developed and its block diagram is presented in Figure 1.3.

The fuzzification block helps to convert the crisp inputs to fuzzy sets and then the inference mechanism uses the fuzzy rules in the rule base to produce fuzzy conclusions or fuzzy aggregations. Finally, the defuzzification block changes these fuzzy conclusions into the crisp outputs. The fuzzy system with singleton fuzzifier, product inference engine, center average defuzzifier and Gaussian membership functions is called as standard fuzzy system (Oysal et al., 2006; Wang, 1998). The main advantages of using fuzzy systems for control and modeling applications are (i) to avoid the need for rigorous crisp mathematical modeling and to be useful for uncertain or approximate reasoning, especially for the system with a mathematical model that is difficult to derive and (ii) to allow fuzzy logic to make decision with the estimated values under incomplete or uncertain information.

Fuzzy controllers are rule-based nonlinear controllers. Therefore, their main application should be the control of nonlinear systems. However, since linear systems are good approximations of nonlinear systems around the operating points, it is of interest to study fuzzy control of linear systems. Additionally, fuzzy controllers due to their nonlinear nature may be more robust than linear controllers even if the plant is linear. Furthermore, fuzzy controllers designed for linear systems may be used as initial controllers for nonlinear adaptive fuzzy control systems where on-line tuning is employed to improve the controller performance. Therefore, systematic fuzzy controllers for linear systems is of theoretical and practical interest. Stability and optimality are the most important requirements in any control system. Stable fuzzy control of linear systems has been studied by a number of researchers (Wang, 1998; Wu et al., 2005; Jenkins & Passino, 1999). It is well-known that the fuzzy controllers are universal nonlinear controllers due to universal

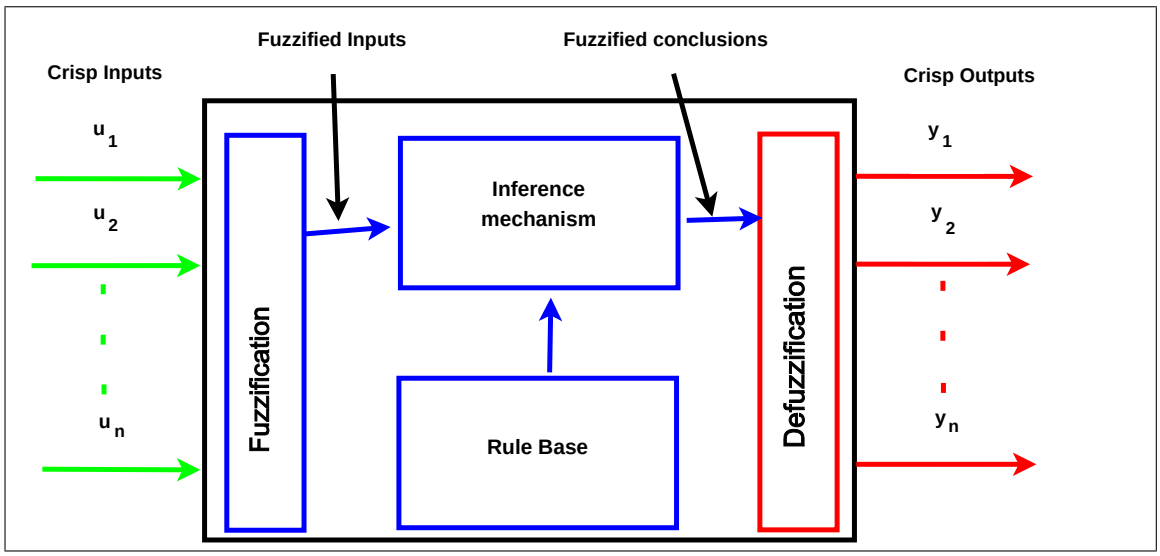

Figure 1.3: Fuzzy Inference System

nonlinear approximated models. All these studies are preliminary in nature and deeper studies can be undertaken. For optimality, problems in the field of optimal fuzzy control is still open for investigation.

1.4.1 Fuzzy control system

Conventional mathematics and control theory exclude vagueness and contradictory conditions. As a consequence, conventional control systems theory does not attempt to study any formulation, analysis and control of what has been called fuzzy systems, which may be vague, incomplete, linguistically described, or even inconsistent. Fuzzy set theory and fuzzy logic play a central role in the investigation of controlling such systems. The main contribution of fuzzy control theory, a new alternative and branch of control systems theory that uses fuzzy logic, is its ability to handle many practical problems that cannot be adequately managed by conventional control techniques. The aim is to extend the existing successful conventional control systems techniques and methods as much as possible and to develop new and special-purposed ones, for a much larger class of complex, complicated and ill-modeled systems-fuzzy systems. Fuzzy models can be static or dynamic. The widely used fuzzy models are rule based, in which the relationship between variables are represented by means of fuzzy IF-THEN rules. Rule-based fuzzy systems include Mamdani models (or linguistic fuzzy model), fuzzy relation models and T-S fuzzy model. T-S fuzzy systems are popular and well used tools in recent years.

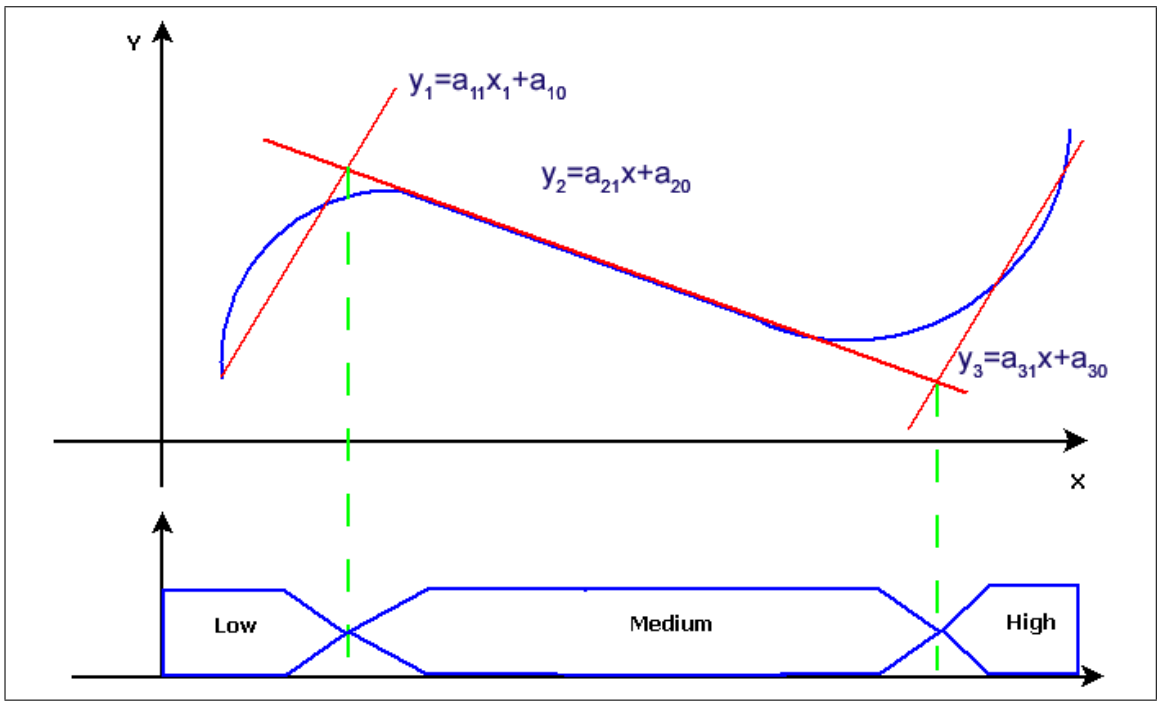

Figure 1.4: Takagi-Sugeno Fuzzy Systems

1.4.1.1 Takagi-Sugeno fuzzy system

The fuzzy inference system was suggested by Takagi and Sugeno (Takagi & Sugeno, 1985). A general T-S fuzzy model employs an affine fuzzy model with a constant term in the consequence. It is known that smooth nonlinear dynamic systems can be approximated by affine T-S fuzzy models (Cao et al., 1996; Ying, 1998). Most recent developments are based on T-S models with linear rule consequences. The main feature of T-S fuzzy models is to represent the nonlinear dynamics by simple (usually linear) models according to the so-called fuzzy rules and then to blend all the simple models into an overall single model through nonlinear fuzzy membership functions. Each simple model is called a local model or a sub-model. The output of the overall fuzzy model is calculated as a gradual activation of the local models by using proper defuzzification schemes. It has been proved that T-S fuzzy models can approximate any smooth nonlinear dynamic systems. The schematic representation for the T-S fuzzy systems is depicted in Figure 1.4. This enables a programmer to automate the inspection of the results and give more insight into the relationship between the changing parameters and the results.

1.5 Fuzzy optimal control problem

There are two types of fuzzy rules: Mamdani fuzzy rules and T-S fuzzy rules. The one that we used here in our present work is the T-S fuzzy rules. In T-S fuzzy rules, functions of input variables are used as the rule consequent as in the following form:
If $y(n)$ is M_1 AND $y(n-1)$ is M_2 AND $y(n-2)$ is M_3 AND $u(n)$ is M_4 AND $u(n-1)$ is M_5 THEN $y(n+1) = F(y(n), y(n-1), y(n-2), u(n), u(n-1))$, where $F(\cdot)$ is an arbitrary function. To construct a T-S fuzzy controller, we need a T-S fuzzy model that can be derived from a nonlinear system using sector nonlinearity approach (Kawamoto et al., 1993). Given the singular non-linear system as

$$E\dot{x}(t) = A(x)\,x(t) + Bu(t), \qquad x(0) = x_0, \tag{1.3}$$

where the matrix E is a singular matrix, $x(t) \in \mathbb{R}^n$ is a generalized state space vector and $u(t) \in \mathbb{R}^m$ is a control variable. $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ are coefficient matrices associated with $x(t)$ and $u(t)$ respectively, x_0 is given initial state vector and $m \leq n$. In order to derive the T-S fuzzy model from the nonlinear system, the first step is to determine the membership functions. For simplicity, the matrix $A(x)$ is taken as

$$A(x) = \begin{bmatrix} 0 & 1 \\ x_1(t) & x_2(t) \end{bmatrix}$$

and the fuzzy variables, x_1 and x_2 are also denoted as z_1 and z_2, respectively. By calculating the maximum and the minimum values of z_1 and z_2, the membership functions can be obtained, thus x_1 and x_2 can be represented for the membership functions M_1, M_2, N_1 and N_2 as follows:

$$z_1(t) = x_1(t) = M_1(z_1(t)) \cdot max(z_1(t)) + M_2(z_1(t)) \cdot min(z_1(t)),$$

$$z_2(t) = x_2(t) = N_1(z_2(t)) \cdot max(z_2(t)) + N_2(z_2(t)) \cdot min(z_2(t)).$$

Since M_1, M_2, N_1 and N_2 are fuzzy sets, their values can be computed by using the following relations

$$M_1(z_1(t)) + M_2(z_1(t)) = 1,$$

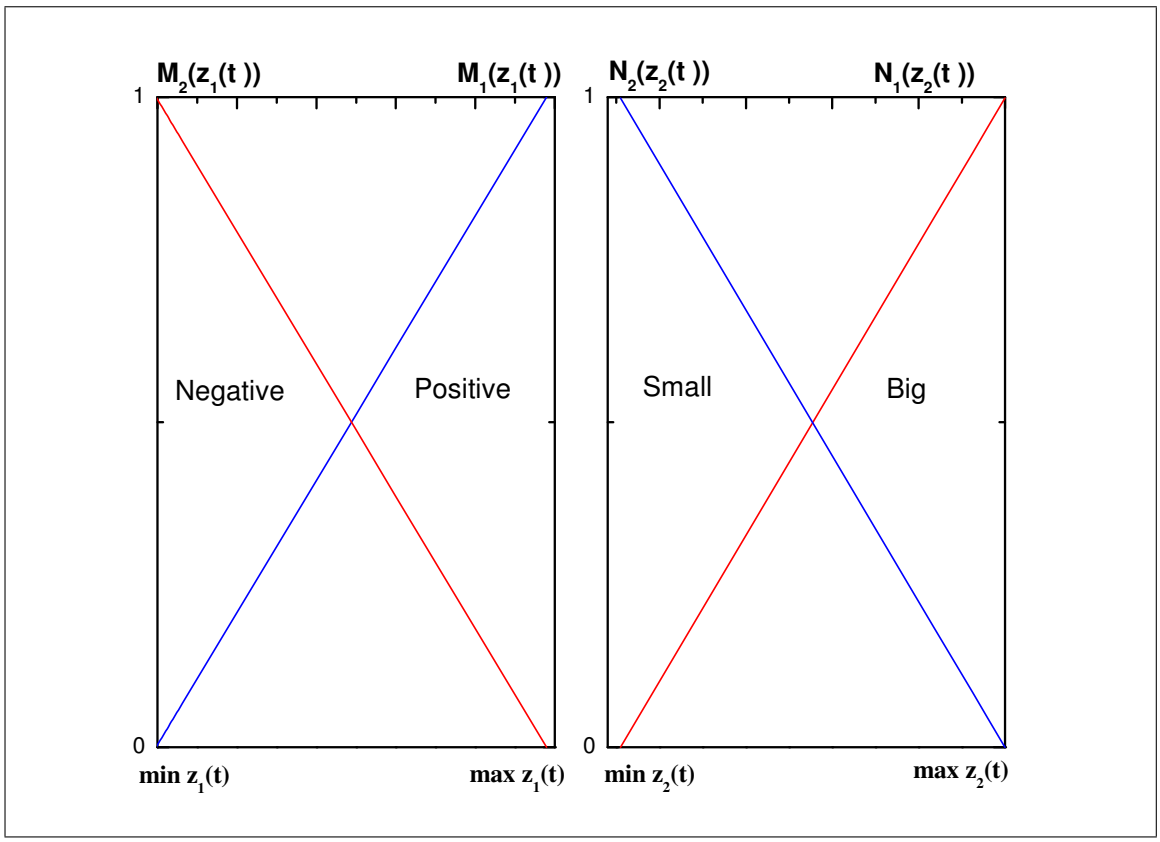

Figure 1.5: Membership function of $z_1\,(t)$ and $z_2\,(t)$

$$N_1\left(z_2\left(t\right)\right) + N_2\left(z_2\left(t\right)\right) = 1.$$

These membership functions are named as Small, Big, Positive and Negative, respectively and from this, the nonlinear systems can be linearized into the i^{th} rule of continuous T-S fuzzy model. Figure 1.5 depicts the membership functions for $z_1\,(t)$ and $z_2\,(t)$.
Consider the singular non-linear system (eq. (1.3)) that can be expressed in the form of T-S fuzzy system: Model Rule i: If $z_1\,(t)$ is M_{i1} and $z_2\,(t)$ is $M_{i2}...z_p\,(t)$ is M_{ip}, then

$$E_i\dot{x}(t) = A_i\left(x\right)x\left(t\right) + B_iu\left(t\right), \qquad x\left(0\right) = x_0,\ i = 1, 2, 3, 4.,$$

where M_{ij} is the fuzzy set rule of the fuzzy model, $x\,(t) \in \mathbb{R}^2$ is a generalized state space vector, $u\,(t) \in \mathbb{R}^1$ is a control variable and it takes value in some Euclidean space, $A \in \mathbb{R}^{2\times2}$, $B \in \mathbb{R}^{2\times1}$are known as coeffiecient matrices associated with $x\,(t)$ and $u\,(t)$, respectively, x_0 is a given initial state vector. Therefore, the nonlinear system is modeled by the following fuzzy rules where the subsystems are defined as

$$A_1 = \begin{bmatrix} 0 & 1 \\ max\left(z_1\left(t\right)\right) & max\left(z_2\left(t\right)\right) \end{bmatrix} \qquad A_2 = \begin{bmatrix} 0 & 1 \\ max\left(z_1\left(t\right)\right) & min\left(z_2\left(t\right)\right) \end{bmatrix}$$

$$A_3 = \begin{bmatrix} 0 & 1 \\ min\left(z_1\left(t\right)\right) & min\left(z_2\left(t\right)\right) \end{bmatrix} \qquad A_4 = \begin{bmatrix} 0 & 1 \\ min\left(z_1\left(t\right)\right) & max\left(z_2\left(t\right)\right) \end{bmatrix}.$$

Now from the defuzzification process the $E_i\dot{x}$ can be computed as

$$E_i\dot{x}(t) = \sum_{i=1}^{4} h_i\left(z\left(t\right)\right) A_i x\left(t\right) + B_i u\left(t\right),$$

where

$$h_i\left(z\left(t\right)\right) = \frac{\prod_{j=1}^{2} M_j^i(z_j(t))}{\sum_{i=1}^{4}\left(\prod_{j=1}^{2} M_j^i(z_j(t))\right)}$$

for all t. To minimize both state and control signals of the feedback control system, a quadratic performance index is minimized:

$$J = \tfrac{1}{2}[x^T(t)Qx(t) + u^T(t)Ru(t) + 2u^T\left(t\right)Hx\left(t\right)]dt$$

where the superscript T represents the transpose operator, $S \in \mathbb{R}^{2\times 2}$ and $Q \in \mathbb{R}^{2\times 2}$ are symmetric and positive definite (or semidefinite) weighting matrices for $x(t)$, $R \in \mathbb{R}^{1\times 1}$ is a symmetric and positive definite weighting matrix for $u(t)$. $H \in \mathbb{R}^{1\times 2}$ is a coefficient matrix. Based on the standard procedure, J can be minimized by minimizing the Hamiltonian equation

$$\begin{aligned} H\left(x\left(t\right), u\left(t\right), \lambda\left(t\right)\right) &= \frac{1}{2}x^T x(t) + \frac{1}{2}u^T(t)Ru(t) + u^T\left(t\right)Hx\left(t\right) \\ &\quad +\lambda\left(t\right)\left[A_i x\left(t\right) + B_i u\left(t\right)\right]. \end{aligned} \tag{1.4}$$

Using calculations of variations and Pontryagins maximum principle, a linear state feedback control law

$$u\left(t\right) = -R^{-1}(B_i^T\lambda\left(t\right) + Hx\left(t\right))$$

can be obtained from eq. (1.4) and

$$\lambda\left(t\right) = K_i\left(t\right)E_i x\left(t\right),$$

where $K_i\left(t\right) \in \mathbb{R}^{2\times 2}$ is a symmetric matrix and it is the solution of the relative MRDE for the singular system.

$$E_i^T\dot{K}_i\left(t\right)E_i + E_i^T K_i\left(t\right)A_i\left(t\right) + A_i^T K_i\left(t\right)E_i$$

$$+Q - (H^T + E_i^T K_i\,(t)\,B_i)R^{-1}(H + B_i^T K_i(t)E_i) \;\; = \;\; 0$$

1.6 Automatic programming

Automatic programming is a new search technique to find programs that solve a problem. This area of research is focusing on generating computer programs automatically.

Despite the success of heuristic optimisation and machine learning algorithms in solving real-world computational problems, their application to newly encountered problems, or even new instances of known problems, remains difficult; not only for practitioners or scientists and engineers in other areas, but also for experienced researchers in the field. The difficulties arise mainly from the significant range of algorithm design choices involved, and the lack of guidance as to how to proceed when choosing or combining them. This motivates the renewed and growing research interest in techniques for automating the design of algorithms in optimisation, machine learning and other areas of computer science, in order to remove or reduce the role of the human expert in the design process. Consider the area of evolutionary computation, for example. Initially, researchers concentrated on optimizing algorithm parameters automatically, which gives rise to adaptive and self-adaptive parameter control methods (Back, 1998). With time, the definition of parameters was broadened to include not only continuous variables, such as crossover and mutation rates, but also include categorical parameters, i.e., evolutionary algorithms components, such as the selection mechanism and crossover and mutation operators (Kramer, 2010). Later, evolutionary algorithms were first used in the meta-level, i.e., to generate a complete evolutionary algorithm, as showed in the works of Oltean (Oltean, 2005). In the area of machine learning, automated algorithm design appeared as a natural extension of the first works focusing on automated algorithm selection. The algorithm selection problem was formally defined by John Rice (Rice, 1976).

Koza was the first to propose Genetic Programming (GP) (Koza, 1992, 1994) which is the common example of automatic programming. Koza mentioned the five preparatory steps which should be fulfilled before searching for a program. The first step is selection of terminal symbols, then followed by the choice of functions, next the fitness function specification, later the selection of certain parameters for controlling the run and finally defining the termination criteria. Many of these principles used in GP are almost similar and can be adapted to ACP. Therefore Boryczkova and co-workers (Boryczka, 2002; Boryczka & Czech, 2002; Boryczka et al., 2003), applied ACP as an alternative method

for automatic programming with two different techniques: the expression and the program approach. In the expression approach, the quest for an approximating function is constructed in the form of an arithmetic expression. These expressions are in prefix notation. In the second technique, the expression is built from a sequence of assignment instructions which evaluates the function. Both techniques are based on a space graph which consists of the variables, functions and constant which is represented by the nodes, except that in the program approach each of these assignment instructions is located on a node. Although both approaches had showed some promising results but they cannot generate more general types of program. Other ant colony optimization (ACO) algorithms that have been extended and used for solving symbolic regression problems, are ant programming (AP) (Roux & Fonlupt, 2002) , generalized ant programming(GAP) (Keber & Schuster, 2002) and the dynamic ant programming (DAP) (Shirakawa et al., 2011). In the next section we will discuss briefly some of the well known and improved version of the ACO method.

1.7 Ant colony programming

The ant colony programming (ACP) is a stochastic approach which is implemented on a space graph. A space graph consists of the variables, functions and constants which are represented by the nodes. Functions are represented in terms of arithmetic operators, operands as well as Boolean functions. The set of functions defining a given problem is called a function set F and the collection of variables and constants to be used are known as the terminal set T.

The ACP can be implemented to generate a set of arithmetic expressions for solving ordinary differential equations. If the number of expressions satisfies the fitness function, then it will become the optimal solution. The four basic steps are listed below which are vital for the searching process based on Boryczka et al. (Boryczka & Wiezorek, 2003):

- Choice of terminals and functions
- Construction of graph
- Defining fitness function
- Defining terminal criteria

In the first colony, the digital ants will move randomly on the connected graph $G(V, E)$ where V indicates the Functions and Terminals whereas the set E represents the edges which connect the vertices. Normally, each ant is being put on a randomly chosen starting node and the pheromone value are distributed equally at all the edges. Each of these ants will move from the node r to the next node s in the graph at time t, by following the probability law (Boryczka, 2005),

$$\rho_{rs}(t) = \frac{\tau_{rs}(t) \cdot [\gamma_s]^{\beta}}{\sum_{i \in J_r^k} [\tau_{ri}(t)] \cdot [\gamma_i]^{\beta}}, \tag{1.5}$$

where parameter β controls the relative weight of the pheromone trail and visibility while J_r^k is the set of unvisited nodes. The γ_s is given as $\gamma_s = \left(\frac{1}{(2+\pi_s)}\right)^d$, where d is the current length of the arithmetic expression and π_s is the power of symbols which can be either a terminal symbol or a function and the power of symbols are given in Table 1.1.

The ant has completed its journey if it reaches the terminal node and based on the idea proposed in the MMAS, only a single ant which found the best solution, is used for the global update of pheromone trail in each generation. The pheromone trail update rule is given by:

$$\tau_{ij}(t+g) \leftarrow (1-\rho) .\tau_{ij}(t) + \triangle \tau_{ij}^{best}$$

$$\triangle \tau_{ij}^{best} = \begin{cases} \frac{1}{L_{best}}, & if\ ant\ best\ uses\ curve\ ij\ in\ its\ tour \\ 0, & otherwise, \end{cases}$$

where τ_{ij} shows the amount of pheromone trail on edge (i,j), g indicates the number of generation, L is the length of the optimal tour found on the edges (i, j), L_{best} is the best length of the optimal tour found on the edges (i, j) and $(1 - \rho)$, $\rho \in (0, 1]$ is the pheromone decay coefficient ($\rho > 0.5$ produces a good solution and this is actually referring to the concentration of pheromone on edge within the time t).

.

Table 1.1: Power of terminal symbols and functions.

Terminal symbol or function	Power
Constant, variable	-1
Functions,)	0
+, -, *, /, (	1

1.7.1 Terminals and Functions

Typically in a heuristic search technique, the space of graphs consists the nodes which represent functions, variables and constants. Functions are defined mathematically in terms of arithmetic operators, operands and boolean functions. The set of functions defining a given problem is called a function set and the collection of variables and constants to be used are known as the terminal set. The symbol $t_i \in T$ is a constant or any variable where $T = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, t\}$. Every function $f_i \in F$, can be evaluated as an arithmetic operator $\{+, -, *, /\}$, arithmetic function $\{\sin, \cos, \exp, \log\}$, special symbol $\{(\ ,\ )\}$ and an arbitrarily defined function appropriate to the problem under consideration. The terminal symbols and functions have the power to express the solution to a problem based on the composition of functions and terminals specified. The terminal symbol or functions are being presented by using the power (arity) and this is given in Table 1.1 (Boryczka, 2005)

1.7.2 Construction of graph

In ACP approach, the search space consists of a graph with l nodes. An example of such a graph is given in Figure 1.6. Each node represents either a function or a terminal. The edge which connects the nodes is weighted by pheromone. This graph is generated by a randomized process.

1.7.3 Fitness function

A fitness function is an objective type of function which is used for determining how close the suggested solutions with the objective goal. The motivation is to obtain the optimal solution from all the available solutions based on the given problem.

There are three selective conditions under this fitness function:

1. If $E_r = \left[\frac{dy}{dt} - g\left(t, y\right)\right]^2 = 0$,

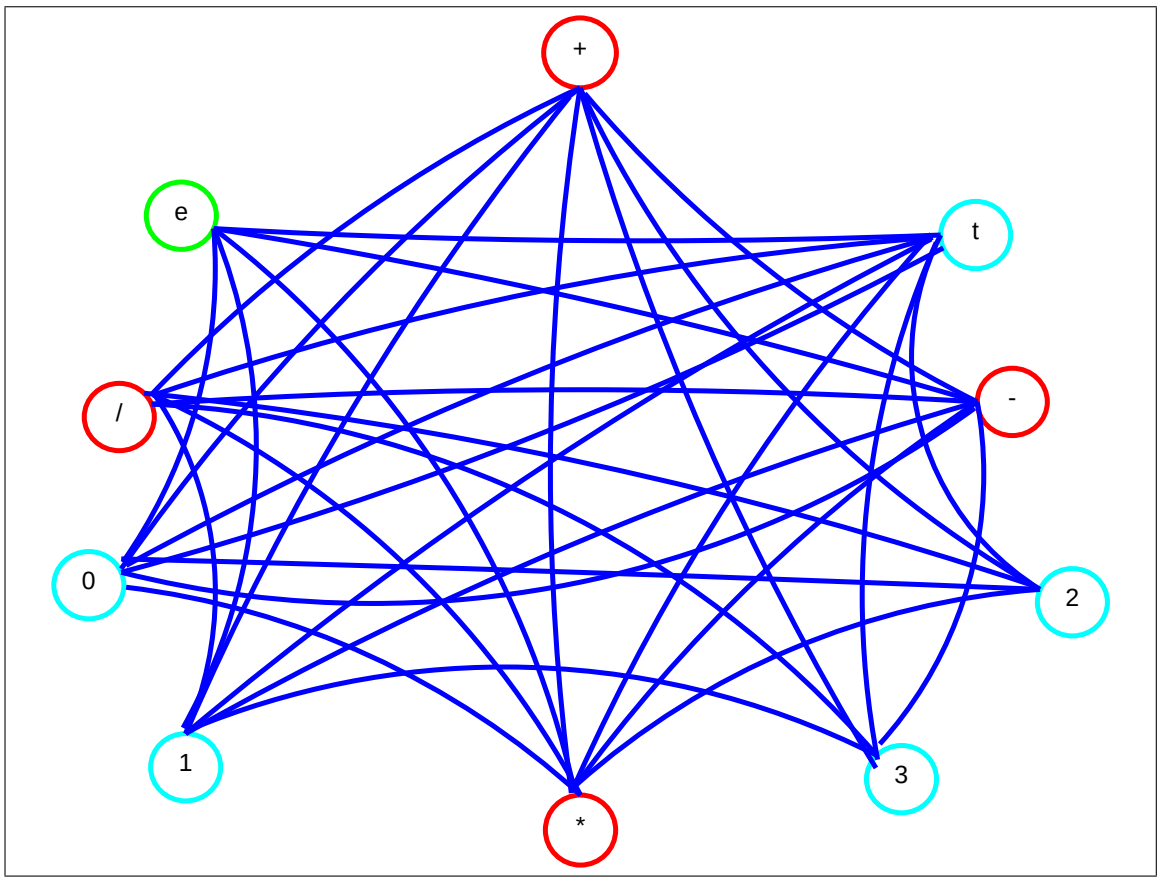

Figure 1.6: Graph with Functions and Terminals

then there will be no global pheromone update, if the exact solution is obtained for the ODE problem and the program will be terminated.

2. If $E_r = \left[\frac{dy}{dt} - g\left(t, y\right)\right]^2 \rightarrow 0$,

then a single ant manage to find a solution which is close to the exact answer. This information will be used to update the whole table of the pheromone values and the iteration for the next colony will fully utilise this piece of information in order to get to the optimal solution.

3. If $E_r = \left[\frac{dy}{dt} - g\left(t, y\right)\right]^2 \neq 0$,

then it is meaning that if not a single ant managed to find any solution which is nearer to the exact answer. Therefore, for the next colony, the dynamical pheromone value will stick to the previous one. Thus, there will be no global update.

The aim of the pheromone value global update rule is to increase the pheromone values on the solution path. This reduces the size of the search within the region in order to find high quality solution with reasonable computation time. On the updated graph, the consecutive cycles of the ant colony algorithm are carried out by sending the ants through the best tour of the previous generation. This procedure is repeated until the fitness function E_r,

becomes zero.

1.7.4 Applications of the Ant Colony Programming

1.7.4.1 Travelling Salesman Problem

The ACO Algorithm has been applied to a broad range of hard combinatorial problems. One of them is what we called the classic Traveling Salesman Problem (TSP). The TSP sets up a condition where a travelling salesman is needed to travel through a number of cities. The objective is the salesman must travel these cities (visiting each city exactly once) and make the total travelling distance as minimal as possible. This problem is one of the most widely studied problems in combinatorial optimization. The problem is easy to state, but hard to solve. The difficulty becomes apparent when one considers the number of possible tours - an astronomical figure even for a relatively small number of cities. For a symmetric problem with n cities there are $(n-1)!/2$ possible tours, which grows exponentially with n. If n is 20, there are more than 1018 tours. The Ant System was the first ACO algorithm proposed to tackle this problem (Dorigo, 1992; Dorigo et al., 1996). The TSP itself, has a large range of applications in real time problems although some of them seemingly have nothing to do with traveling routes. Its versatility is listed in the following examples such as:

- Transportation routing
- Route optimizization in robotic
- Chronological sequencing
- Maximum efficiency or minimum cost in process allocation

1.8 Simulink

The Simulink tool is a companion to MATLAB software. This add-on package can be used to create a block of diagrams which can be translated into a system of ODE. By using this, systems of ODE can be solved easily by using Runge-Kutta 4^{th} and 5^{th} order. One of the main advantages of Simulink is the ability to model a nonlinear dynamical system. Another advantage of Simulink is the ability to take on initial conditions.

Procedure for simulink solution

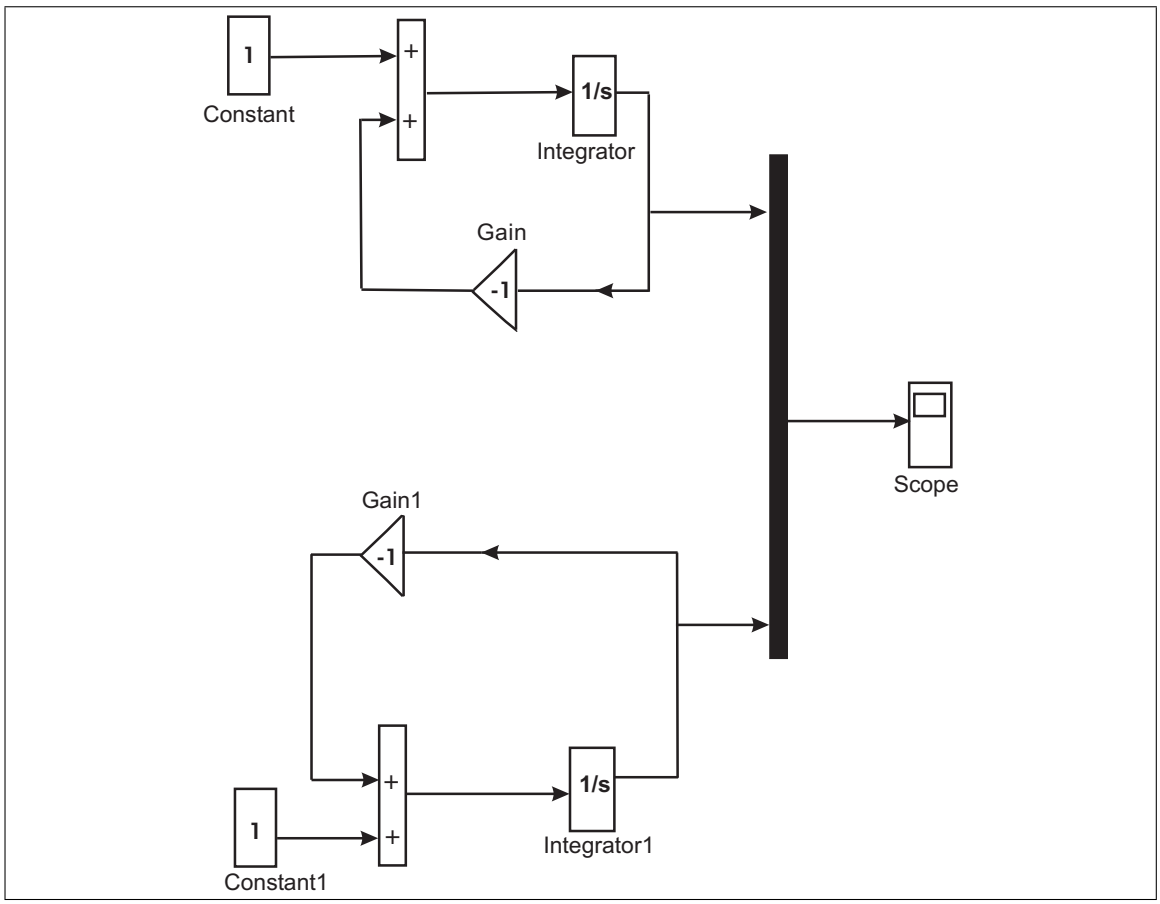

Figure 1.7: Simulink

Step 1: Choose or select the required graphical block diagrams from the Simulink Library.

Step 2: Connect the appropriate blocks.

Step 3: Set up the simulation parameters

Step 4: Run the Simulink

Example:

A Simulink model is constructed based on the following system of two differential equations as shown in Figure 1.7

$$x'(t) = -x(t) + 1, \qquad x(0) = -1$$

$$y'(t) = -y(t) + 1, \qquad y(0) = 1$$

1.9 Motivating Examples

The design of a zero propellant maneouvre for the international space station As a form of motivation, consider the design of a zero propellant maneouvre for the international space station by means of control moment gyroscopes. The example work was reported by Bhatt (Bhatt, 2007) and Bedrossian and co-workers (Bedrossian et al., 2009). The original 90 and 180 degree maneouvres were computed using a pseudospectral method. Implemented on the International Space Station on 5 November 2006 and 2 January 2007. Savings for NASA of around US$1.5m in propellant costs. The case described below corresponds with a 90 degree maneovure lasting 7200 seconds and using 3 Control Momentum Gyroscopes (CMG's). The problem is formulated as follows where the vital factor here is to find $q_c(t) = [q_{c,1}(t)\ q_{c,2}(t)\ q_{c,3}(t)\ q_{c,4}(t)]^T$, $t \in [t_0,\ t_f]$ and the scalar parameter γ in order to minimise,

$$J = 0.1\gamma + \int_{t_0}^{t_f} ||u\,(t)\,||^2 dt,$$

subject to the dynamical equations:

$$\dot{q}\,(t) = 0.5T\,(q)\,(\omega\,(t) - \omega_0\,(q))$$

$$\dot{\omega}\,(t) = J^{-1}\,(\tau\,(q) - \omega\,(t) \times (J\omega\,(t)) - u\,(t))$$

$$\dot{h}\,(t) = u\,(t) - \omega\,(t) \times h\,(t),$$

where J is a 3×3 inertia matrix, $q = [q_1;\ q_2;\ q_3;\ q_4]^T$ is the quaternion vector, ω is the spacecraft angular rate relative to an inertial reference frame and expressed in the body frame, h is the momentum, $t_0 = 0$s, $t_f = 7200s$. The path constraints:

$$||q\,(t)\,||_2^2 = ||q_c\,(t)\,||_2^2 = 1$$

$$||h\,(t)\,||_2^2 \leq \gamma$$

$$||\dot{h}\,(t)\,||_2^2 \leq \dot{h}_{max}^2.$$

The parameter bounds : $0 \leq \gamma \leq h_{max}^2$ whereas the boundary conditions is given as

$$q(t_0) = \bar{q}_0\ \omega(t_0) = \omega_0(\bar{q}_0)\ h(t0) = \bar{h}_0$$

$$q(t_f) = \bar{q}_f\ \omega(tf) = \omega_0(\bar{q}_f)\ h(t_f) = \bar{h}_f.$$

$T(q)$is given as :

$$T(q) = \begin{bmatrix} -q_2 & -q_3 & -q_4 \\ q_1 & -q_4 & q_3 \\ q_4 & q_1 & -q_2 \\ -q_3 & q_2 & q_1 \end{bmatrix},$$

where as $u(t)$ is the control force.

The atmospheric re-entry problem The precise problem under consideration is the following. We call atmospheric phase the period of time in which the altitude of the engine is between around 20 and 120 kilometers. It is indeed in this range that, in the absence of any motor thrust, the aerodynamic forces (friction with the atmosphere) can be employed to adequately control the space shuttle to as to steer it to a desired final point and meanwhile satisfying the state constraints in particular on the thermal flux. Thus, during this phase the shuttle can be considered as a glider, only submitted to the gravity force and the aerodynamic forces. The control is the bank angle, and the minimization criterion under consideration is the total thermal flux. The model of the control system is

$$\begin{aligned}
&\tfrac{dr}{dt} = \upsilon sin\gamma \\
&\tfrac{d\upsilon}{dt} = -gsin\gamma - \tfrac{1}{2}\rho\tfrac{SC_D}{m}\upsilon^2 + \Omega^2 r\, cosL\,(sin\gamma cosL - cos\gamma\, sinL\, cos\chi) \\
&\tfrac{d\gamma}{dt} = cos\gamma\left(\tfrac{-g}{\upsilon} + \tfrac{\upsilon}{r}\right) + \tfrac{1}{2}\rho\tfrac{SC_L}{m}\upsilon cos\mu + 2\Omega cos\, L\, sin\,\chi \\
&\quad +\Omega^2\tfrac{r}{\upsilon}cos\, L\,(cos\,\gamma\, cos\, L + sin\gamma\, sinL\, cos\chi) \\
&\tfrac{dL}{dt} = \tfrac{\upsilon}{r}cos\gamma\, cos\chi \\
&\tfrac{dl}{dt} = \tfrac{\upsilon}{r}\tfrac{cos\gamma\, sin\chi}{cosL} \\
&\tfrac{d\chi}{dt} = \tfrac{1}{2}\rho\tfrac{SC_L}{m}\tfrac{\upsilon}{cos\gamma}sin\mu + \tfrac{\upsilon}{r}cos\gamma\, tanL\, sin\chi + 2\Omega\,(sinL - tan\gamma\, cosL\, cos\chi) \\
&\quad +\Omega^2\tfrac{r}{\upsilon}\tfrac{sinL\, cosL\, sin\chi}{cos\gamma},
\end{aligned}$$

where r shows the distance between the center of gravity of the shuttle to the center of the Earth, υ refers to the modulus of its relative velocity, γ is the flight angle (or path inclination, that is, the angle of the velocity vector with respect to an horizontal plane), L is the latitude, l is the longitude, and χ is the azimuth (angle between the projection of the velocity vector onto the local horizontal plane measured with respect to the axis South-North of the planet). The gravitational force appears with a usual model $g(r) = \frac{\mu_0}{r^2}$, where μ_0 is the gravitational constant. The aerodynamic forces consist of the drag force, with the modulus $0.5\rho SC_D\upsilon^2$, which is opposite to the velocity vector, and of the lift

force, whose modulus is $0.5\rho SC_L\upsilon^2$, which is perpendicular to the velocity vector. Here, $\rho = \rho(r) = \rho_0 e^{-\beta r}$ is the air density, S is some positive coefficient (reference area) featuring the engine, and C_D and C_L are, respectively, the drag and the lift coefficients; they depend on the angle of attack and on the Mach number of the shuttle. The control is the bank angle μ; it acts on the orientation of the lift force and thus its action may be to make the shuttle turn left or right but also to act on the altitude. It is a scalar control that is assumed to take values in $[0, \pi]$. The mass m of the engine is a constant along this atmospheric phase since it is assumed that there is no thrust. Finally, Ω denotes the angular rotation speed of the planet. In the above model, the terms linear in Ω represent the Coriolis force, and the terms proportional to Ω^2 are due to the centripetal force. The optimal control problem under consideration is to steer the vehicle from initial conditions to final conditions, in free final time, and moreover the system is submitted to three state constraints:

- a constraint on the (instantaneous) thermal flux: $\varphi = C_q\sqrt{\rho}\upsilon^3 \leq \varphi_{max}$,
- a constraint on the normal acceleration: $\gamma_n = \gamma_{n0}\rho\upsilon^2 \leq \gamma_n^{max}$,
- a constraint on the dynamic pressure: $0.5\rho\upsilon^2 \leq P^{max}$,

where C_q, φ_{max}, γ_{n0} , γ_{n0}^{max} and P^{max} are positive constants. The minimization criterion is the total thermal flux along the flight

$$J(\mu) = \int_0^{t_f} C_q\sqrt{\rho}\upsilon^3 dt.$$

1.10 Organization of thesis

Besides this introduction chapter, the thesis is organized as follows :

Chapter 2 is concerned with the literature review of the previous work applied to solve the differential equations and the MRDE.

Chapter 3 is dedicated for the description of the modified Ant Colony Programming (ACP). The significance and the usage of this modified ACP approach for solving several differential equations problems were discussed in this chapter.

Chapter 4 shows the implementation of the modified ACP for solving the MRDE as well as some fuzzy modelling problems.

Chapter 5 shows the implementation of Simulink to solve the MRDE and some biological problems.

The final chapter concludes the thesis and provides future directions of the proposed research work.

CHAPTER 2

LITERATURE REVIEW

2.1 Solving differential equations

With vast advancements in computer technology, a lot of different methods have been developed for solving ordinary differential equations (ODEs) and partial differential equations (PDEs). Among the methods that have been applied and used for solving these types of equations are the Runge-Kutta, radial basis function (Fasshauer, 1999), genetic programming (Burgess, 1999) and feedforward neural network (Lagaris et al., 1998). Some of these methods compute solution in an array form which contains the value of the solution whereas others apply the basis functions to represent the exact solution and convert the original problem into a system of algebraic equations. In the feedforward neural network, the ODEs and PDEs depend on the function approximation capabilities. In order to obtain the solution, the feedforward neural network is trained to minimize the suitable error function, by employing the optimization techniques. Another alternative, is the genetic programming (GP) method. It is an optimization process which was based on the large number of possible solutions through genetic operations such as mutation, crossover and replication. Tsoulos and Lagaris (Tsoulos & Lagaris, 2006) , used grammatical evolution which is based on genetic programming to solve ordinary and partial differential equations. They reported that their method can create trial solutions and seeks to minimize an associated error. In most of the problems, the exact solution can be obtained but there are cases when the solution cannot be expressed in an analytical form. When this happens, the need to get an approximate answer with controlled level of accuracy will be produced.

2.2 Solving the matrix Riccati differential equation

The importance of solving the matrix Riccati differential equations is vital in the optimal control theory. It was reported that the stochastic linear quadratic regulator problems can be well posed if solutions can be produced for the Riccati equation, thus an optimal feedback control can be obtained (Chen et al., 1998). However, due to the presence of complicated nonlinear terms in the Riccati equations , this made the problems more difficult to be solved.

Since then, we have seen quite a number of non-traditional methods which try to solve this MRDE with less calculus effort and computational time. These non-traditional methods were inspired from the collective behaviour of biological systems. Their approaches which work based on to a certain degree of randoms were implemented for solving the MRDEs. The neural networks have been implemented to control nonlinear systems (Chen & Liu, 1994; Rovithakis & Christodoulou, 1994; Sadegh, 1993; Polycarpou, 1996). The usage of this neural networks approach is vital since they can effectively extend adaptive control techniques to nonlinearly parameterized systems. Miller et. al (Miller et al., 1990) were the first to show the use of neural networks for finding optimal control laws by using the Hamiltonian-Jacobian-Bellman equation. Later, Parisini and Zoppoli (Parisini & Zoppoli, 1998) utilized the neural networks for deriving the optimal control laws for discrete-time stochastic nonlinear system. This was followed later by Balasubramaniam and co-workers (2006, 2007a,b), where they were the first to implement the neural networks for solving the MRDE for linear singular system. Samath and Selvaraju (Abdul Samath & Selvaraju, 2010) incorporate the neural networks for solving the MRDE in nonlinear singular systems. They reported that the neuro computing approach yields solution of MRDE significantly faster than the Runge-Kutta method. The lengthy computational time for finding optimal control is avoided by using neuro optimal controller.

Another well known non-traditional approach that has been used to solve the MRDE is the genetic programming (GP) method. It was proved that GP obtained faster convergence when they applied the GP for finding the numerical solution of MRDE for singular systems (Vincent Antony Kumar & Balasubramaniam, 2007). Furthermore, they also extended their work to the nonlinear singular systems. Others such as Kumaresan and Ratnavelu (2014) reported their work on optimal control stochastic linear quadratic singular neuro Takagi-Sugeno (T-S) fuzzy system. They also applied the GP to compute the

solution for the MRDE. The theoretical group from India led by Balasubramaniam and co-workers (Balasubramaniam & Kumar, 2009; Kumaresan & Balasubramaniam, 2010), showed how they solved the MRDE using the neural networks and genetic programming.

Last but not least is the ant colony programming (ACP). The ACP has been used as an engineering approach to the design and implement automatic software systems instead of complex optimization problems. In 2009, Ast et al. (Ast et al., 2009) implemented a novel Ant Colony Optimization (ACO) algorithm which they called as the Fuzzy ACO for the automated design of optimal control policies for continuous-state dynamic systems. This algorithm integrates the multi-agent optimization heuristic of ACO with a fuzzy partitioning of the state space of the system. Later, Kumaresan and co-workers (Kumaresan, 2010; Kumaresan & Balasubramaniam, 2010; Kumaresan, 2012, 2011) reported quite a number of work, when they applied the ACP to solve the optimal control for stochastic linear quadratic singular fuzzy system. In order to get the optimal control, the solution MRDE is computed by solving the differential algebraic equation using a novel and nontraditional ACP approach.

2.3 Solving the microbial growth

There are vast number of mathematical models that have been developed to predict microbial growth in food and culture media. These models are based on some basic mathematical models such as the logistic model and Gompertz model. In 1987, Gibson et al. were the first to modify the logistic model to fit the bacterial growth. Similarly, he proposed a modified Gompertz model for bacterial growth. The modified logistic and Gompertz models fit bacterial growth, but the latter model gave better results (Gibson et al., 1987, 1988). Although these modified models are practical, but they are mechanically unacceptable. Some attempts have been made to develop more mechanistic growth models. In this respect, Baranyi and Roberts (1994, 1995) developed the Baranyi model. The Baranyi model (1994) is valid under dynamic environmental conditions and has become one of the most commonly preferred growth models due to the fact that it has a good fitting capability. It can also be applied for dynamic environmental conditions and most of the model parameters are biologically interpretable. (Lopez et al., 2004; Pin et al., 2002; Van Impe et al., 2005). Although clearly interpretable, this model, inherited from the logistic type, fails in describing more complex yet more realistic situations (for

example the co-cultural growth and the growth in structured media). Therefore, Van Impe et al. extended the Baranyi models by proposing two types of models: the S and P-type (Van Impe et al., 2006). These models explicitly incorporates nutrient exhaustion and/or metabolic waste product effects. Furthermore, these models can be extended in a natural way towards microbial interactions in co-cultures and microbial growth in structured foods. All the models that we described above is called the primary models. The primary models describe the change in bacterial count over time, under given environmental and cultural conditions. Such models can generate information about microorganisms, such as generation time, lag-time, exponential growth rate, and maximum population density, also called kinetic parameters.

Secondary models refer to the response of one or more kinetic parameters estimated from the primary model (e.g., lag time) to change in multi-environmental conditions (pH, temperature, additives, etc.). Examples of this type of model include the model of Davey (Davey, 1991), Artificial neural network(ANN) models (Garcia-Gimeno et al., 2002) etc. Artificial neural networks can be included under what are known as Artificial Intelligence models.

Tertiary models represent the applications of one or more secondary models to generate systems for providing predictions to nonmodelers, user-friendly software, and expert systems (Adair & Briggs, 1993; Jones, 1993) that can be included under Artificial Intelligence. There are several microbial modeling software packages currently available, including the Food Spoilage Predictor (Neumeyer et al., 1997), Decision Support System (Zwietering et al., 1992; Wijtzes et al., 1998), Seafood Spoilage Predictor (Dalgaard et al., 2000), Chefcad software (Nicolai & Baerdemaeker, 1996), and Quality Risk Assessment (Brown et al., 1998).

2.4 Solving the human immunodeficiency virus immunology model

An incurable disease caused by Human Immunodeficiency Virus (HIV), AIDS attacks and destroys the human immune system. This leaves the patient defenseless against illnesses that can lead to death. Scientists are trying very hard, in the search for an anti-HIV vaccine. Other efforts such as chemotherapies are aimed at killing or halting the pathogen, but treatment which can boost the immune system can serve to help the body fight infection on its own. New treatments focus more on reducing the viral population and improving

the immune response. This enlighten some new hope to the treatment of HIV infection, and some researchers explored these strategies for such treatments using the optimal control techniques (Joshi, 2002; Zhou et al., 2014; Ghanbari & Farahi, 2014; Roshanfeki et al., 2014). Others try to incorporate the fuzzy approach in order to control a nonlinear dynamic model of the HIV immunology (Miguel et al., 2006; Zarei et al., 2012). Miguel et al implemented the genetic fuzzy system approach for controlling a nonlinear dynamic model of the HIV immunology. They set up to find Mamdani fuzzy controllers that are capable of boosting the immune response while reducing the impact on the body because of potentially toxic medications usage. Zarei et al. (2012) proposed a fuzzy mathematical model of HIV dynamics where these three-dimensional FDEs are capable to describe the ambiguous immune cells level and HIV viral load which are due to existing patients with various strength of their immune system. They also utilized the fuzzy model and studied a fuzzy optimal control problem minimizing both the viral load and drug cost.

CHAPTER 3

SOLUTION OF DIFFERENTIAL EQUATIONS USING MODIFIED ANT COLONY PROGRAMMING

In this chapter, we will describe the development of the ant colony programming (ACP) that have been incorporated with some new features that is not available in other ACP methods. We will also discuss the algorithm as well as the computational details of the present modified ACP. Furthermore, we show some examples where we implement the proposed algorithm in order to obtain solutions for the differential equations problems.

3.1 Unique criteria of the modified ACP.

First, we included two new properties under the terminal symbols, which is the open bracket '(' and the close bracket ')'. The inclusion of this two new properties are vital in order to differentiate with the use of multiplication symbol '*'. In the work reported by Kumaresan and co-workers (Kumaresan & Balasubramaniam, 2010), the '*' symbol is used not only for representing multiply operations but also for showing the implementations of bracket in the expressions. For example the tour of an ant which generated an expression such as $e * t + 1 * + 5$ actually represents $e^{(t+1)} + 5$. If the number of nodes are increased, the expression will expand and this will make our expression look very complicated. Therefore the inclusion of these new properties is meant to simplify the expression and to make it more readable and easier for the programming language to evaluate the expression.

Second, the infix formation. The expression is generated by implementing the infix formation and this is totally different from the algorithm used by the previous works. Suppose that the algorithm starts with a fixed number of nodes, such as 6 nodes and we fixed the first node starts from exp or e. The ants will move freely to the next node

Table 3.1: Expressions generated for 6 nodes.

ant	node 1	node 2	node 3	node 4	node 5	node 6
ant 1	e	$*$	t	(	)	2
ant 2	e	(	t	$+$	2	)
ant 3	e	2	4	)	$+$	t
ant 4	e	)	(	$*$	5	5
ant 5	e	(	2	$-$	$+$	t
ant 6	e	(	4	)	$+$	t

according to the probability law. Therefore a lot of possible expressions can be generated from the ants as they are jumping from one node to another.

The implementation of infix form will help to evaluate the generated expression faster by following the terms and conditions of the syntax used in the programming language. For example, in Table 3.1 by initializing the value $t = 0$, expressions generated by ant 2 and ant 6 can be evaluated. Other expressions which do not follow the rules and syntax set by the programming language will be ignored since these expressions cannot be evaluated. Therefore the program will skip these unwanted expressions and jump to the next expression. Only expressions that can be evaluated will be channelled for fitness function trial.

Normally in the ACP method, the shortest distance is found out by using the quantity of the pheromeone whereas in this modified ACP, the best tour is found out using the pheromone quantity. The present ACP algorithm is unique as it does not depend on the distance but it utilizes more on the probability function which is connected to the quantity of the pheromone level in the ACP. The data for the quantity of the pheromone values in each edge is depicted in Table 3.2. The data was collected after the ants have completed their travels through 6 nodes. The symbols or the Terminals and Functions are represented by the nodes in the space graph. Initially, the pheromone values are equally distributed through out all the edges ($\tau_{ij} = 0.2$). As the ants move through out all the edges, there will be some path which will be favorable as these paths satisfy the initial conditions. The best path can be obtained only if the path satisfies the fitness function. From Table 3.2, the favorable edges are listed down in Table 3.3 and possible solutions are shown in Table 3.4. Thus, in this work, we will implement the modified ACP to generate expression for solving ordinary (ODEs), nonlinear and partial differential equations (PDEs). In the next subsection, we present the modified ACP algorithm that we have used in this thesis.

Table 3.2: Pheromone values depicted in each edge.

Symbols		0	1	2	3..	5	6	7..	t..	+..	(	)
	Nodes	0	1	2	3..	5	6	7..	10	12..	15...	17
0	0	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20
1	1	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20
2	2	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20
3	3	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.30
4	4	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20
5	5	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.30	0.20	0.20
6	6	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.30
7	7	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.30	0.20	0.20
8	8	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20
9	9	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20
t	10	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.35	0.20	0.35
$+$	11	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20
$*$	12	0.20	0.20	0.20	0.30	0.20	0.30	0.20	0.35	0.20	0.20	0.20
$-$	13	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20
\	14	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20
(	15	0.20	0.20	0.20	0.20	0.30	0.20	0.30	0.35	0.20	0.20	0.20
e	16	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.39	0.20
)	17	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20

3.1.1 Modified ACP algorithm

Step 1: *Start by constructing a graph with ℓ nodes.*

Step 2: *Distribute equal quantity or weight of pheromone in each edge of the graph as a starting point.*

Step 3: *Pass k ants through the graph from the starting node and move to the next node according to the probability law, eq.(1.5).*

Step 4: *Construct parse trees from the tours of k ants.*

Step 5: *Extract the expression which is generated according to the infix formation.*

Step 6: *Evaluate these expressions and skip unwanted ones.*

Step 7: *Only expressions that can be evaluated will be channelled for the fitness function, E_r*

Step 8: *If $E_r \rightarrow 0$ and they satisfy the terminal conditions, then stop. Otherwise, apply global update.*

Step 9: *Identify the best tour of the previous generation.*

Table 3.3: List of the most favorable edges

Edges	Pheromone values
(16,15)	0.39
(10,12)	0.35
(10,17)	0.35
(12,10)	0.35
(15,10)	0.35
(3,17)	0.30
(5,12)	0.30
(6,17)	0.30
(7,12)	0.30
(12,3)	0.30
(12,6)	0.30
(15,5)	0.30
(15,7)	0.30
Others	0.20

Step 10: *Pass the same k ants through the best tour and go to step 3.*

3.1.2 Modified ACP pseudocode

The flow chart for the ACP is given in Figure 3.1 .

Step 1: Initialize Construct a graph with ℓ nodes.
Setting up the equal weight of pheromone in each $edge(i, j)$ of the graph.
Input value for No. of Generations of ants.
Input value for No. of ants.

Step 2: For $m = 1$ to No. of Generations.

For $k = 1$ to No. of ants

For $\ell_1 = 1$ to ℓ nodes.

Pass k ants through the graph from k starting points and they move to the next node according to the probability law.

$p_{rs}(t) = \frac{\tau_{rs}(t).[\gamma_s]^\beta}{\sum_{i \in J_r^k}[\tau_{ri}(t)].[\gamma_i]^\beta}$,

This information will be stored in tour[m, k, ℓ_1].

Step 3: For $m = 1$ to No. of Generations.

For $k = 1$ to No. of ants.

For $\ell_1 = 1$ to ℓ nodes.

Analyze the data stored in tour[m, k, ℓ_1] by

-constructing parse trees from the tours of k ants.

Table 3.4: Possible solutions.

Nodes	16	15	5	12	10	17
Symbols	e	(	5	$*$	t	)
Nodes	16	15	7	12	10	17
Symbols	e	(	7	$*$	t	)
Nodes	16	15	10	12	6	17
Symbols	e	(	t	$*$	6	)
Nodes	16	15	10	12	3	17
Symbols	e	(	t	$*$	3	)

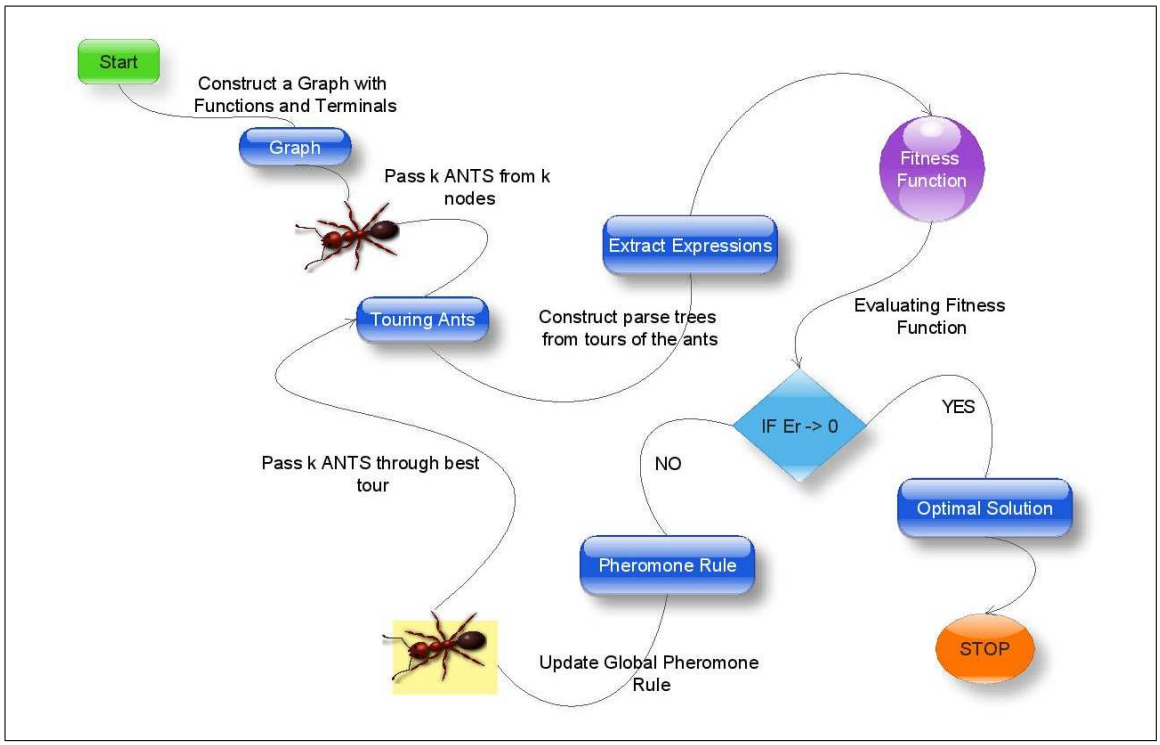

Figure 3.1: Flow Chart for ACP

-Extract the expression which is generated according to the infix formation.

-Evaluate these expressions and skip unwanted ones.

Step 4: Evaluating the fitness function, E_r.

If ($E_r \to 0$ and satisfy the terminal conditions)

the process will be stopped and the result is obtained.

else

update the global pheromone rule and

$\tau_{ij}(t+g) = (1-\rho).\tau_{ij}(t) + \rho.\frac{1}{L}$,

identify the best tour of the previous generation.

Pass the same k ants through the best tour and the process will go to Step 2.

3.2 Differential equations

Differential equations are widely used to model physical phenomena in the real world. In this paper, a nontraditional ant colony programming approach is implemented. The motivation in this present work is to describe the consistency and the applicability of the nontraditional ACP method in solving various ODEs and PDEs problems. Comparatively, similar and exact solution is achieved by using the ACP approach.

3.2.1 Linear Ordinary Differential Equations

In this work, we present the results from the first and second order linear ODEs. The ACP is applied in each equation and in every experiment the analytical solution is found. The solution is obtained using ACP and shown in the parse trees structure.

ODE1

$$y^{'} = \frac{2t - y}{t}$$

with $y(0.1) = 20.1$ and $t \in [0.1, 1.0]$. The tour of the ant is $t + (2/t)$. The parse tree of the tour is given in Figure 3.2(a). Expression extracted from the parse tree is given as $y(t) = t + \frac{2}{t}$.

ODE2

$$y^{'} = \frac{1 - ycos\,(t)}{sin\,(t)}$$

with $y(0.1) = \frac{2.1}{sin(0.1)}$ and $t \in [0.1, 1]$. The tour of the ant is $t + 2/sin(t)$. The parse tree of the tour is given in Figure 3.2(b). Expression extracted from the parse tree is given as $y(t) = \frac{t+2}{sin(t)}$.

ODE3

$$y^{''} = -100y$$

with $y(0) = 0$, $y\prime(0) = 10$ and $t \in [0, 1]$. The tour of the ant is $sin(2 * 5 * t)$. The parse tree of the tour is given in Figure 3.2(c). Expression extracted from the parse tree is given as $y(t) = sin(10t)$.

ODE4

$$y^{''} = 6y^{'} - 9y$$

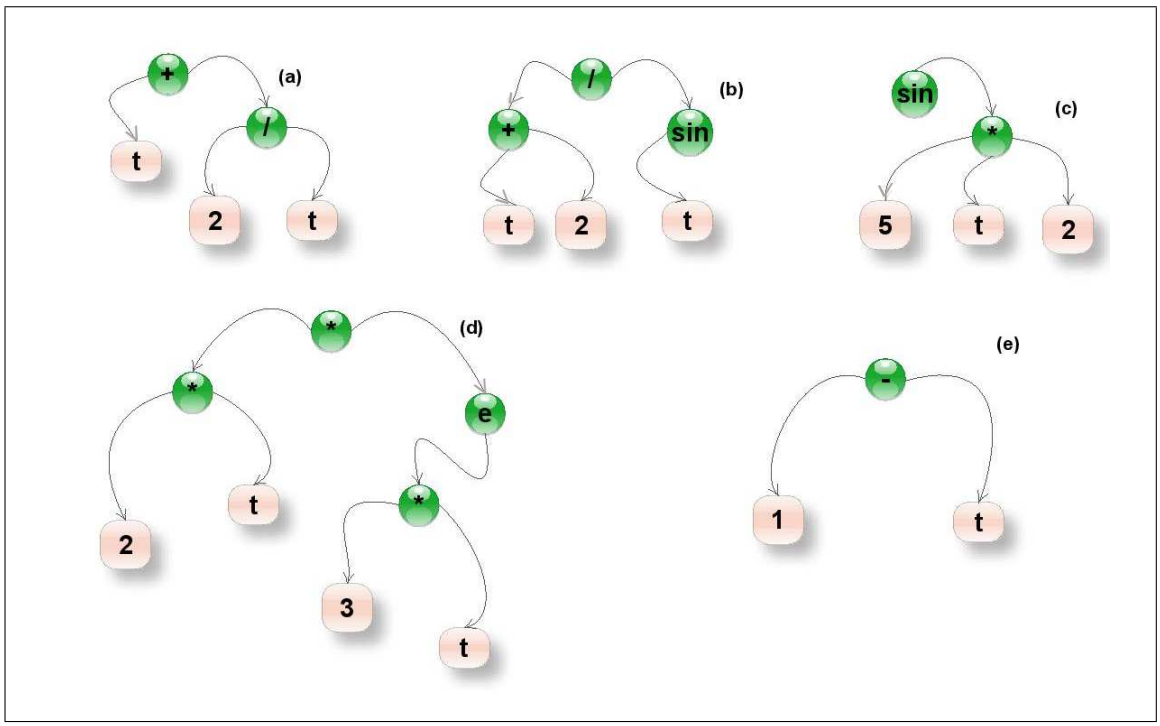

Figure 3.2: Parse tree solution for (a) ODE1, (b) ODE2, (c) ODE3, (d) ODE4 and (e) ODE5.

with $y(0) = 0$, $y\prime(0) = 2$ and $t \in [0, 1]$. The tour of the ant is $2 * t * exp(3 * t)$. The parse tree of the tour is given in Figure 3.2(d). Expression extracted from the parse tree is given as y(t) =$2te^{3t}$.

ODE5

$$ty'' + (1 - t)\, y' + y = 0$$

with $y(0) = 1$ and $y(1) = 0$ and $t \in [0, 1]$. The tour of the ant is $(1 - t)$. The parse tree of the tour is given in Figure 3.2(e). Expression extracted from the parse tree is given as $y(t) = 1 - t$. In Table 3.5, the average number of colonies and ants together with the time are shown in order to obtain the exact solutions for the ODEs. In Figure 3.3, the average number of generations obtained for finding the solutions in each of the problems discussed above are shown. Comparison between the present ACP method and the genetic programming(GP) (Tsoulos & Lagaris, 2006) are illustrated here. The blue scale on the left hand side of the y-axis referred to the ACP method whereas the red scale on the right hand side is for the GP method. The figure showed that the ACP method compute the solution efficiently and faster compare to the GP method.

Table 3.5: Results for linear ODEs using ACP

ODE	Average No.Generations	Average No. Ants	Average Time(secs)
ODE1	58	52	2.53
ODE2	70	63	7.77
ODE3	60	63	8.15
ODE4	75	65	10.3
ODE5	50	55	2.33

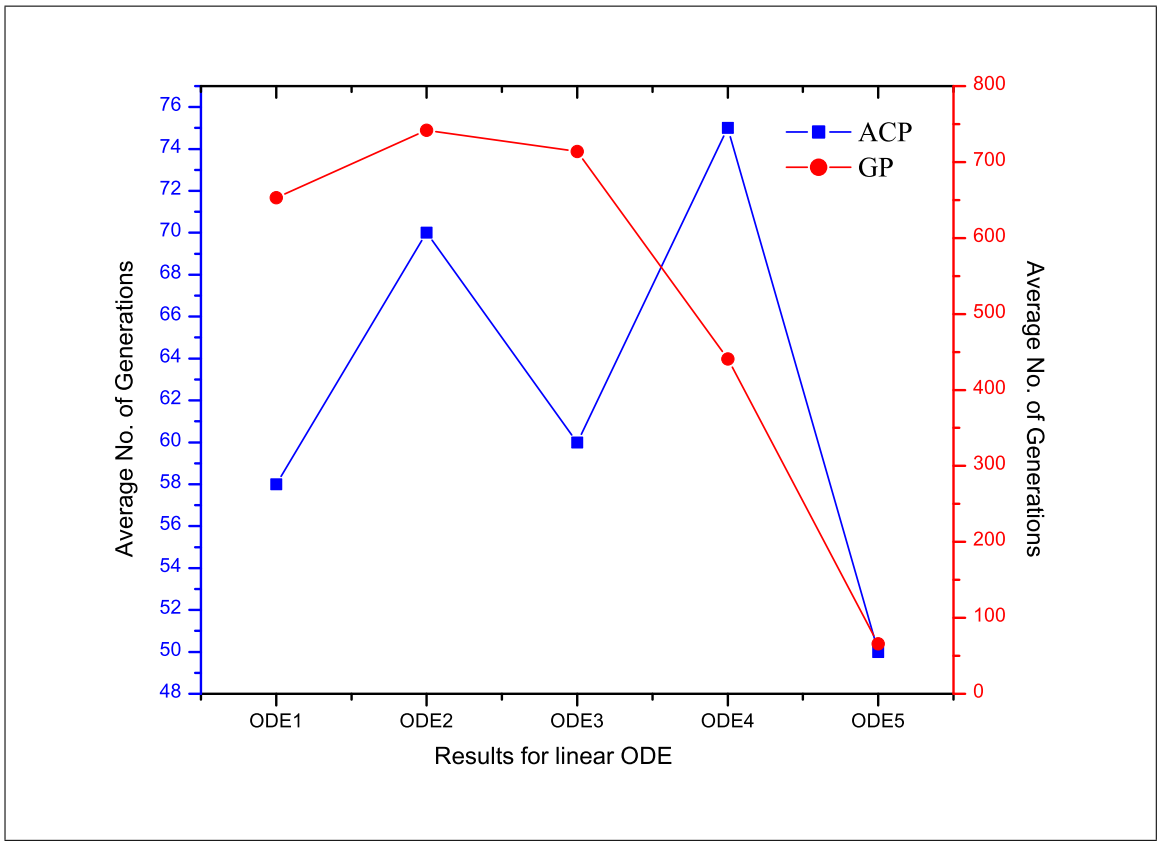

Figure 3.3: Comparison between ACP and GP.

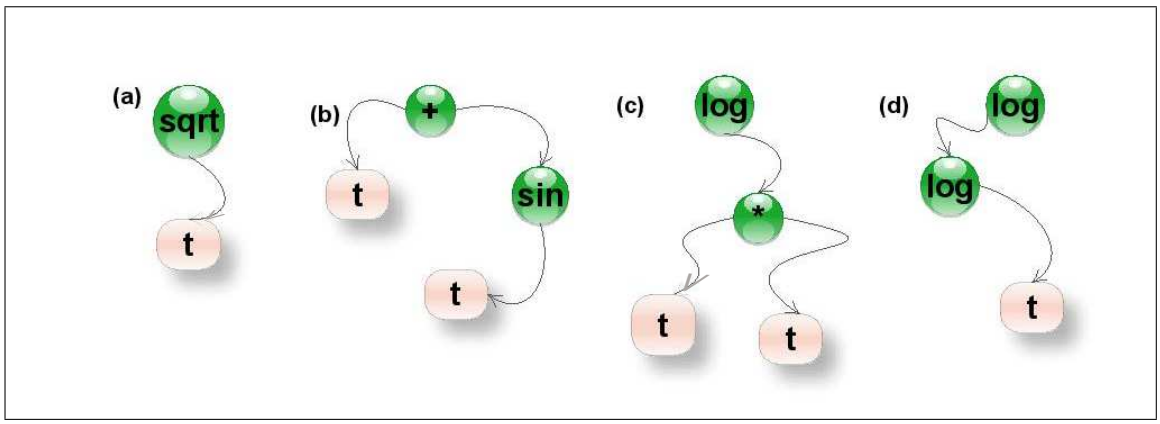

Figure 3.4: Parse tree solution for (a) NLODE1, (b) NLODE2, (c) NLODE3 and (d) NLODE4.

3.2.2 Non-linear Ordinary Differential Equations

In this work, we present the results for solving selected non-linear ODEs. The ACP is applied in each equation and in every experiment the analytical solution is found. The solution is obtained using ACP and shown in the parse trees structure.

NLODE1

$$y^{'} = \frac{1}{2y}$$

with $y(1)$ =1 and $t \in [1, 4]$. The tour of the ant is $sqrt(t)$. The parse tree of the tour is given in Figure. 3.4(a). Expression extracted from the parse tree is given as $y(t) = \sqrt{t}$.

NLODE2

$$\left(y^{'}\right)^{2} + log\,(y) - cos^{2}\,(t) - 2cos\,(t) - 1 - log\,(t + sin\,(t)) = 0$$

with $y(1) = 1 + sin(1)$ and $t \in [1, 2]$. The tour of the ant is $t + sin(t)$. The parse tree of the tour is given in Figure 3.4(b). Expression extracted from the parse tree is given as $y(t) = t + sin(t)$.

NLODE3

$$y^{''}y^{'} = -\frac{4}{t^{3}}$$

with $y(1) = 0$ and $t \in [1, 2]$. The tour of the ant is $log(t * t)$. The parse tree of the tour is given in Figure 3.4(c). Expression extracted from the parse tree is given as $y(t) = log(t^{2})$.

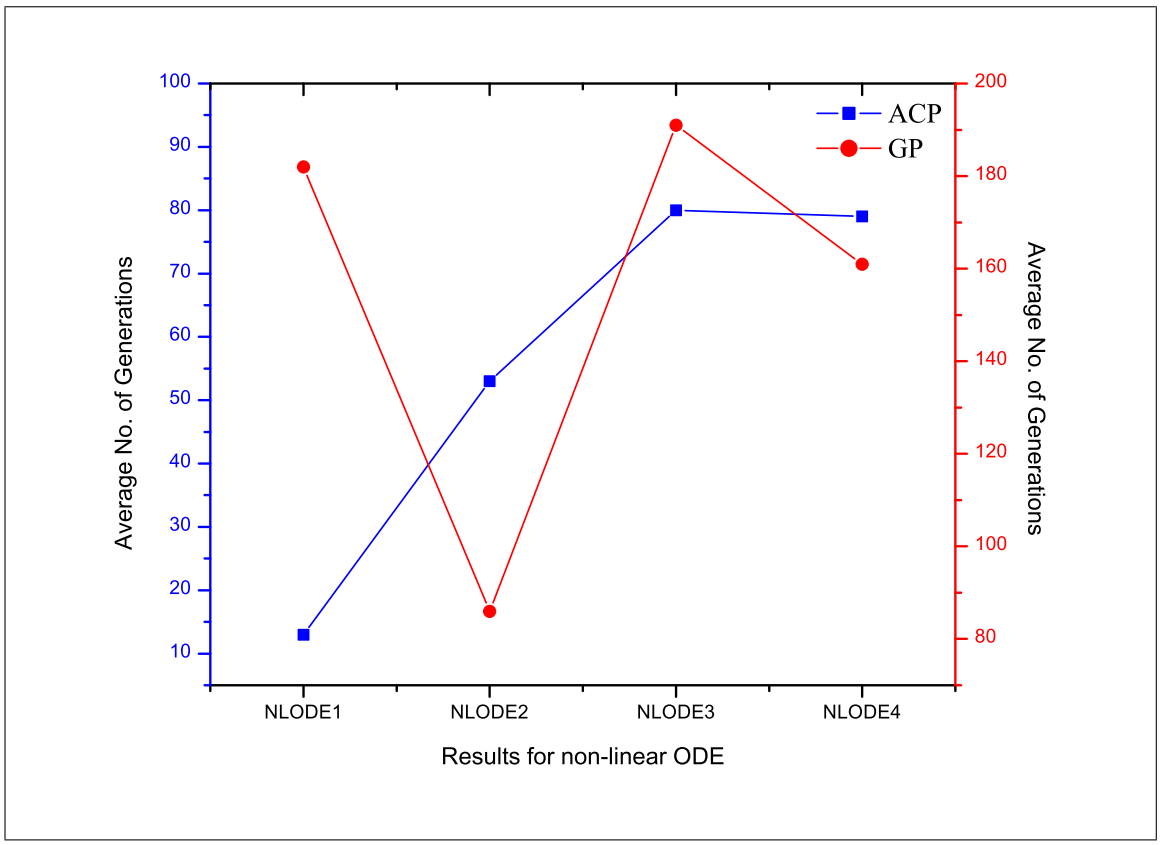

Figure 3.5: Comparison between ACP and GP.

NLODE4

$$t^2y'' + \left(ty'\right)^2 + \frac{1}{log\,(t)} = 0$$

with $y(e) = 0$, $y\prime(e) = \frac{1}{e}$ and $t \in [e, 2e]$. The tour of the ant is $loglog(t)$. The parse tree of the tour is given in Figure 3.4(d). Expression extracted from the parse tree is given as $y(t) = log(log(t))$.

In Table 3.6, the average number of colonies and ants together with the time are shown in order to obtain the exact solutions for the NLODEs. In Figure 3.5, for the non-linear ODEs case, the average number of generations increase for the ACP method as the given problem getting more difficult. Comparison with the GP method (Tsoulos & Lagaris, 2006) still showed that the ACP gives faster solution through out the non-linear ODEs case.

3.2.3 Systems of Ordinary Differential Equations

In this subsection, the results for solving the systems of ODEs are obtainded by using the ACP and in every experiment the analytical solution is found. The solution is obtained

Table 3.6: Results for non-linear ODEs using ACP.

NLODE	Average No. Generations	Average No. Ants	Average Time(secs)
NLODE1	13	15	3.24
NLODE2	53	68	11.52
NLODE3	80	60	28.86
NLODE4	79	65	25.32

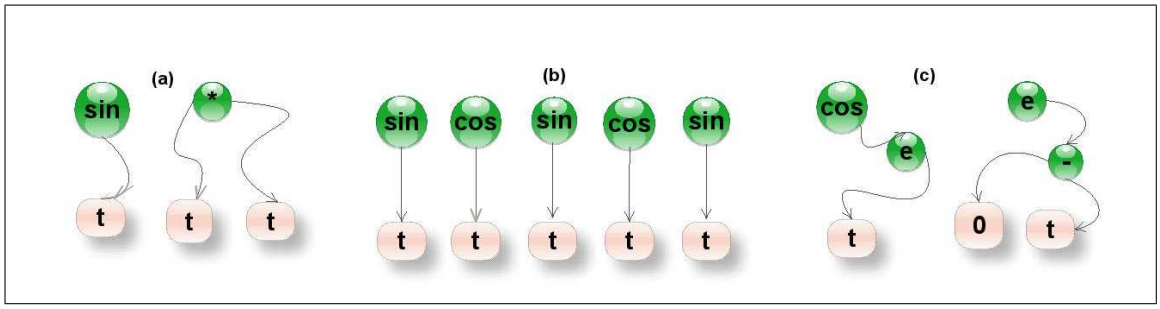

Figure 3.6: Parse tree solution for (a) SODE1, (b) SODE2 and (c) SODE3.

using ACP and shown in the parse trees structure.

SODE1

$$y_1' = \ cos(t) + y_1^2 + y_2 - (t^2 + sin^2(t)),\ y_2' = 2t - t^2 sin(t) + y_1 y_2$$

with $y_1(0) = 0$, $y_2(0) = 0$ and $t \in [0,\ 1]$. The tours of the ants are $sin(t)$ and $(t * t)$. The parse trees of the tours are given in Figure 3.6(a). Expressions extracted from the parse trees are given as $y_1 = sin(t)$ and $y_2 = t^2$.

SODE2

$$y_1^{'} \ = \ \cos(t),\ y_2^{'} = -y_1,\ y_3^{'} = y_2,\ y_4^{'} = -y_1,\ y_5^{'} = y_2,$$

with $y_1(0) = 0$, $y_2(0) = 1$, $y_3(0) = 0$, $y_4(0) = 1$, $y_5(0) = 0$ and $t \in [0,\ 1]$. The tours of the ants are $sin(t)$, $cos(t)$, $sin(t)$, $cos(t)$ and $sin(t)$. The parse trees of the tours are given in Figure 3.6(b). Expression extracted from the parse trees are given as $y_1 = sin(t)$, $y_2 = cos(t)$, $y_3 = sin(t)$, $y_4 = cos(t)$ and $y_5 = sin(t)$.

SODE3

$$y_1^{'} \ = \ -\frac{1}{y_2}\sin(\exp(t)),\ y_2^{'} = -y_2,$$

with $y_1(0) = cos(1.0)$, $y_2(0) = 1$ and $t \in [0, 1]$. The tours of the ants are $cosexp(t)$ and $exp(0 - t)$. The parse trees of the tours are given in Figure 3.6(c). Expression extracted

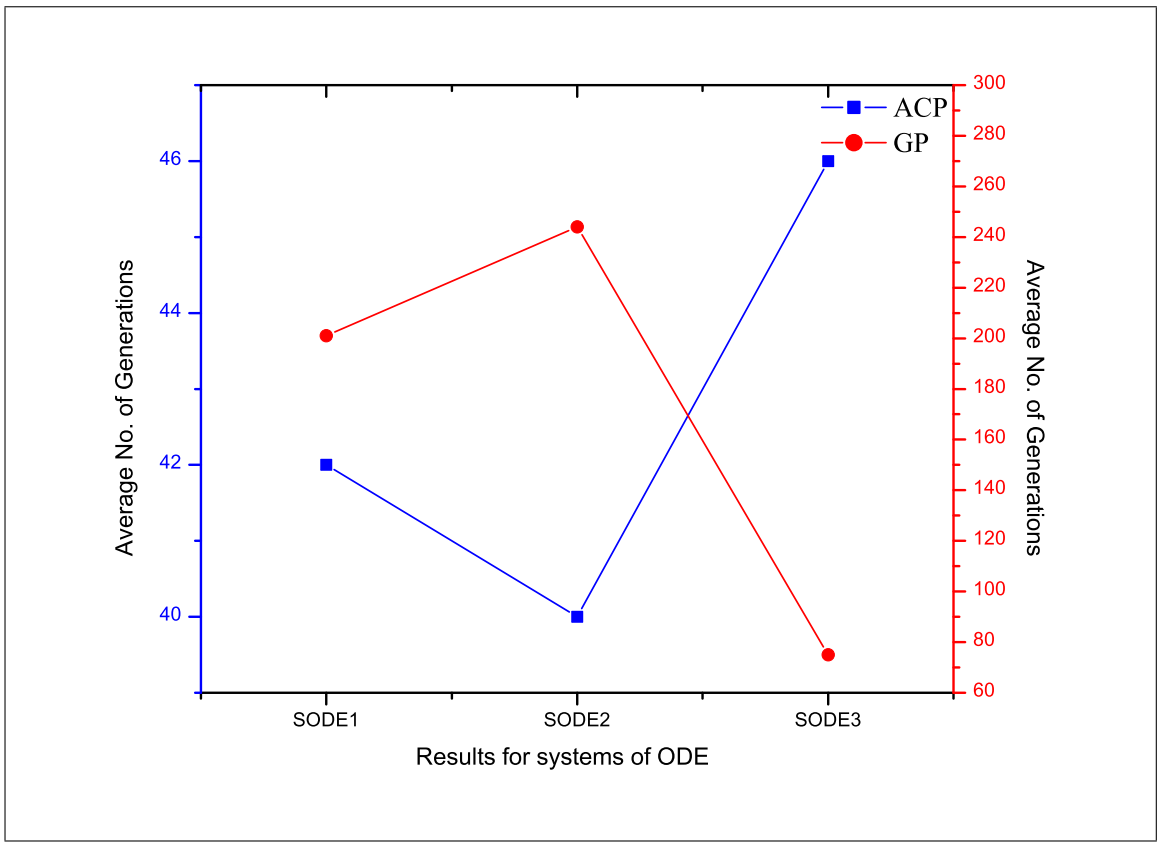

Figure 3.7: Comparison between ACP and GP.

Table 3.7: Results for systems of ODEs using ACP

SODE	Average No. Generations	Average No. Ants	Average Time(secs)
SODE1	42	35	5.64
SODE2	40	45	10.52
SODE3	46	50	15.36

from the parse trees are given as $y_1 = cos(exp(t))$ and $y_2 = exp(-t)$. In Table 3.7, the average number of generations and ants together with the time are shown in order to obtain the exact solutions for the SODEs. In Figure 3.7, comparison between the ACP and the GP method (Tsoulos & Lagaris, 2006) are depicted. In the systems of ODEs, solutions obtained are more than one, therefore the task for searching the solution using this non-traditional method will be very difficult. From the figure, the ACP method succeeded in finding the solutions within the range of 40-50 but the GP method compute the solutions within the range of 70-250.

3.2.4 Partial Differential Equations.

In this work, we present the results for solving the PDEs. The ACP is applied in each equation and in every experiment, the analytical solution is found. The solution is obtained using ACP and shown in the parse trees structure.

PDE1

$$\nabla^2\Psi(x,y) = exp(-x)(x-2+y^3+6y),$$

with $x \in [0,1]$ and $y \in [0,1]$ and boundary conditions: $\Psi(0,y) = y^3$, $\Psi(1,y) = (1+y^3)exp(-1)$, $\Psi(x,0) = xexp(-x)$ and $\Psi(x,1) = (x+1)exp(-x)$. The tour of the ant is $(x+y*y*y)exp(0-x)$. The parse tree of the tour is given in Figure 3.8(a). Expression extracted from the parse tree is given as $\Psi(x,y) = (x+y^3)exp(-x)$.

PDE2

$$\nabla^2\Psi(x,y) = -2\Psi\left(x,y\right),$$

with $x \in [0,1]$ and $y \in [0,1]$ and boundary conditions: $\Psi(0,y) = 0, \Psi(1,y) = sin(1)cos(y)$, $\Psi(x,0) = sin(x), \Psi(x,1) = sin(x)cos(1)$. The tour of the ant is $sin(x)cos(y)$. The parse tree of the tour is given in Figure 3.8(b). Expression extracted from the parse tree is given as $\Psi(x,y) = sin(x)cos(y)$.

PDE3

$$\nabla^2\Psi(x,y) = 4,$$

with $x \in [0,1]$ and $y \in [0,1]$ and boundary conditions: $\Psi(0,y) = y^2+y+1$, $\Psi(1,y) = y^2+y+3$, $\Psi(x,0) = (x^2+x+1)$ and $\Psi(x,1) = x^2+x+3$. The tour of the ant is $(x*x+y*y+x+y+1)$. The parse tree of the tour is given in Figure 3.8(c). Expression extracted from the parse tree is given as $\Psi(x,y) = x^2+y^2+x+y+1$.

PDE4

$$\nabla^2\Psi(x,y) = -\left(x^2+y^2\right)\Psi\left(x,y\right),$$

with $x \in [0,1]$ and $y \in [0,1]$ and boundary conditions: $\Psi(x,0) = 0$, $\Psi(x,1) = sin(x)$, $\Psi(0,y) = 0$ and $\Psi(1,y) = sin(y)$. The tour of the ant is $sin(x*y)$. The parse tree of the tour is given in Figure 3.8(d). Expression extracted from the parse tree is given as $\Psi(x,y) = sin(xy)$.

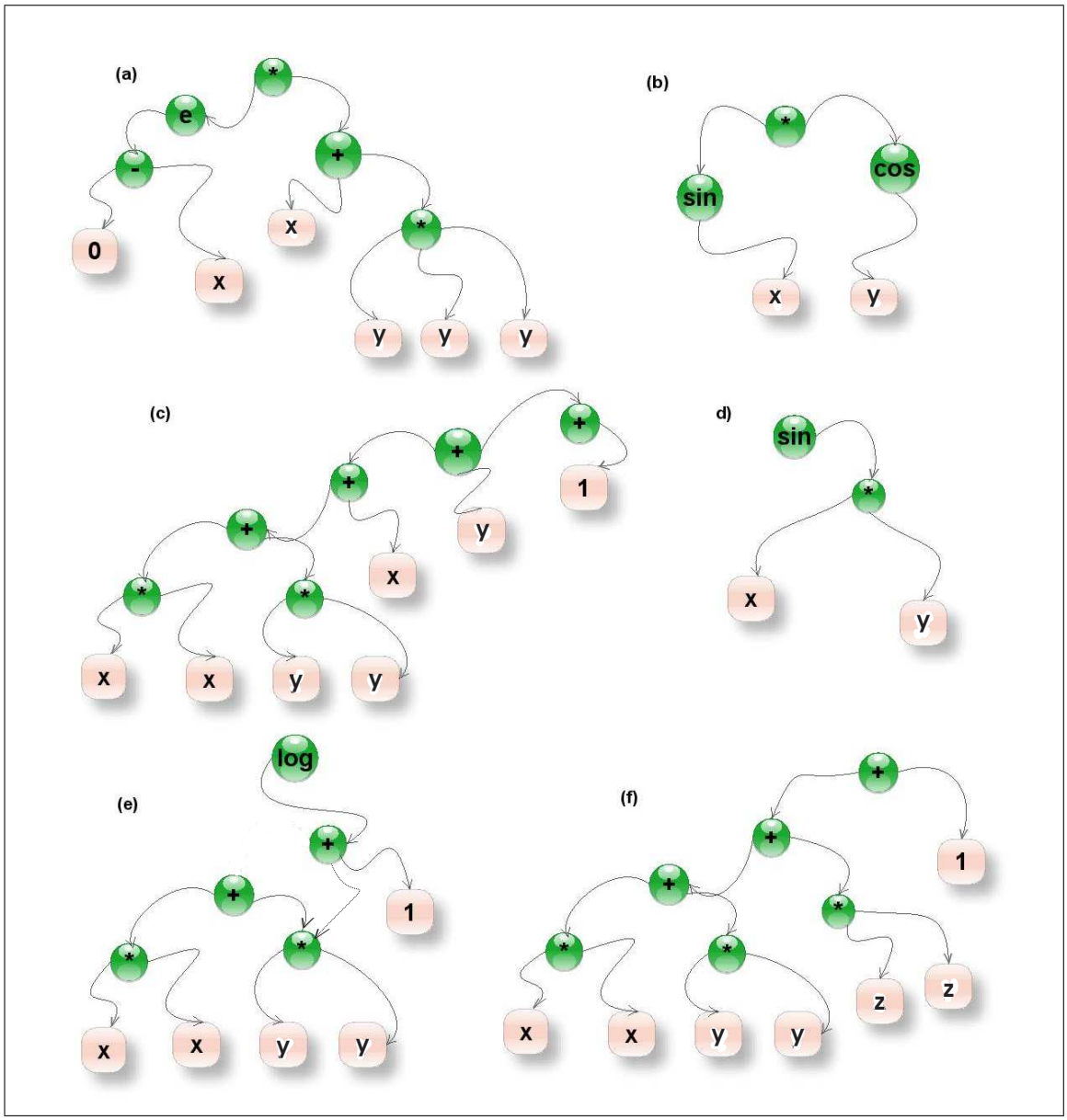

Figure 3.8: Parse tree solution for (a) PDE1, (b) PDE2, (c) PDE3, (d) PDE4, (e) PDE5 and (f) PDE6.

Table 3.8: Results for PDEs using ACP

PDE	Average No. Generations	Average No. Ants	Average Time(secs)
PDE1	81	75	15.57
PDE2	75	65	8.52
PDE3	85	77	16.33
PDE4	56	50	4.42
PDE5	77	70	15.46
PDE6	82	73	17.52

PDE5

$$\nabla^2\Psi(x,y) + exp(\Psi(x,y)) = 1 + x^2 + y^2 + \frac{4}{(1+x^2+y^2)^2},$$

with $x \in [-1,1]$ and $y \in [-1,1]$ and boundary conditions: $f(0,y) = log(1+y^2)$, $f(1,y) = log(2+y^2)$, $g(x,0) = log(1+x^2)$ and $g(x,1) = log(2+x^2)$. The tour of the ant is $log(1+x*x+y*y)$. The parse tree of the tour is given in Figure 3.8(e). Expression extracted from the parse tree is given as $\Psi(x,y) = log(1+x^2+y^2)$.

PDE6

$$\nabla^2\Psi(x,y,z) = 6$$

with $x \in [0,1]$, $y \in [0,1]$ and $z \in [0,1]$ and boundary conditions: $\Psi(0,y,z) = y^2+z^2$, $\Psi(1,y,z) = y^2+z^2+1$, $\Psi(x,0,z) = x^2+z^2$, $\Psi(x,1,z) = x^2+z^2+1$, $\Psi(x,y,0) = x^2+y^2$ and $\Psi(x,y,1) = x^2+y^2+1$. The tour of the ant is $(x*x+y*y+z*z+1)$. The parse tree of the tour is given in Figure 3.8(f). Expression extracted from the parse tree is given as $\Psi(x,y) = x^2+y^2+z^2+1$. In Table 3.8, the average number of generations and ants together with the time are shown in order to obtain the exact solutions for the PDEs. In Figure 3.9, the ACP method shows that as the solution is getting more complicated to obtain, the average number of generations will increase. This can be observed and analyzed from each of the PDE problems given above. For example at PDE4, where the analytical solution is quite simple, the average number of generations obtained for finding the solution by using ACP is 56 whereas in the GP method (Tsoulos & Lagaris, 2006) is about 207. It shows that the proposed ACP method works faster than the GP method.

3.2.5 System of Partial Differential Equations.

In the following, the proposed ACP algorithm is implemented to solve the system of PDE's. The first example are related to linear case whereas the second problem involves

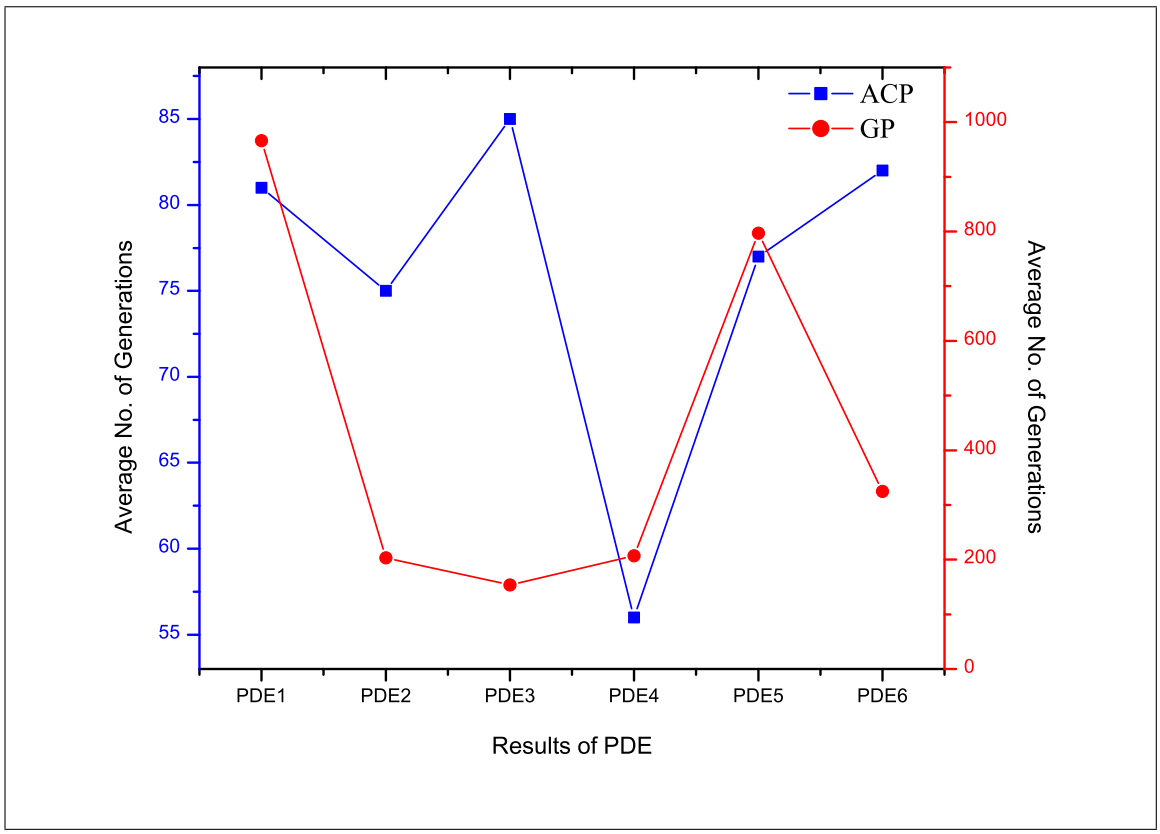

Figure 3.9: Comparison between ACP and GP.

nonlinear ones. In each experiment, the analytical solution is obtained.

SPDE1

Given a linear system of PDEs

$$u_t + u_x - 2v = 0, \quad v_t + v_x - 2u = 0,$$

where the initial condition are given as $u\,(x,0) = sin\,(x)$ and $v\,(x,0) = cos\,(x)$. The tours of the ants are obtained as $sin\,(t+x)$ and $cos\,(x+t)$. The parse trees of the tours are given in Figure 3.10(a). Expressions extracted from the parse trees are given as $u = sin(t+x)$ and $v = cos(x+t)$.

SPDE2

Consider this non-linear case where

$$u_t + vu_x + u = 1, \quad v_t + uv_x - v = -1,$$

with the initial data given as $u(x,0) = e^x$and $v(x,0) = e^{-x}$. The tours of the ants are obtained as $e(x-t)$, $e(t-x)$. The parse trees of the tours are given in Fig. 3.10(b). Expressions extracted from the parse trees are given as each $u = e^{x-t}$ and $v = e^{t-x}$. The average number of colonies and ants together with the time taken to obtain the exact

Table 3.9: Results for PDEs using ACP

SPDE	Average No. Generations	Average No. Ants	Average Time(secs)
SPDE1	88	75	33.70
SPDE2	95	80	48.72

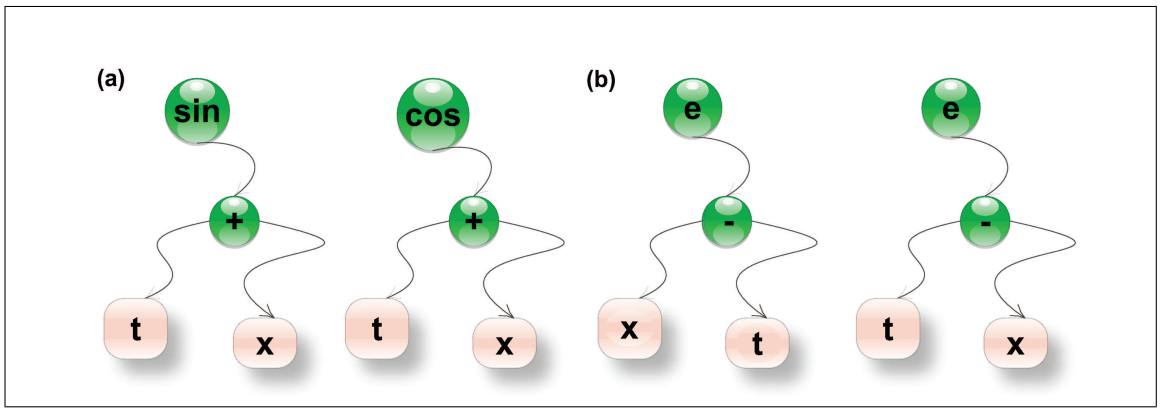

Figure 3.10: Parse tree solution for (a)SPDE1, (b)SPDE2.

solutions for the SPDE's are depicted in Table 3.9. From Table 3.9, the ACP approach predicted the solutions within reasonable computational time.

CHAPTER 4

SOLUTION OF MATRIX RICCATI DIFFERENTIAL EQUATION AND OTHER FUZZY MODELLING PROBLEMS USING MODIFIED ANT COLONY PROGRAMMING

In this chapter, we implement the modified ACP method to compute the solution of MRDE. Solving MRDE will lead to the key for obtaining the optimal feedback with the minimum cost function. The capability of the ACP method is further tested out by solving some fuzzy modelling problems such as in the engineering and biological fields. From these studies that have been carried out, the modified ACP computes solutions which are either exact or approximately close enough to the analytical solutions. The modifed ACP suggests simpler solutions with very good accuracy for solving complicated differential equations.

4.1 Modified ant colony programming for solving MRDE with linear singular fuzzy system: singular cost and cross term

The linear singular system is given as

$$E\dot{x}\left(t\right) = Ax\left(t\right) + Bu\left(t\right),$$

where E is a singular matrix, while A and B are vector valued functions for x (the state vector) and u (the control vector), respectively. Since E is singular, there are some limitations or conditions on x imposed by the above equation, and because of this reason, x is sometimes referred to as a semi-state or a descriptor variable. Singular systems are also referred as descriptor, generalized, or differential-algebraic systems. A singular system is a combination of algebraic and differential equations. Due to this combination, the algebraic parts represent the constraints to the solution of the differential equation

and this led to a lot of disadvantages either in analytical or numerical treatment of such systems, especially when there is a need for their control. The system appears as a linear approximation of system models in many applications such as robotics, biology, aircraft dynamics, etc. In real life applications, physical systems are complicated in their system structure, and are extremely challenging to model using precise mathematical equations. Therefore, another method is needed to represent these complicated physical systems by implementing approximate modeling. How can a good approximation in a system produce good, reasonable and satisfactory outputs if we have imprecise information or if the system itself is too complicated to be described? The key to this answer is the fuzzy logic and the interval mathematics. Both can be applied in mathematical modeling to represent or to describe many complicated systems. These are known as the fuzzy systems modeling. Fuzzy systems range from fuzzy linear systems and fuzzy differential equations to control chaotic systems etc. In a fuzzy linear system,

$$A\tilde{x} = \tilde{b},$$

where A is a $n \times n$ singular matrix and $\tilde{b}$ refers to a vector of fuzzy numbers in parametric form.

In the present work, the modified ACP is used to find the solution of MRDE for the linear fuzzy singular system with singular cost (R=0) and cross term (R=1) scenarios. The optimal control problem is considered where we need to minimize the cost function given below,

$$J = \tfrac{1}{2}x^T(t_f)E_i^T S E_i x(t_f) + \tfrac{1}{2}\int_0^{t_f}[x^T(t)Qx(t) + u^T(t)Ru(t)]dt,$$

subject to the linear singular fuzzy system R^i : If x_j is (μ_{ji}, σ_{ji}), $i = 1, 2, ..r$ and $j = 1, 2, ..n$, then

$$E_i\dot{x}(t) = A_i x\,(t) + B_i u\,(t)\,, \qquad x\,(0) = x_0,$$

where

$$S = \begin{bmatrix} 3 & 0 \\ 0 & 0 \end{bmatrix}, E_i = \begin{bmatrix} 3 & 0 \\ 0 & 0 \end{bmatrix}, A_1 = \begin{bmatrix} -1 & -1 \\ 0 & 1 \end{bmatrix},$$

$$A_2 = \begin{bmatrix} -2 & -2 \\ 0 & 2 \end{bmatrix}, B_i = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, Q = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}.$$

The numerical implementation can be adapted by taking $t_f = 2$ for solving the MRDE of the above linear singular fuzzy system. The above matrices are substituted in the equation below and these are later transformed into a set of nonlinear differential equations in k_{11} and k_{12}. Then, the optimal control can be obtained by the solution of MRDE,

$$E_i^T \dot{K}_i(t) E_i + E_i^T K_i(t) A_i + A_i^T K_i(t) E_i + Q - E_i^T K_i(t) B_i R^{-1} B_i^T K_i(t) E_i = 0.$$

Both k_{11} and k_{12} are computed simultaneously by using the modified ACP. In this ACP approach, the construction graph has 18 nodes: ($\mathbb{T} = \{0, 1, 2, 3, 4, .., 9, t\}$) and ($\mathbb{F} = \{+, -, *, /, (,), exp\}$). In the first generation, 50-100 ants are sent to visit 7-10 nodes. The ants start from any of the nodes randomly, until the ants reach to the limit where the terminal condition is satisfied. Although the value for the fitness function may not be close to zero, but the path or tour that have been taken by the ants might lead to the final solution. Therefore after completion of each generation, a global update of pheromone trail takes place in order to increase the pheromone value on the solution path. This significant piece of information will be used for the next generation. Furthermore, the number of nodes can also be increased automatically one at a time if the ants could not find any combinations of expression which can satisfy the fitness function in the current or present nodes. The above process will be repeated several times until the final solution is obtained. Working on this MRDE problem, the expression is generated randomly up to 22-26 nodes, where $\rho = 0.5$, $\tau_{ij}(0) = 0.2$ and $\beta = 1$. After 22-26 nodes, the expression satisfies the terminal condition as well as the fitness function. In Figures 4.1 and 4.2, the evolution of trial solutions for the above problem are shown. These trial solutions are compared with the exact solutions.

4.1.1 R=0

For $k_{11}(t)$, at generation 15, with fitness value equal to 0.01235, the intermediate solution was:

$$\text{Tours: } k_{11}(t) = e(2/3 * t);$$

$$\text{Expressions: } k_{11}(t) = e^{\frac{2}{3}t}.$$

Next, at the 45^{th} generation, again with the same fitness value 0.01235, the corresponding candidate solution was:

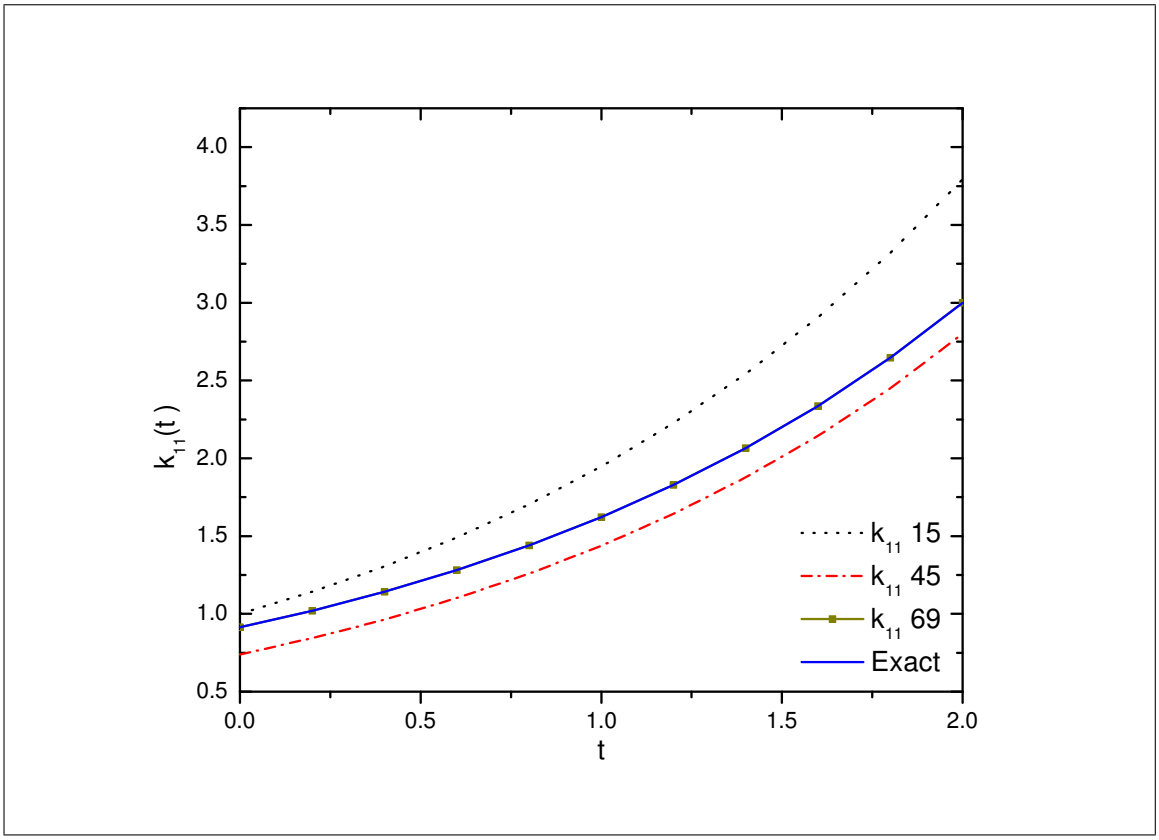

Figure 4.1: Candidate solutions for $k_{11}(t)$ by ACP for various generations and comparison with the exact solution.

$$\text{Tours: } k_{11}(t) = e((2 * t - 4)/3) * 7 * 2/5;$$

$$\text{Expressions: } k_{11}(t) = \tfrac{14}{5} e^{\frac{2t-4}{3}}.$$

Finally, at the 69^{th} generation, the ACP computed the solution with fitness value less than $1.0e^{-09}$, with its functional form given as

$$\text{Tours: } k_{11}(t) = e((2 * t - 4)/3) * 5 * 3 + 2/6 + 1/6;$$

$$\text{Expressions: } k_{11}(t) = \tfrac{17}{6} e^{\frac{2t-4}{3}} + \tfrac{1}{6}.$$

The ACP method was applied to obtain the solution for $k_{12}(t)$, where the first trial solution was obtained at the 19^{th} generation, similar to the $k_{11}(t)$ with the fitness function is 0.01235. Here,

$$\text{Tours: } k_{12}(t) = 0 - e(2/3 * t);$$

$$\text{Expressions: } k_{12}(t) = -e^{\frac{2}{3}t}.$$

Later, the ACP predicted another trial solution with the same fitness value at the 56^{th} generation. The candidate solution was given as

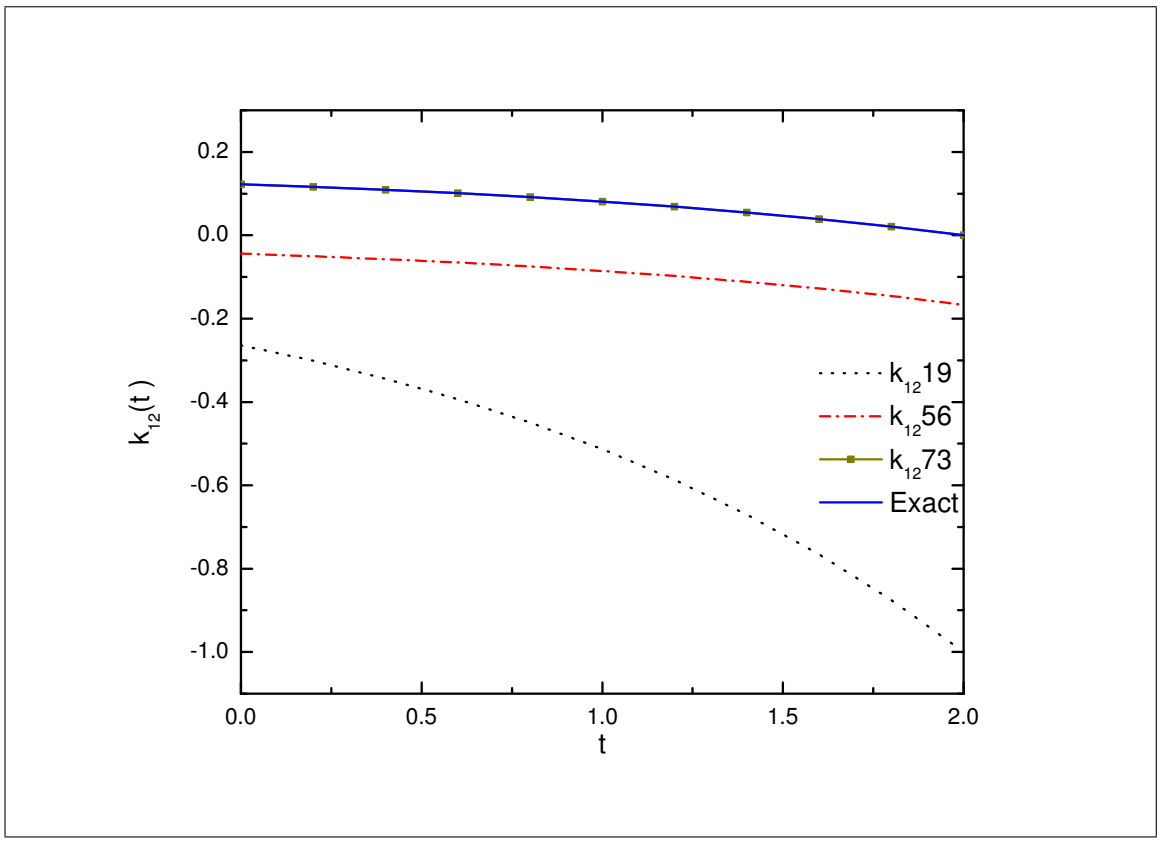

Figure 4.2: Candidate solutions for $k_{12}(t)$ by ACP for various generations and comparison with the exact solution.

$$\text{Tours: } k_{12}(t) = 0 - e((2 * t - 4)/3) * (1/6);$$

$$\text{Expressions: } k_{12}(t) = \tfrac{-1}{6} e^{\frac{2t-4}{3}}.$$

Finally, at the 73^{rd} generation the solution is achieved where the fitness function is less than $1.0e^{-09}$. Here,

$$\text{Tours: } k_{12}(t) = 0 - e((2 * t - 4)/3) * (1/6) + 1/6;$$

$$\text{Expressions: } k_{12}(t) = \tfrac{-1}{6} e^{\frac{2t-4}{3}} + \tfrac{1}{6}.$$

The parse trees for the solutions $k_{11}(t)$ and $k_{12}(t)$ are shown in Figures 4.3 and 4.4, respectively. The numerical solutions, which are given in Table 4.1, show that the ACP approach predicts solutions which are equivalent to the analytical ones. However, the RK4 method showed some differences with the exact solutions especially at 7 to 9 decimal places at $t \geq 0.8$.

4.1.2 R=1

For searching the solution of MRDE for linear singular fuzzy system with cross term (R=1) using the ACP, 50-100 digital ants are sent throughout the space graph, to visit

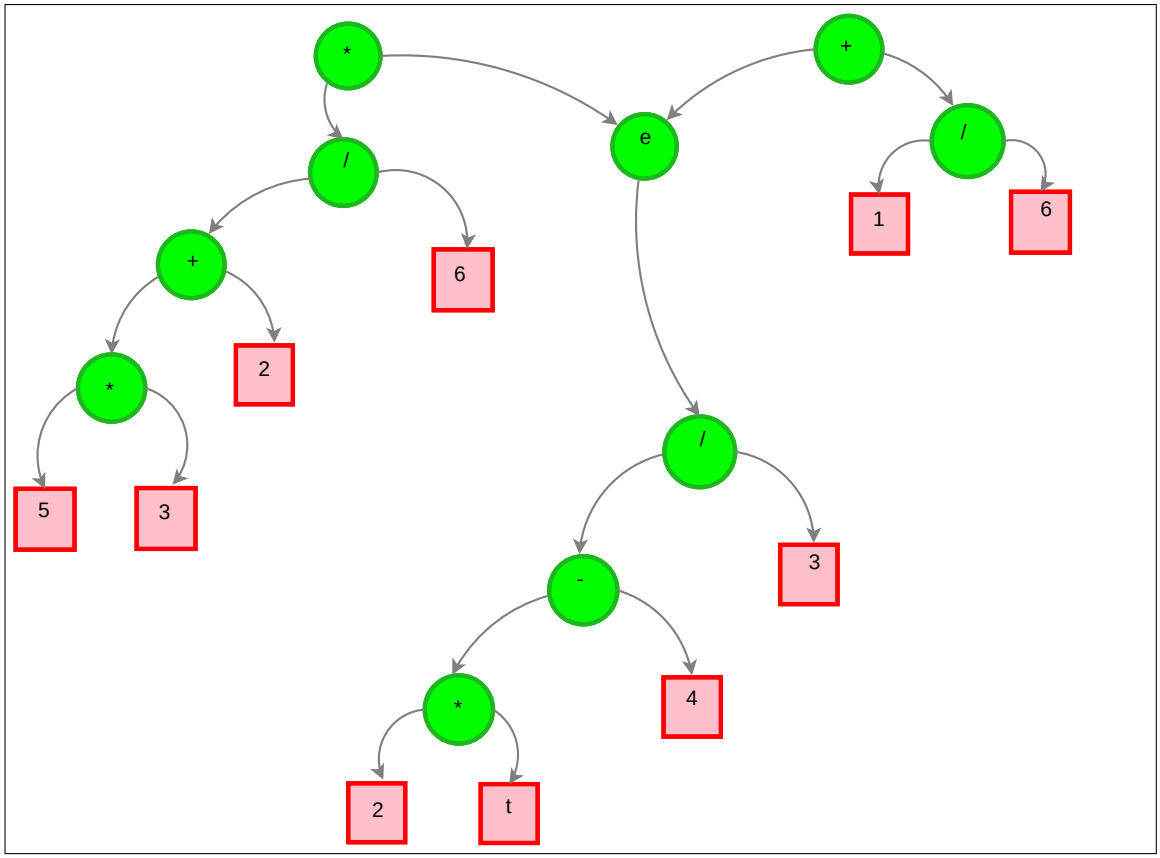

Figure 4.3: Parse tree for $k_{11}\,(t)$

Table 4.1: Numerical solutions for $k_{11}(t)$ and $k_{12}(t)$ when R=0.

	ACP		RK4		Exact	
t	$k_{11}\,(t)$	$k_{12}\,(t)$	$k_{11}\,(t)$	$k_{12}\,(t)$	$k_{11}\,(t)$	$k_{12}\,(t)$
0.0	0.913525225	0.122733810	0.913525238	0.122733810	0.913525225	0.122733810
0.2	1.020050267	0.116467631	1.020050269	0.116467631	1.020050267	0.116467631
0.4	1.141769063	0.109307702	1.141769063	0.109307702	1.141769063	0.109307702
0.6	1.280848709	0.101126547	1.280848707	0.101126547	1.280848709	0.101126547
0.8	1.439765398	0.091778506	1.439765393	0.091778506	1.439765398	0.091778506
1.0	1.621348504	0.081097147	1.621348496	0.081097147	1.621348504	0.081097147
1.2	1.828830955	0.068892297	1.828830942	0.068892298	1.828830955	0.068892297
1.4	2.065906797	0.054946659	2.065906779	0.054946660	2.065906797	0.054946659
1.6	2.336796959	0.039011944	2.336796933	0.039011945	2.336796959	0.039011944
1.8	2.646324404	0.020804447	2.646324370	0.020804449	2.646324404	0.020804447
2.0	3.000000000	0.000000000	3.000000000	0.000000000	3.000000000	0.000000000

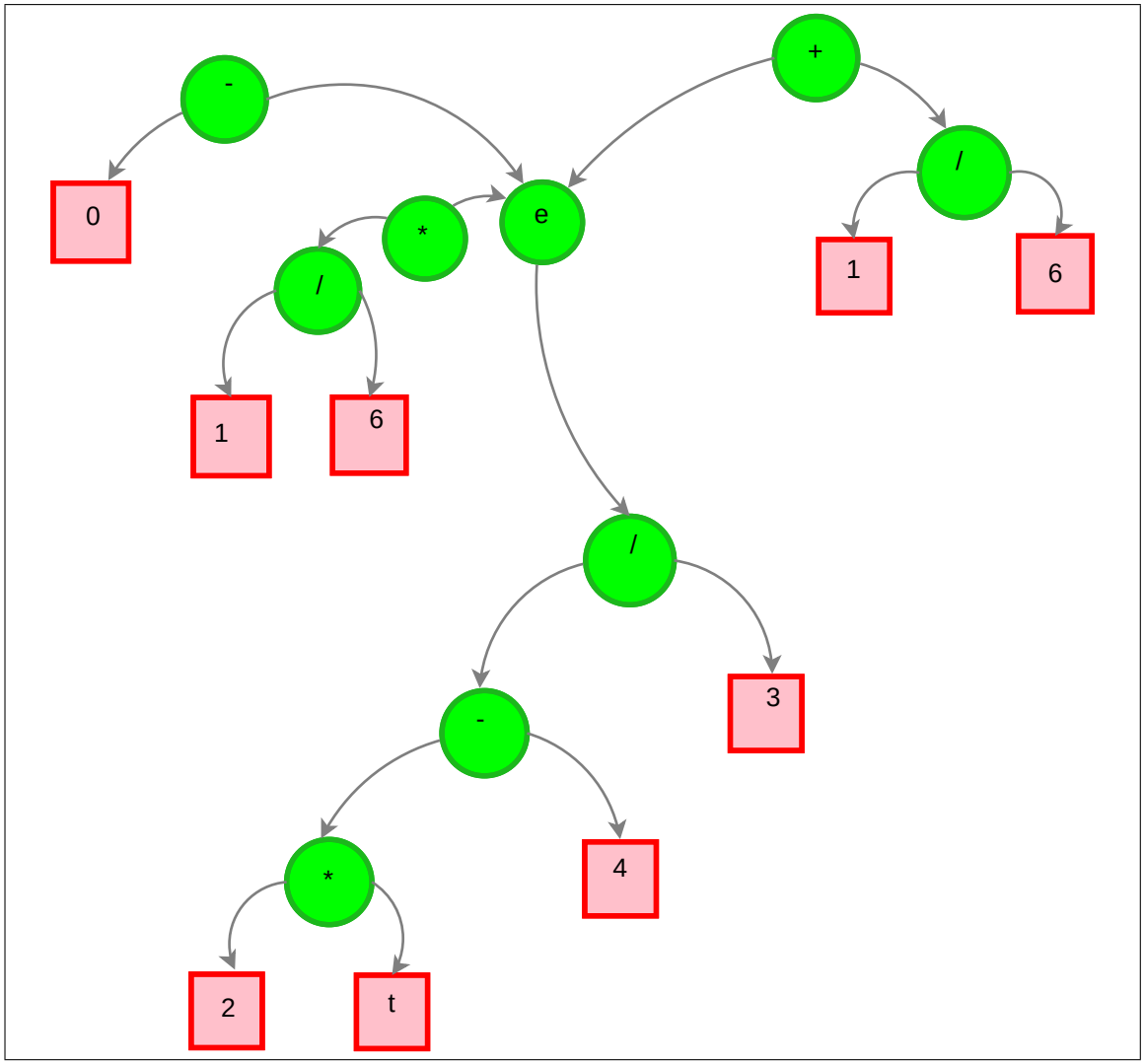

Figure 4.4: Parse tree for $k_{12}\,(t)$

15 nodes. At 23^{rd} generation, with fitness function equal to 49, the ACP predicted an intermediate solution which was:

$$\text{Tours: } k_{11}(t) = 3/(2 * e(2-t) - 1));$$

$$\text{Expressions: } k_{11}(t) = \frac{3}{2e^{2-t}-1}\ .$$

Then, after the global pheromone update, the ACP method gives an expression with the fitness function is equal to 1 and its functional form is given as:

$$\text{Tours: } k_{11}(t) = 3/(3 * e((8-4*t)/3) - 2);$$

$$\text{Expressions: } k_{11}(t) = \frac{3}{3e^{\frac{8-4t}{3}}-2}.$$

at 47^{th}generation. Then at 87^{th} generation, again after updating the global pheromone values, a new expression is predicted with the fitness function equal to $6.67e^{-6}$. The candidate solution is given as:

$$\text{Tours: } k_{11}(t) = 4/(7 * 3/5 * e((8-4*t)/3) - 3);$$

$$\text{Expressions: } k_{11}(t) = \frac{4}{\frac{21}{5}e^{\frac{8-4t}{3}}-3}.$$

Finally at the 93^{rd} generation, the ACP predicted an expression with a fitness function less than to $1.0e^{-9}$, with its functional form given as:

$$\text{Tours: } k_{11}(t) = 4/(6 * 2 + 1/3 * e((8-4*t)/3) - 3);$$

$$\text{Expressions: } k_{11}(t) = \frac{4}{\frac{13}{3}e^{\frac{8-4t}{3}}-3}.$$

The parse tree for the solutions k_{11} is shown in Figure 4.5. The comparison between the ACP, RK4 and the analytical solutions are given in Table 4.2. The differences between RK4 and the other methods are clearly seen in the data starting from 3 to 4 decimal places. In Figure 4.6, the candidature solutions are compared with the analytical solutions. The ACP method improves the quality of its calculations based on the fitness functions in order to obtain the final solutions. Therefore, the solution of MRDE for linear singular fuzzy system with cross term and singular cost for the matrix A_2 can also be obtained by using the modified ACP method.

Table 4.2: Numerical solutions for $k_{11}(t)$ when R=1.

t	ACP k_{11}	RK4 k_{11}	Exact k_{11}
0.0	0.067379804	0.067431300	0.067379804
0.2	0.089351336	0.089420571	0.089351336
0.4	0.119096627	0.119190746	0.119096627
0.6	0.159856850	0.159986532	0.159856850
0.8	0.216647867	0.216829853	0.216647867
1.0	0.297636154	0.297898000	0.297636154
1.2	0.417045908	0.417435458	0.417045908
1.4	0.602045991	0.602647984	0.602045991
1.6	0.911863579	0.912766058	0.911863579
1.8	1.505104706	1.506414596	1.505104706
2.0	3.000000000	3.000000000	3.000000000

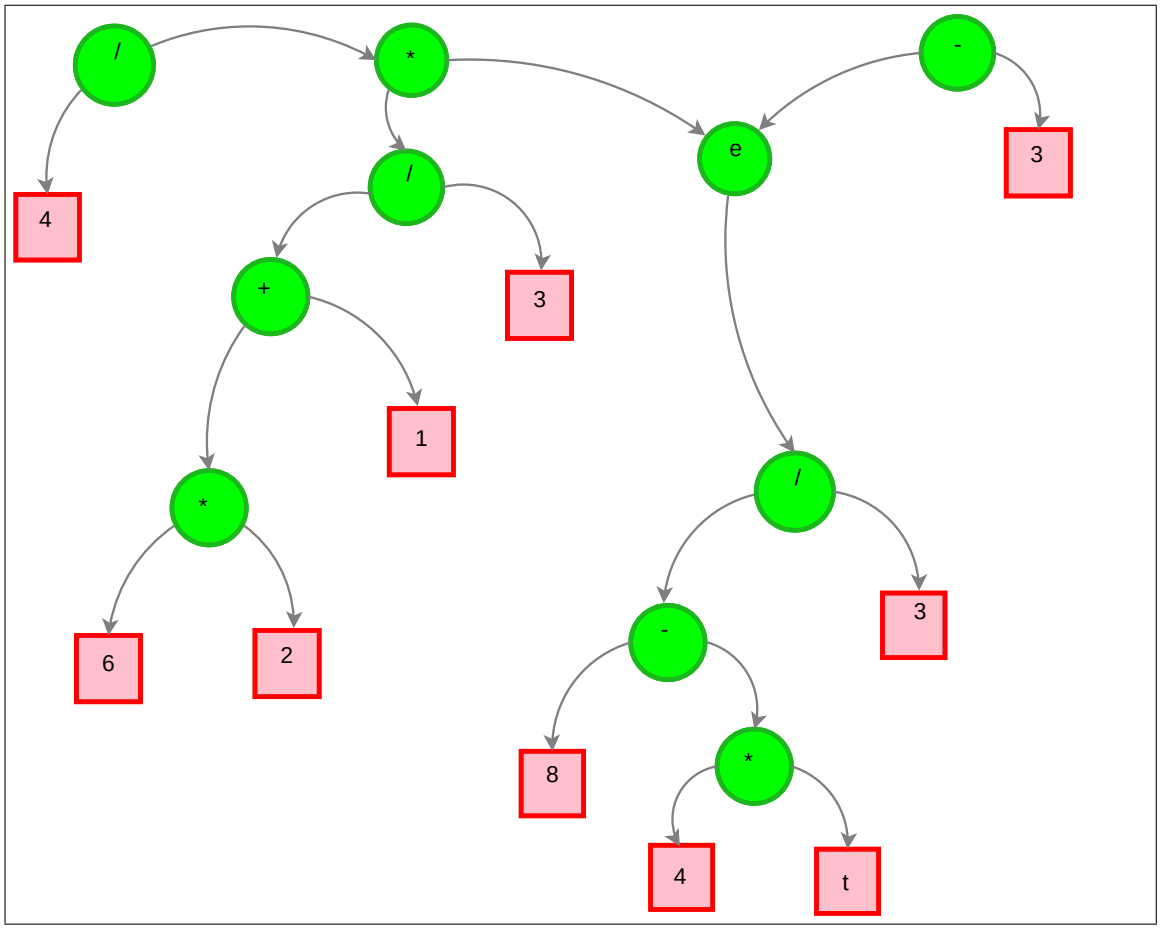

Figure 4.5: Parse tree for $k_{11}\,(t)$

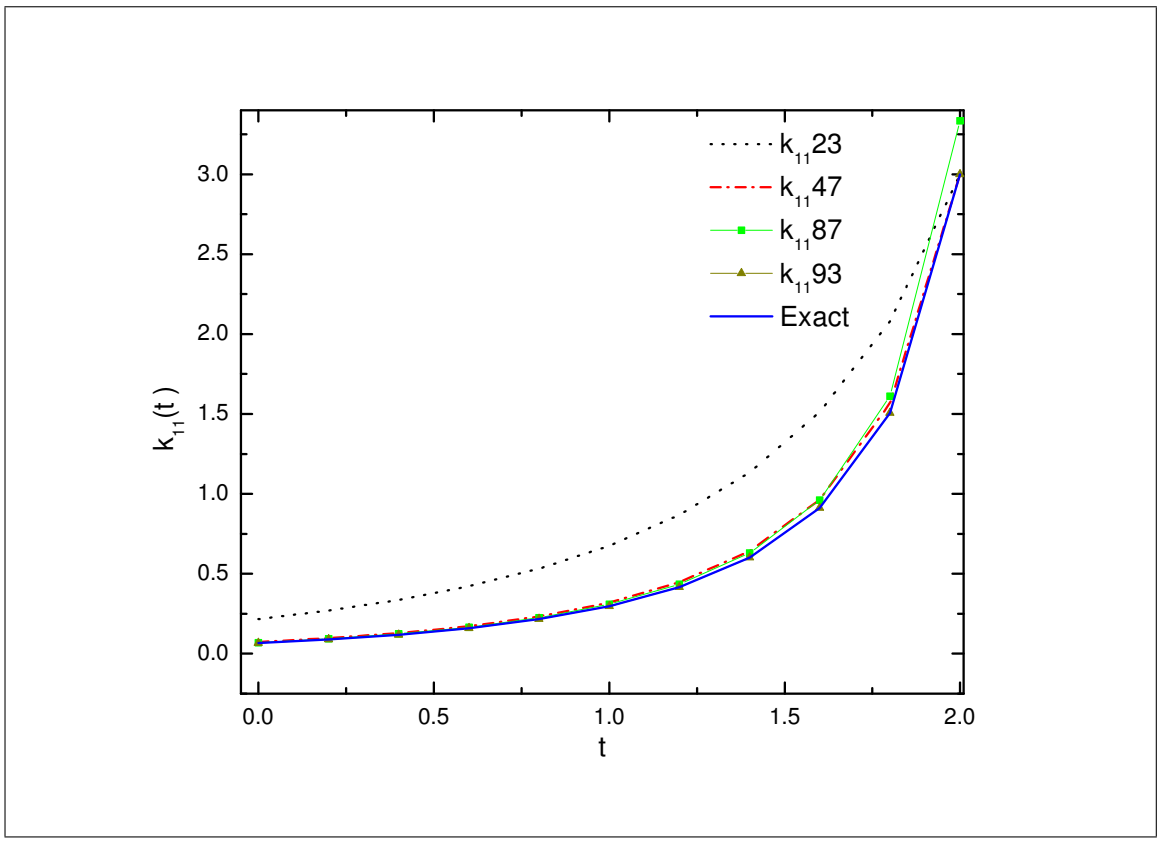

Figure 4.6: Candidate solutions for $k_{11}(t)$ by ACP for various generations and comparison with the exact solution.

4.2 Modified ant colony programming for solving MRDE with nonlinear singular fuzzy system: singular cost

Fuzzy modelling has proven its capability as a universal approximator for smooth nonlinear systems. The fuzzy controller which consists of several linear models in their own local dynamics in different states, allows the researcher to utilize a complex controller design within an intuitively straightforward framework. The total output for the nonlinear systems is obtained by utilizing a fuzzy "blending" of these linear models.

In this section, the singular nonlinear system is given as

$$E_i\dot{x} = A\left(x\right)x\left(t\right) + Bu\left(t\right),$$

where E_i is a singular matrix, $x\left(t\right) \in \mathbb{R}^n$ is a generalized state space vector and $u\left(t\right) \in \mathbb{R}^m$is a control variable, $A \in \mathbb{R}^{n\times n}$and $B \in \mathbb{R}^{n\times m}$ are the coefficient matrices that associated with $x\left(t\right)$ and $u\left(t\right)$, respectively. A is a vector valued function of x that is possibly nonlinear. x_0 is the given initial state vector and $m \leq n$. To derive the T-S fuzzy model

from the above equation, the first step is to determine the membership function. For example, the matrix $A(x)$ is taken as

$$A\left(x\right) \;=\; \left[\begin{array}{cc} 0 & 1 \\ x_1\left(t\right) & x_2\left(t\right) \end{array}\right],$$

Let $x_1 \in [0.5,\ 3.5]$ and $x_2 \in [-1,\ 4]$. The fuzzy variables x_1 and x_2 are also denoted as z_1 and z_2, respectively. The maximum and minimum values of z_1 and z_2 can be calculated, therefore x_1 and x_2 can be represented for the membership functions M_1, M_2, N_1 and N_2. From these membership functions, the nonlinear systems can be linearized into the i^{th} rule of continuous T-S fuzzy model of the following forms. Given the singular non-linear system that can be expressed in the form of T-S fuzzy system R^i: Model Rule i: If $z_1\left(t\right)$ is M_{i1} and $z_2(t)$ is M_{i2} ...and $z_p(t)$ is M_{ip}, then

$$E_i\dot{x}(t) = A_ix(t) + B_iu(t), \qquad x\left(0\right) = x_0,\ i = 1, 2, ..r. \tag{4.1}$$

Therefore the nonlinear system is modeled by the following fuzzy rules where the subsystems are defined as

$$A_1 = \left[\begin{array}{cc} 0 & 1 \\ max\left(z_1\left(t\right)\right) & max\left(z_2\left(t\right)\right) \end{array}\right], A_2 = \left[\begin{array}{cc} 0 & 1 \\ max\left(z_1\left(t\right)\right) & min\left(z_2\left(t\right)\right) \end{array}\right]$$

$$A_3 = \left[\begin{array}{cc} 0 & 1 \\ min\left(z_1\left(t\right)\right) & min\left(z_2\left(t\right)\right) \end{array}\right], A_4 = \left[\begin{array}{cc} 0 & 1 \\ min\left(z_1\left(t\right)\right) & max\left(z_2\left(t\right)\right) \end{array}\right].$$

To minimize both state and control signals of the feedback control system, a quadratic performance index is defined:

$$J \;=\; \frac{1}{2}x^T(t_f)E_i^TSE_ix(t_f) + \frac{1}{2}\int_0^{t_f}[x^T(t)Qx(t) + u^T(t)Ru(t)]dt,$$

subject to the linear singular fuzzy system (4.1). The superscript T represents the transpose operator. $S \in \mathbb{R}^{n\times n}$and $Q \in \mathbb{R}^{n\times n}$ are symmetric and positive definite weighting matrices for $x(t)$. $R \in \mathbb{R}^{m\times m}$ is a symmetric and positive definite weighting matrix for $u(t)$.

$$S = \left[\begin{array}{cc} 1 & 0 \\ 0 & 0 \end{array}\right], E_i = \left[\begin{array}{cc} 1 & 0 \\ 0 & 0 \end{array}\right], A_1 = \left[\begin{array}{cc} 0 & 1 \\ 3.5 & 4 \end{array}\right],$$

$$A_2 = \begin{bmatrix} 0 & 1 \\ 3.5 & -1 \end{bmatrix}, A_3 = \begin{bmatrix} 0 & 1 \\ 0.5 & 4 \end{bmatrix}, A_4 = \begin{bmatrix} 0 & 1 \\ 0.5 & -1 \end{bmatrix},$$

$$B_i = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, R = 0, Q = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}.$$

The matrices above are substituted into the following MRDE for the nonlinear singular fuzzy system below:

$$E_i^T \dot{K}_i(t) E_i + E_i^T K_i(t) A_i + A_i^T K_i(t) E_i + Q - E_i^T K_i(t) B_i R^{-1} B_i^T K_i(t) E_i = 0.$$

The numerical implementation can be adapted by taking $t_f = 2$ for solving the MRDE and these are later transformed into sets of differential equations in k_{11} and k_{12}, with their terminal conditions given as $k_{11}(2) = 1$ and $k_{12}(2) = 0$. Then, the optimal control can be obtained by the solution of MRDE.

In this study, the first generations with 50-100 ants are sent out to visit 8-12 nodes from any random initial node until the ants reach to the limit where the terminal condition is satisfied. These digital ants are sent out in order to find solutions for both k_{11} and k_{12}, simultaneously. If the value for the fitness function are close to zero, the global update of pheromone trail takes place in order to increase the pheromone value on the solution path. This significant piece of information will be used for the next generation and it will be repeated several times until the final solution is achieved.

At 27^{th} generation, the mechanism of the ACP approach predicted an expression with the fitness function equivalent to 0.0625. The tours is given as:

$$\text{Tours: } k_{11}(t) = e(t/2 - 1);$$

$$\text{Expressions: } k_{11}(t) = e^{\left(\frac{t}{2} - 1\right)}.$$

This piece of information will be added up into the global update of the pheromone values and the number of nodes will be increased for the next generation. The number of nodes will be increased only if the ants could not find any combination of expression which satisfies the fitness function in that current number of nodes. As the number of nodes is increased, the value of the fitness function can also be changed. This is vital in order to guide the ACP method to the final solution. At 44^{th} generation, with the fitness function (E_r) is 1, the ACP predicted an expression given as:

$$\text{Tours: } k_{11}(t) = e(7/4 * t - 7/2);$$

$$\text{Expressions: } k_{11}(t) = e^{\left(\frac{7}{4}t - \frac{7}{2}\right)}.$$

Again the global pheromone value will be updated and the next generation will be sent out to the space graph. This process will keep on repeating until it reaches up to 20 nodes, where at this stage, the modified ACP algorithm finds a combination of expressions which gives the fitness function equals to 0.5. This candidate solution is achieved at the 78^{th} generation and its given as:

$$\text{Tours: } k_{11}(t) = e(7/4 * t - 7/2) * 3/5 + 1/2;$$

$$\text{Expressions: } k_{11}(t) = \tfrac{3}{5}e^{\left(\frac{7}{4}t - \frac{7}{2}\right)} + \tfrac{1}{2}.$$

Finally, at 95^{th} generation, the modified ACP predicted the solution which satisfied the terminal condition as well as the fitness function where it is given in the functional form as:

$$\text{Tours: } k_{11}(t) = e(7/4 * t - 7/2) * 3/7 + 4/7;$$

$$\text{Expressions: } k_{11}(t) = \tfrac{3}{7}e^{\left(\frac{7}{4}t - \frac{7}{2}\right)} + \tfrac{4}{7}.$$

The modified ACP is also used to compute the solution for the k_{12} where the final solution is achieved when the ACP reached up to 18 nodes. The fitness function (E_r) is equal to zero and the terminal condition is satisfied. Below we listed down the trial solutions and the fitness functions obtained in order to find the solution for k_{12}:

Tours: $k_{12}(t) = e((t-2)/7)/9 \Longrightarrow 27^{th}$ generation, $E_r = 0.1837$;

Expressions: $k_{12}(t) = e^{\left(\frac{t}{2} - 1\right)}$.

Tours: $k_{12}(t) = e((t-2)/7) - 1 \Longrightarrow 35^{th}$ generation, $E_r = 0.0115$;

Expressions: $k_{12}(t) = e^{\left(\frac{t}{2} - 1\right)} - 1$.

Tours: $k_{12}(t) = e(7/4 * t - 7/2)/5 - 1/5 \Longrightarrow 69^{th}$ generation, $E_r = 0.01$;

Expressions: $k_{12}(t) = \frac{1}{5}e^{\left(\frac{7}{4}t - \frac{7}{2}\right)} - \frac{1}{5}$.

Tours: $k_{12}(t) = e(7/4 * t - 7/2)/7 - 1/7 \Longrightarrow 77^{th}$ generation, $E_r = 0.00$;

Expressions:$k_{12}(t) = \frac{1}{7}e^{\left(\frac{7}{4}t - \frac{7}{2}\right)} - \frac{1}{7}$.

The parse trees for both solutions are given in Figures 4.7 and 4.9, whereas the candidate solutions are shown in Figures 4.8 and 4.10, respectively. These solutions are compared with the exact solutions. The numerical solutions are shown in Table 4.3.

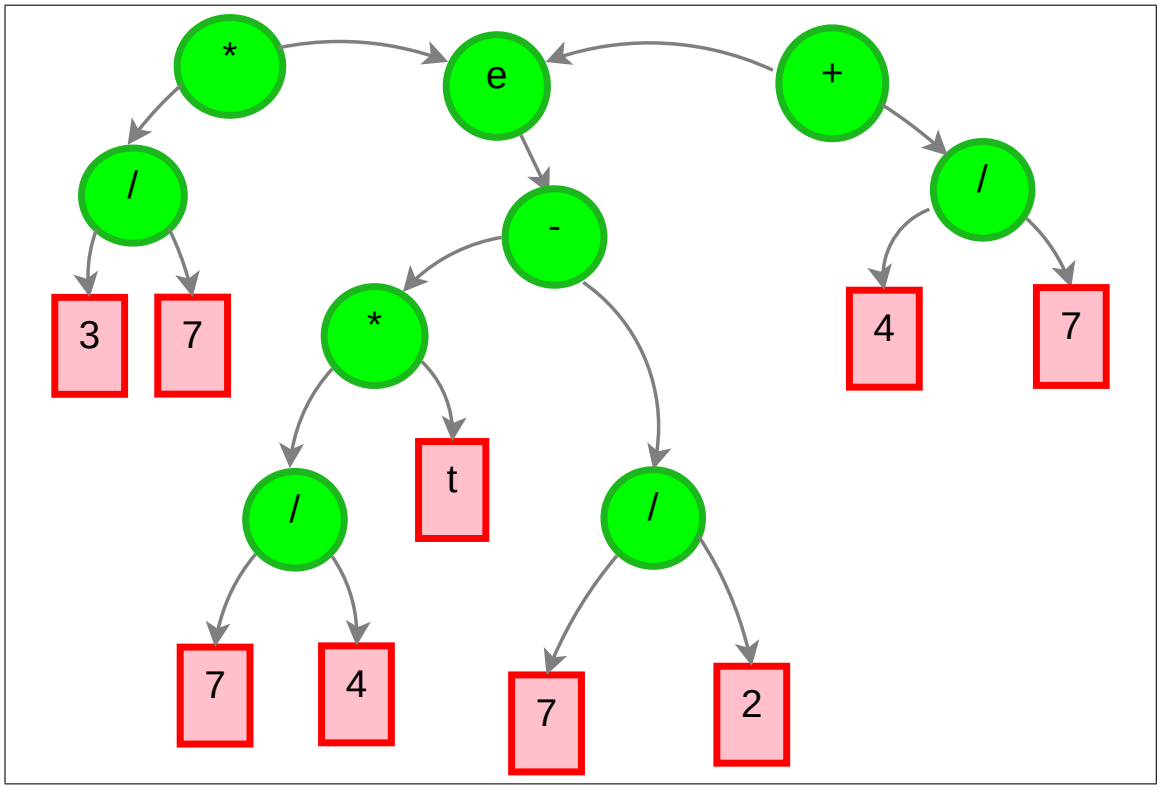

Figure 4.7: Parse tree for $k_{11}\,(t)$

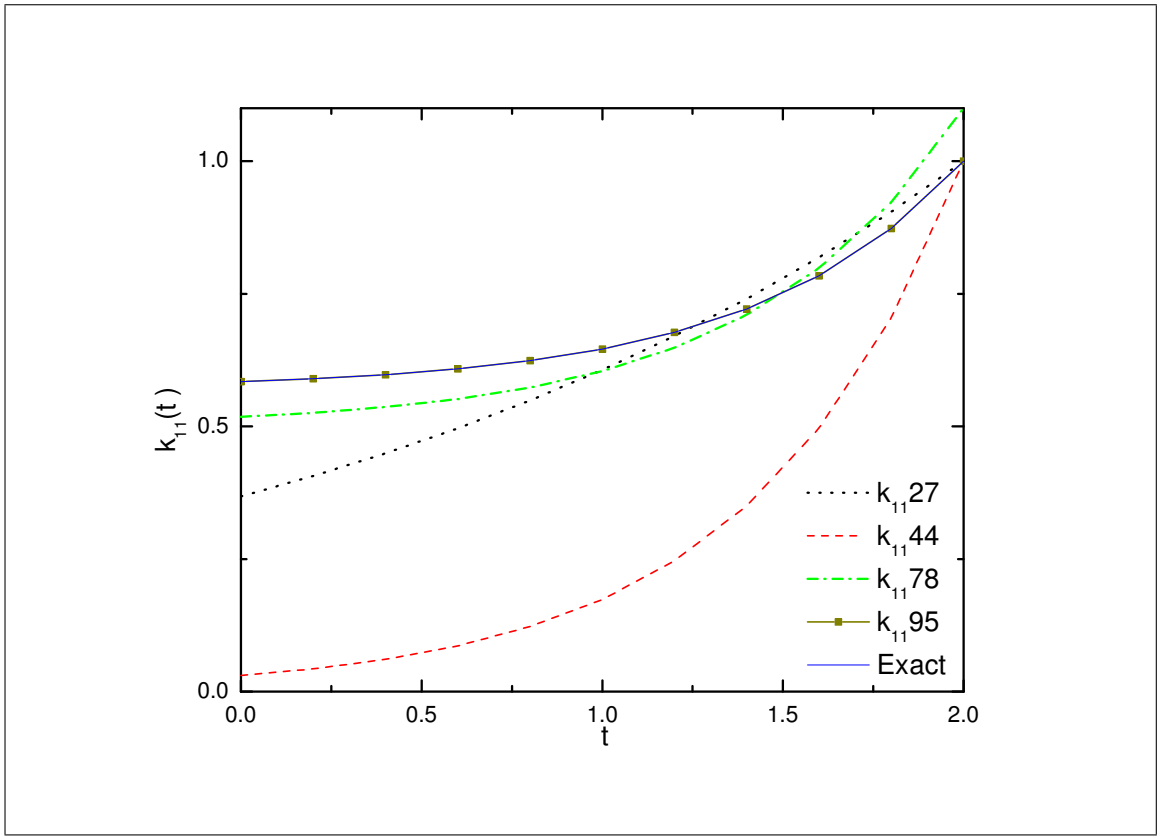

Figure 4.8: Candidate solutions according to generation for $k_{11}(t)$ by ACP and comparison with the exact solution.

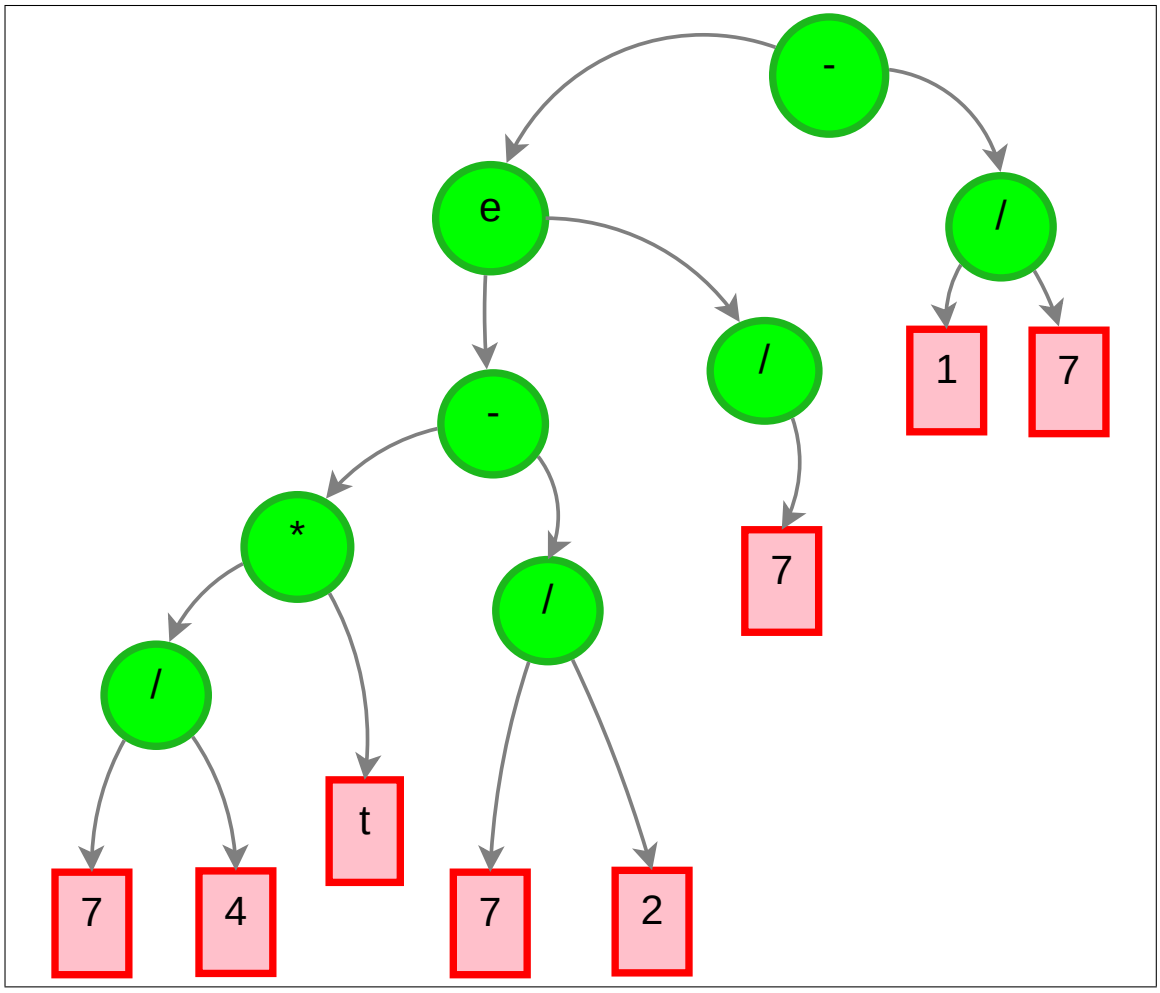

Figure 4.9: Parse tree for $k_{12}\,(t)$

Table 4.3: Numerical solutions for $k_{11}(t)$ and $k_{12}(t)$ when $R = 0$.

	ACP		RK4		Exact	
t	$k_{11}\,(t)$	$k_{12}\,(t)$	$k_{11}\,(t)$	$k_{12}\,(t)$	$k_{11}\,(t)$	$k_{12}\,(t)$
0.0	0.58437	-0.13854	0.58438	-0.13854	0.58437	-0.13854
0.2	0.58979	-0.13674	0.58980	-0.13673	0.58979	-0.13674
0.4	0.59749	-0.13417	0.59750	-0.13417	0.59749	-0.13417
0.6	0.60841	-0.13053	0.60843	-0.13052	0.60841	-0.13053
0.8	0.62391	-0.12536	0.62393	-0.12536	0.62391	-0.12536
1.0	0.64590	-0.11803	0.64593	-0.11803	0.64590	-0.11803
1.2	0.67711	-0.10763	0.67714	-0.10762	0.67711	-0.10763
1.4	0.72140	-0.09287	0.72143	-0.09286	0.72140	-0.09287
1.6	0.78425	-0.07192	0.78428	-0.07191	0.78425	-0.07192
1.8	0.87344	-0.04219	0.87346	-0.04218	0.87344	-0.04219
2.0	1.00000	0.00000	1.00000	0.00000	1.00000	0.00000

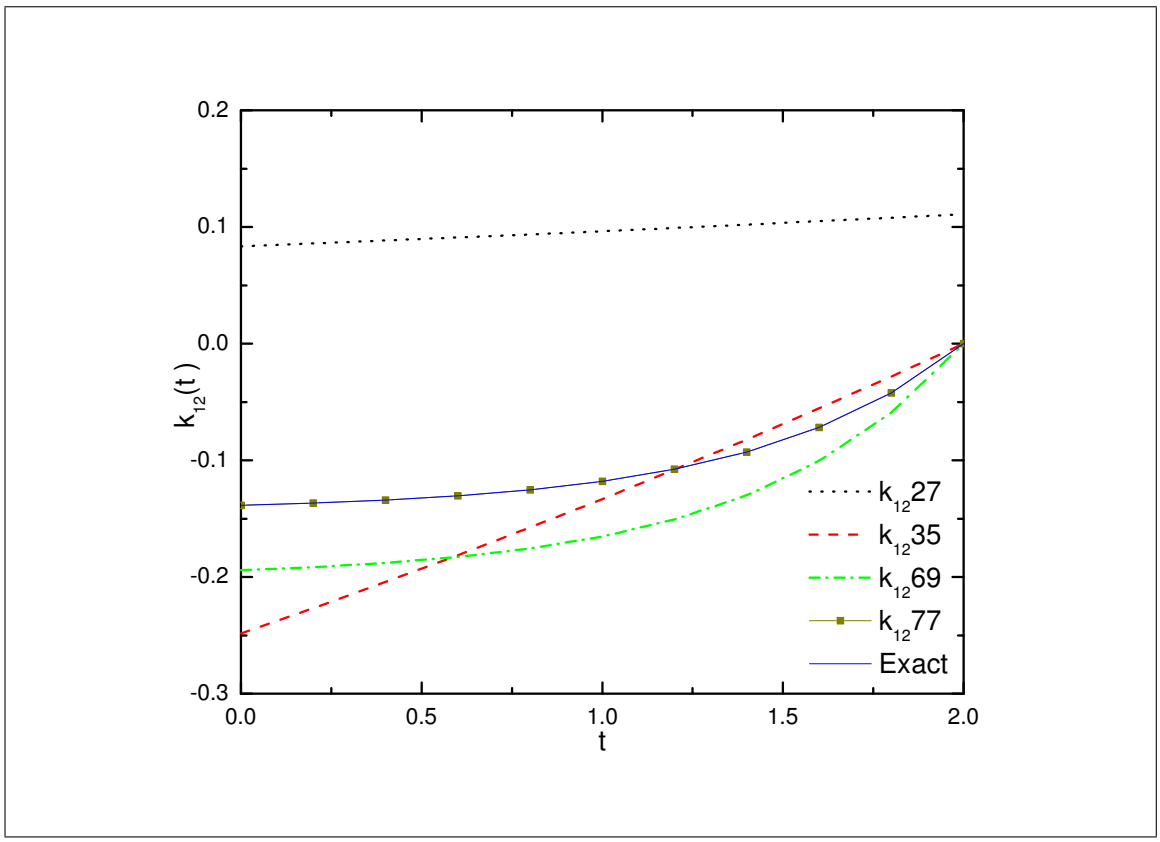

Figure 4.10: Candidate solutions according to generation for $k_{12}(t)$ by ACP and comparison with the exact solution.

4.2.1 Nonlinear singular fuzzy system with cross term

In this subsection, we are dealing with the optimal control problem where a quadratic performance index is required to be minimized in order to minimize both the state and the control signals of the feedback control system. The performance index is defined as

$$J = \frac{1}{2}\int_{t_0}^{t_f}[x^T(t)Qx(t) + u^T(t)Ru(t) + 2u^T(t)Hx(t)]dt,$$

where the superscript T represents the transpose operator, $S \in \mathbb{R}^{n\times n}$and $Q \in \mathbb{R}^{n\times n}$ are symmetric and positive definite weighting matrices for $x(t)$, and $R \in \mathbb{R}^{m\times m}$ is a symmetric and positive definite weighting matrix for $u(t)$. $H \in \mathbb{R}^{m\times n}$ is a coefficient matrix.

The quadratic performance index J is minimized subject to the linear singular fuzzy system $\mathbb{R}^i$: If $z_1(t)$ is M_{i1} and $z_2(t)$ is M_{i2} and....$z_p(t)$ is M_{ip}, then

$$E_i\dot{x}(t) = A_ix(t) + B_iu(t), \qquad x(0) = x_0,\ i = 1, 2, ..r,$$

where

$$S = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, E_i = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, A_1 = \begin{bmatrix} 0 & 1 \\ 2 & 1 \end{bmatrix},$$

$$A_2 = \begin{bmatrix} 0 & 1 \\ 2 & -1 \end{bmatrix}, A_3 = \begin{bmatrix} 0 & 1 \\ 0.1 & -1 \end{bmatrix}, A_4 = \begin{bmatrix} 0 & 1 \\ 0.1 & 1 \end{bmatrix},$$

$$B_i = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, R = 1, Q = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, H = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

The matrices above are substituted into the following MRDE for the nonlinear singular fuzzy system:

$$E_i^T \dot{K}_i(t) E_i + E_i^T K_i(t) A_i + A_i^T K_i(t) E_i + Q - (H^T + E_i^T K_i(t) B_i) R^{-1} (H + B_i^T K_i(t) E_i) = 0.$$

The numerical implementation can be adapted by taking $t_f = 2$ for solving the MRDE and these are later transformed into sets of differential equations in k_{11} and k_{12}. Then, the optimal control can be obtained by the solution of MRDE.

In this ACP approach, the construction graph has 18 nodes T={0, 1, 2, 3, 4, 5, 6, 7, 8, 9, t} and F={+, -, *, /, exp, (,)}. In the first generation, 50-100 ants are sent to visit 5-10 nodes from any initial nodes randomly until the ants reach to the limit where the terminal condition is satisfied. Although the value for the fitness function may not be close to zero, but the path or tour that have been taken by the ants, might lead to the final solution. Therefore after completion of each generation, a global update of pheromone trail is taken place in order to increase the pheromone value on the solution path. This significant piece of information will be used for the next generation. Thus from 5-10 nodes, the mechanism of the ACP will jump to 6-12 nodes and then if it still does not satisfy the initial conditions and the fitness function, the process will jump to 7-14 nodes and this process will be repeated several times until the final solution is obtained. Working on this MRDE problem, the expression is generated randomly up to 16 to 18 nodes, where $\rho = 0.5$, $\tau_{ij} = 0.2$ and $\beta = 1$. After the expression satisfies the terminal condition and the fitness function, then the solution is obtained.

Below is listed the trial solutions and the fitness functions obtained in order to find the solution for k_{11} and k_{12}.

k_{11}:

Tours: $k_{11}(t) = e(2-t) \Longrightarrow 13^{th}$ generation, $E_r = 16$;

Expressions: $k_{11}(t) = e^{(2-t)}$.

Tours: $k_{11}(t) = 2/(e(2-t)+1) \Longrightarrow 55^{th}$ generation, $E_r = 6.25$;

Expressions: $k_{11}(t) = \frac{2}{e^{2-t}+1}$.

Tours: $k_{11}(t) = 3/(5 * e(4-2*t)-2) \Longrightarrow 79^{th}$ generation, $E_r = 0.1111$;

Expressions: $k_{11}(t) = \frac{3}{5e^{(4-2t)}-2}$.

Tours: $k_{11}(t) = 2/(3 * e(4-2*t)-1) \Longrightarrow 105^{th}$ generation, $E_r = 0.00$;

Expressions:$k_{11}(t) = \frac{2}{3e^{(4-2t)}-1}$.

k_{12}:

Tours: $k_{12}(t) = (8-e(t))/7 \Longrightarrow 37^{th}$ generation, $E_r = 3.781$;

Expressions: $k_{12}(t) = \frac{8-e^t}{7}$.

Tours: $k_{12}(t) = 3/(e(t)+2) \Longrightarrow 75^{th}$ generation, $E_r = 7.554$;

Expressions: $k_{12}(t) = \frac{3}{e^t+2}$.

Tours: $k_{12}(t) = 1 - 3/(5 * e(4-2*t)-2) \Longrightarrow 97^{th}$ generation, $E_r = 0.1111$;

Expressions: $k_{12}(t) = 1 - \frac{3}{5e^{(4-2t)}-2}$.

Tours: $k_{12}(t) = 1 - 2/(3 * e(4-2*t)-1) \Longrightarrow 157^{th}$ generation, $E_r = 0.00$;

Expressions:$k_{12}(t) = 1 - \frac{2}{3e^{(4-2t)}-1}$.

These candidate solutions are depicted in the Figures 4.11 and 4.12 for $k_{11}(t)$ and $k_{12}(t)$, respectively. The parse trees for the solutions $k_{11}(t)$ and $k_{12}(t)$ are shown in Figures 4.13 and 4.14, respectively. The numerical solutions are given in Table 4.4. In Table 4.5, the average number of generations and ants, together with the computational time, are shown. In comparing the ACP and the genetic programming (GP) method (Tsoulos & Lagaris, 2006), in terms of the average number of generations (AVG), we found that the ACP method provide faster solutions compared to the GP method. Similarly, the MRDE can be solved for the matrices A_2, A_3 and A_4.

4.3 Takagi-Sugeno fuzzy modelling for solving the Human Immunodeficiency Virus immunology model using modified ACP

The equations for the HIV immunology model used in this paper is extracted from the paper by Kirschner and Webb (Kirschner & Webb, 1998). The equations are given as

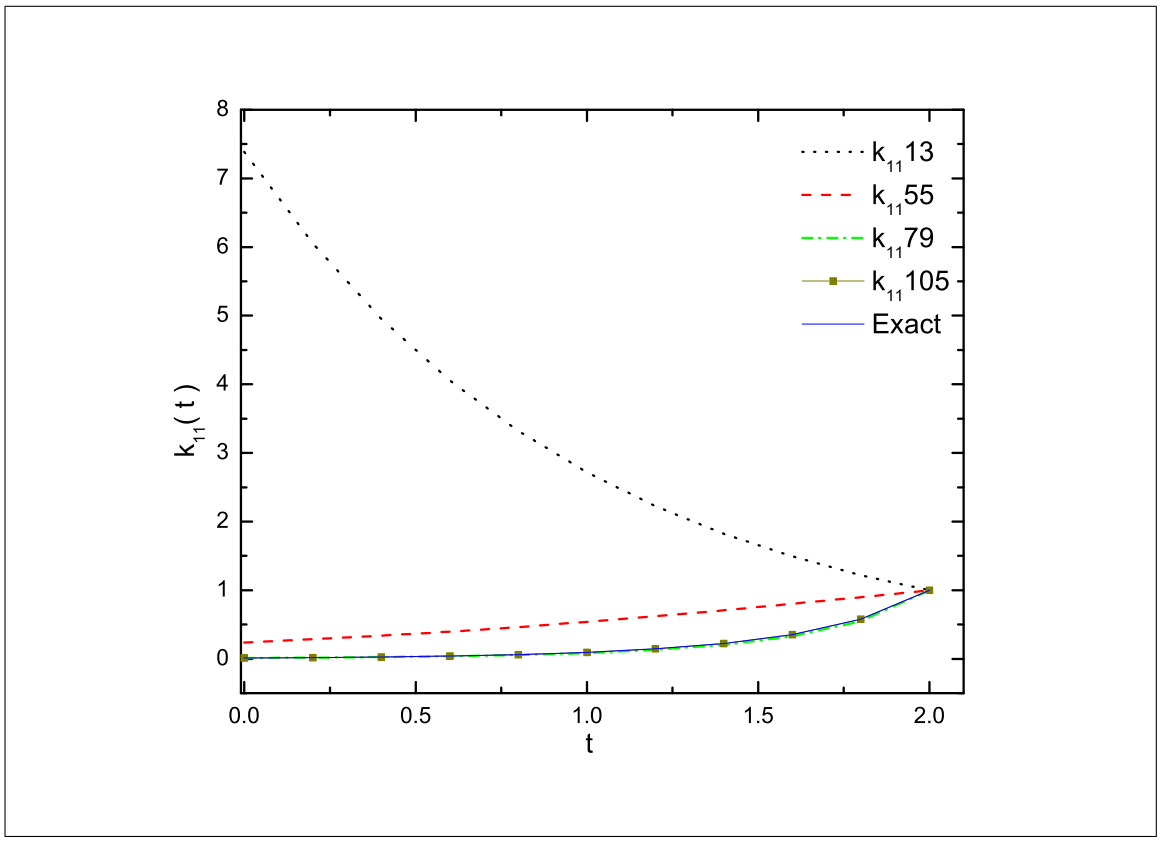

Figure 4.11: Candidate solutions for $k_{11}(t)$ by ACP for various generations and comparison with the exact solution.

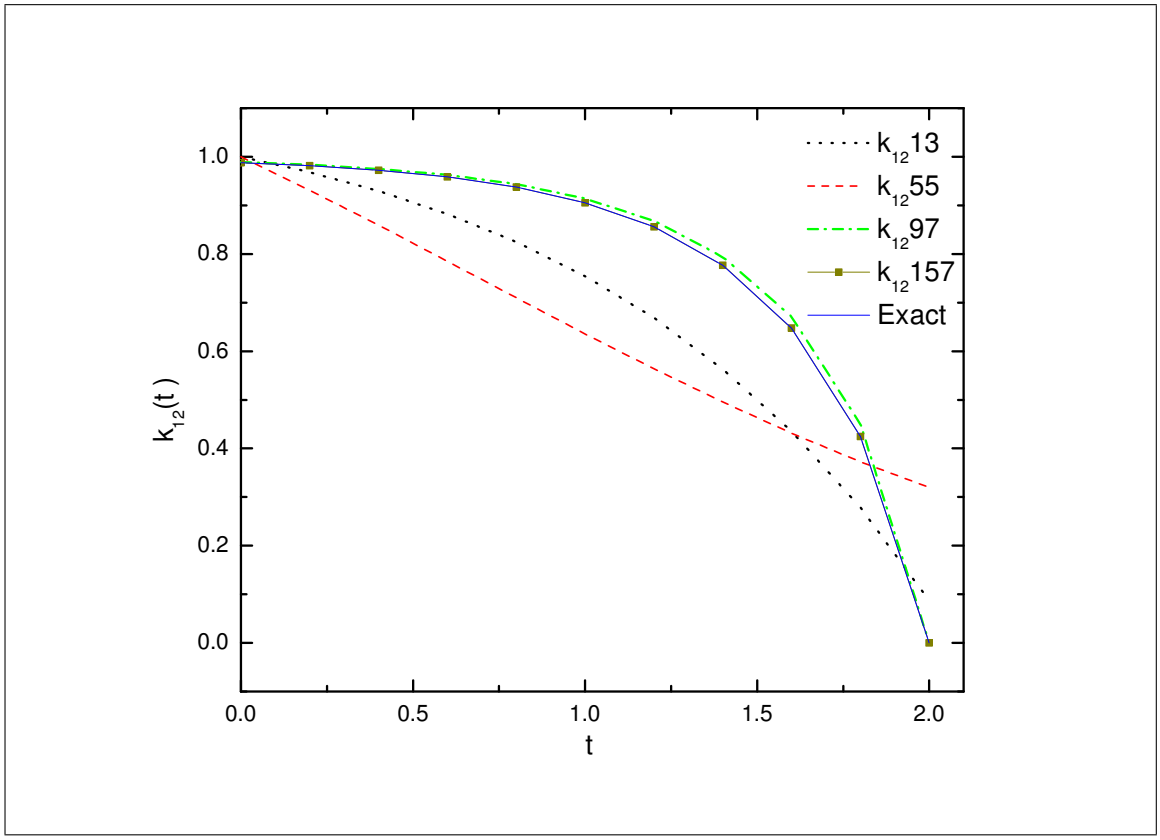

Figure 4.12: Candidate solutions according to generations for $k_{12}(t)$ by ACP and comparison with the exact solution.

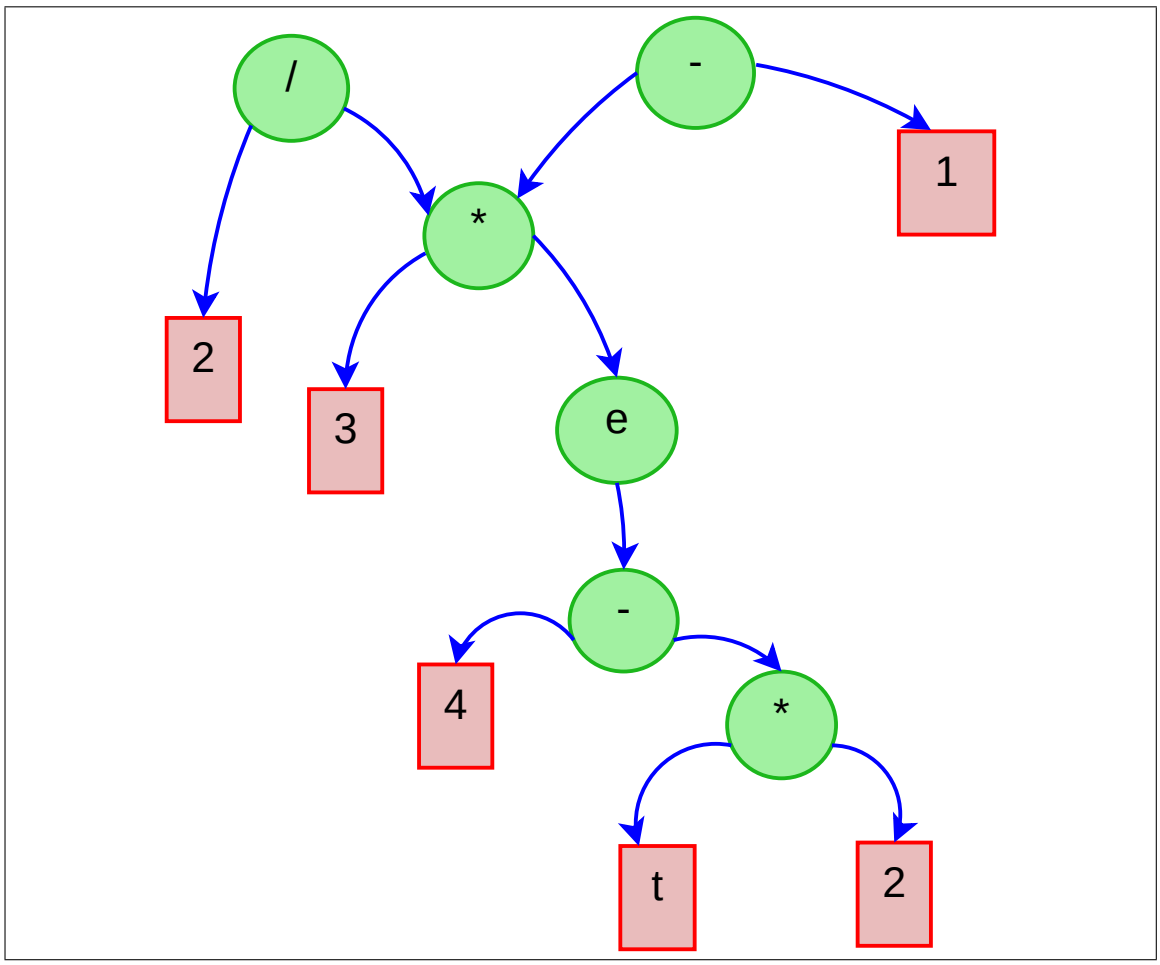

Figure 4.13: Parse tree for $k_{11}(t)$

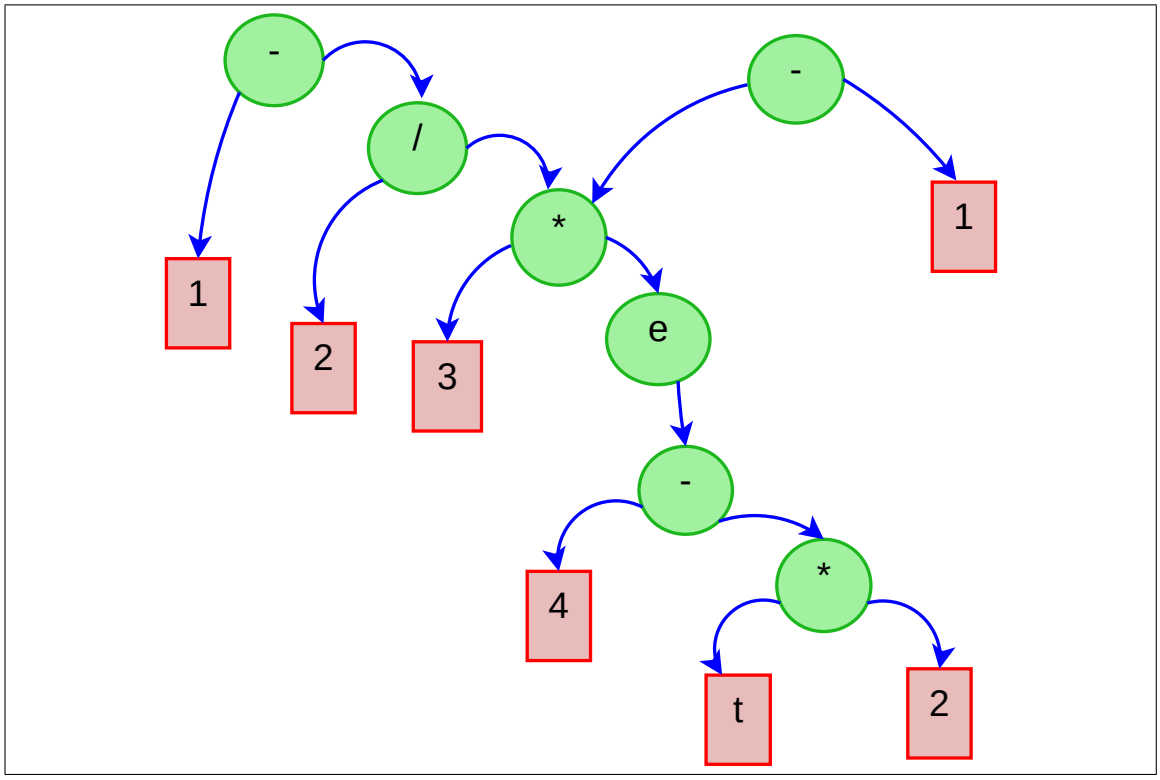

Figure 4.14: Parse tree for $k_{12}(t)$

Table 4.4: Results obtained by ACP, RK4-method and the exact solutions.

	ACP		RK4		Exact	
t	$k_{11}(t)$	$k_{12}(t)$	$k_{11}(t)$	$k_{12}(t)$	$k_{11}(t)$	$k_{12}(t)$
0.0	0.01229	0.98772	0.01231	0.98772	0.01229	0.98772
0.2	0.01838	0.98162	0.01842	0.98386	0.01838	0.98162
0.4	0.02755	0.97245	0.02760	0.97808	0.02755	0.97245
0.6	0.04138	0.95862	0.04145	0.96936	0.04138	0.95862
0.8	0.06236	0.93764	0.06247	0.95613	0.06236	0.93764
1.0	0.09449	0.90551	0.09463	0.93593	0.09449	0.90551
1.2	0.14431	0.85569	0.14452	0.85548	0.14431	0.85569
1.4	0.22321	0.77679	0.22349	0.77651	0.22321	0.77679
1.6	0.35232	0.64768	0.35269	0.64730	0.35232	0.64768
1.8	0.57546	0.42454	0.57588	0.42412	0.57546	0.42454
2.0	1.00000	0.00000	1.00000	0.0000	1.00000	0.00000

Table 4.5: Comparison results for $k_{11}(t)$ and $k_{12}(t)$ between ACP and GP.

$k_{11}(t)$	Average No.Generations	Average No. Ants	Average Time(secs)
ACP	98	80	120.53
GP	441	-	-
$k_{12}(t)$	Average No.Generations	Average No. Ants	Average Time(secs)
ACP	105	95	135.77
GP	451	-	-

follows:

$$\frac{dT(t)}{dt} = s_1 - \frac{s_2 V(t)}{b_1 + V(t)} - \mu T(t) - kV(t)T(t), \qquad T(0) = T_0 \tag{4.2}$$

$$\frac{dV(t)}{dt} = \frac{gV(t)}{b_2 + V(t)} - cV(t)T(t), \qquad V(0) = V_0. \tag{4.3}$$

The parameters of HIV model are given in Table 4.6. Substituting these parameters in (4.2) and (4.3), the HIV immunology equations can be written as

$$\frac{dT(t)}{dt} = 2 - \frac{V(t)}{2 + V(t)} - 0.1T(t) - 0.1V(t)T(t)$$

$$\frac{dV(t)}{dt} = \frac{2V(t)}{1 + V(t)} - 0.1V(t)T(t).$$

Table 4.6: The parameters and their units used in the HIV immunology model .

Parameters	Definition
T	Uninfected $CD4^+$ T cell population
V	HIV population
$s_1 = 2.0mm^3d^{-1}$	source of $CD4^+$ T cells
$s_2 = 1.0mm^3d^{-1}$	source of HIV cells
$\mu = 0.1d^{-1}$	death rate of uninfected $CD4^+$ T cell
$k = 0.1mm^3d^{-1}$	rate $CD4^+$ cells which get infected by the virus V
$g = 2d^{-1}mm^3$	input rate of external viral source
$c = 0.1mm^3d^{-1}$	lost rate of virus
$b_1 = 2.0mm^3$	half saturation constant
$b_2 = 1.0mm^3$	half saturation contant

Now from this given nonlinear systems, the main task is to derive the Takagi-Sugeno fuzzy model. The nonlinear systems can be written in matrices form as

$$\dot{T} = \begin{bmatrix} -0.1 & z_1(t) \\ 0 & z_2(t) \end{bmatrix} T(t)$$

where

$$z_1(t) = f(T,V) = \frac{4+V}{2V+V^2} - 0.1T$$

$$z_2(t) = g(T,V) = \frac{2}{1.0+V} - 0.1T.$$

The next step is to obtain the membership functions. In order to do it, assume that $T(t) \in [0.1, 1]$ and $V(t) \in [0.1, 1]$, then calculate the min and max values for $z_1(t)$ and $z_2(t)$.

$$max\ z_1(t) = 19.5138,\ min\ z_1(t) = 1.5667$$

$$max\ z_2(t) = 1.8082,\ min\ z_2(t) = 0.9.$$

From these max and min values, the $z_1(t)$ and $z_2(t)$ can be represented as

$$z_1(t) = \frac{4+V}{2V+V^2} - 0.1T = M_1(z_1(t)).max\ z_1(t) + M_2(z_1(t)).min\ z_1(t)$$

$$z_2(t) = \frac{2}{1.0+V} - 0.1T = N_1(z_2(t)).max\ z_2(t) + N_2(z_2(t)).min\ z_2(t).$$

Thus, from this membership function, the nonlinear systems can be linearized into these

fuzzy differential equations according to the following continuous T-S fuzzy rules with $T(0) = 1$ and $V(0) = 1$:

- Model Rule 1: If $z_1(t)$ is Positive and $z_2(t)$ is Big, Then $\dot{T} = A_1 T$.
- Model Rule 2: If $z_1(t)$ is Positive and $z_2(t)$ is Small, Then $\dot{T} = A_2 T$.
- Model Rule 3: If $z_1(t)$ is Negative and $z_2(t)$ is Small, Then $\dot{T} = A_3 T$.
- Model Rule 4: If $z_1(t)$ is Negative and $z_2(t)$ is Big, Then $\dot{T} = A_4 T$,

where the subsystems are defined as

$$A_1 = \begin{bmatrix} -0.1 & max\left(z_1\left(t\right)\right) \\ 0 & max\left(z_2\left(t\right)\right) \end{bmatrix}, A_2 = \begin{bmatrix} -0.1 & max\left(z_1\left(t\right)\right) \\ 0 & min\left(z_2\left(t\right)\right) \end{bmatrix}$$

$$A_3 = \begin{bmatrix} -0.1 & min\left(z_1\left(t\right)\right) \\ 0 & min\left(z_2\left(t\right)\right) \end{bmatrix}, A_4 = \begin{bmatrix} -0.1 & min\left(z_1\left(t\right)\right) \\ 0 & max\left(z_2\left(t\right)\right) \end{bmatrix}$$

or

$$A_1 = \begin{bmatrix} -0.1 & 19.5138 \\ 0 & 1.8082 \end{bmatrix}, A_2 = \begin{bmatrix} -0.1 & 19.5138 \\ 0 & 0.9 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} -0.1 & 1.5667 \\ 0 & 0.9 \end{bmatrix}, A_4 = \begin{bmatrix} -0.1 & 1.5667 \\ 0 & 1.8082 \end{bmatrix}.$$

Consider the Model Rule 1: If $z_1(t)$ is Positive and $z_2(t)$ is Big, then

$$\begin{pmatrix} \dot{T} \\ \dot{V} \end{pmatrix} = \begin{bmatrix} -0.1 & 19.5138 \\ 0 & 1.8082 \end{bmatrix} \begin{pmatrix} T \\ V \end{pmatrix}$$

and the initial values is given as $T(0) = 1$ and $V(0) = 1$.

The ACP is implemented to solve the above equation. By generating the graph randomly, 80 generations, with 50-100 number of ants each, are sent out through the graph with $\rho = 0.5$ and $\beta = 1$. These digital ants are sent to find solutions for $T(t)$ and $V(t)$, simultaneously. These trial solutions are compared with the exact solutions given as:

$$T\left(t\right) = \frac{9756905}{954091} e^{\frac{904091}{500000}t} - \frac{8802814}{954091} e^{\frac{-1}{10}t} ;$$

$$V\left(t\right) = e^{\frac{904091}{500000}t} .$$

At generation 20, with fitness value equal to 0.3437, for $t = 0$, the intermediate solution was:

$$\text{Tours} : T(t) = 5 * 2 * e(2 * t) - 9;$$

$$\text{Expressions} : T(t) = 10e^{2t} - 9.$$

Next, at the 35^{th} generation, the fitness value was less than 0.1492. This actually predicted the corresponding candidate solution which was

$$\text{Tours} : T(t) = 5 * 2 * e(9/5 * t) - 9 * e(0 - 1/5 * t);$$

$$\text{Expressions} : T(t) = 10e^{\frac{9}{5}t} - 9e^{\frac{-1}{5}t}.$$

Later at the 39^{th} generation, the ACP generated an expression with its fitness value 0.0075. The functional form was given as

$$\text{Tours} : T(t) = 5 * 2 * e(9/5 * t) - 9 * e(0 - 1/6 * t);$$

$$\text{Expressions} : T(t) = 10e^{\frac{9}{5}t}\text{-}9e^{\frac{-1}{6}t}.$$

Finally, at the 67^{th}generation, the ACP computed the solution. This time the fitness function equals 0.0025. The final solution was given as

$$\text{Tours :}$$
$$T(t) = (sqr(5)*2+1)/5*e((sqr(6*5)+4*t)/(sqrt(5*2)*5))-(5*9+1)/5*e(0-t/9);$$

$$\text{Expressions} : T(t) = \frac{51}{5}e^{\frac{904}{500}t}\text{-}\frac{46}{5}e^{\frac{-t}{9}}.$$

Similarly, the ACP method was applied to obtain the solution for V. At generation 15, the fitness value equalled 0.0368 and the intermediate solution was

$$\text{Tours} : V(t) = e(2 * t);$$

$$\text{Expressions} : V(t) = e^{2t}.$$

Then another candidate solution was predicted at 23^{rd} generation with

$$\text{Tours} : V(t) = e(9/5 * t);$$

$$\text{Expressions} : V(t) = e^{\frac{9}{5}t}.$$

Table 4.7: Results comparison between ACP and exact solutions.

	ACP		Exact	
t	T	V	T	V
0	1.0000	1.000	1.000	1.000
0.1	3.1230	1.1982	3.1186	1.1982
0.2	5.6456	1.4356	5.6381	1.4357
0.3	8.6470	1.7201	8.6379	1.7202
0.4	12.2223	2.0610	12.2137	2.0612
0.5	16.4857	2.4695	16.4796	2.4697
0.6	21.5736	2.9588	21.5726	2.9592
0.7	27.6497	3.5452	27.6568	3.5457
0.8	34.9099	4.2478	34.9289	4.2484
0.9	43.5895	5.0896	43.6245	5.0904
1.0	53.9695	6.0982	54.0259	6.0993

Table 4.8: Comparison results for $T\,(t)$ and $V\,(t)$ between the ACP and GP.

$T\,(t)$	Average No.Generations	Average No. Ants	Average Time(secs)
ACP	146	95	150.53
GP	234	-	-
$V\,(t)$	Average No.Generations	Average No. Ants	Average Time(secs)
ACP	95	88	95.77
GP	234	-	-

where the fitness value is less than $7e^{-05}$. The final solution is only achieved when the ACP reached at the 44^{th}generation with its functional form given as:

$$\text{Tours} : V(t) = e((sqr(6*5) + 4\text{*}t)/(sqrt(5*2)*5));$$

$$\text{Expressions} : V(t) = e^{\frac{904}{500}t}.$$

The fitness value is given as $4e^{-08}$. In Figures 4.15 and 4.16, the evolution of a trial solutions for the above problem are shown. We also included an inset in each graph in order to show how close are the final solutions predicted by the ACP when compared to the analytical functions as the parameter time gets larger. The parse trees for the solutions T and V are shown in Figures 4.17 and 4.18. The numerical solutions are given in Table 4.7 whereas the average number of generations and ants together with the computational time are depicted in Table 4.8. In the systems of ODEs, solutions obtained are more than one, therefore the task for searching the solution using this non-traditional method will be very difficult. From Table 4.8, the ACP method succeeded in finding the solutions within the range of 90-150 average no. of generations but the GP method requires about 234. Similarly the solution for the HIV for fuzzy model rules 2, 3 and 4 can be obtained by using the modified ACP.

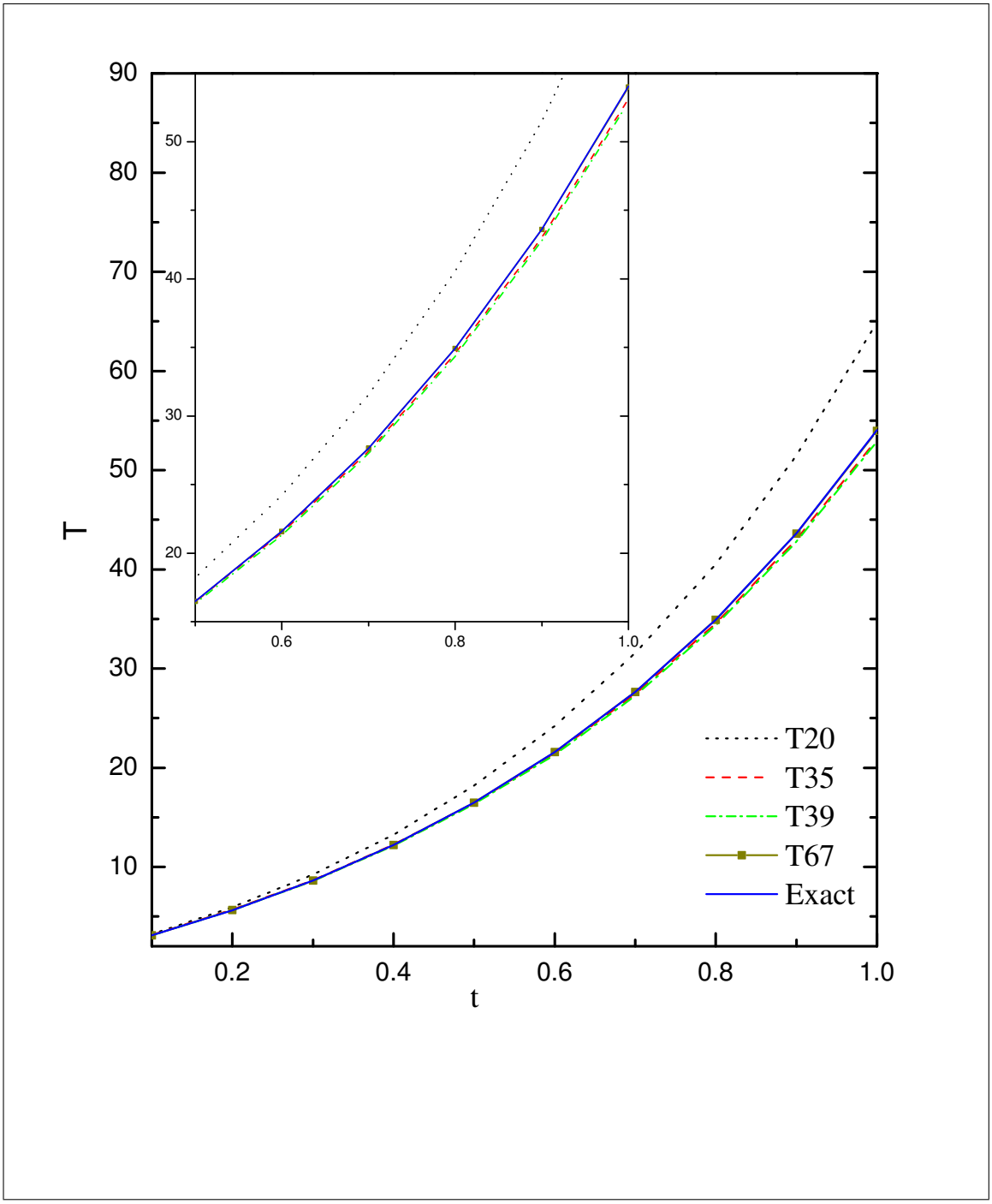

Figure 4.15: Candidate solutions for $T(t)$

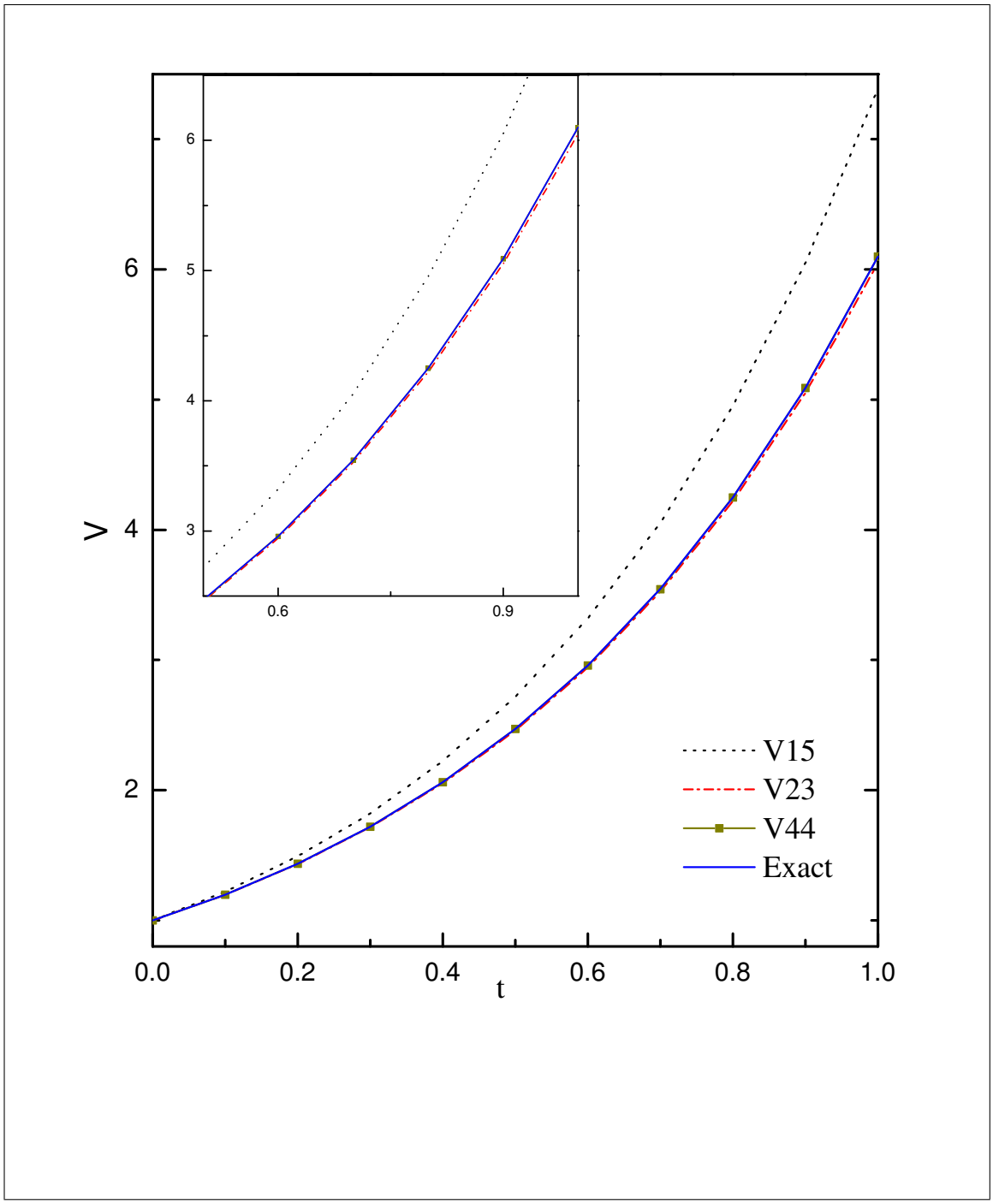

Figure 4.16: Candidate solutions for $V(t)$

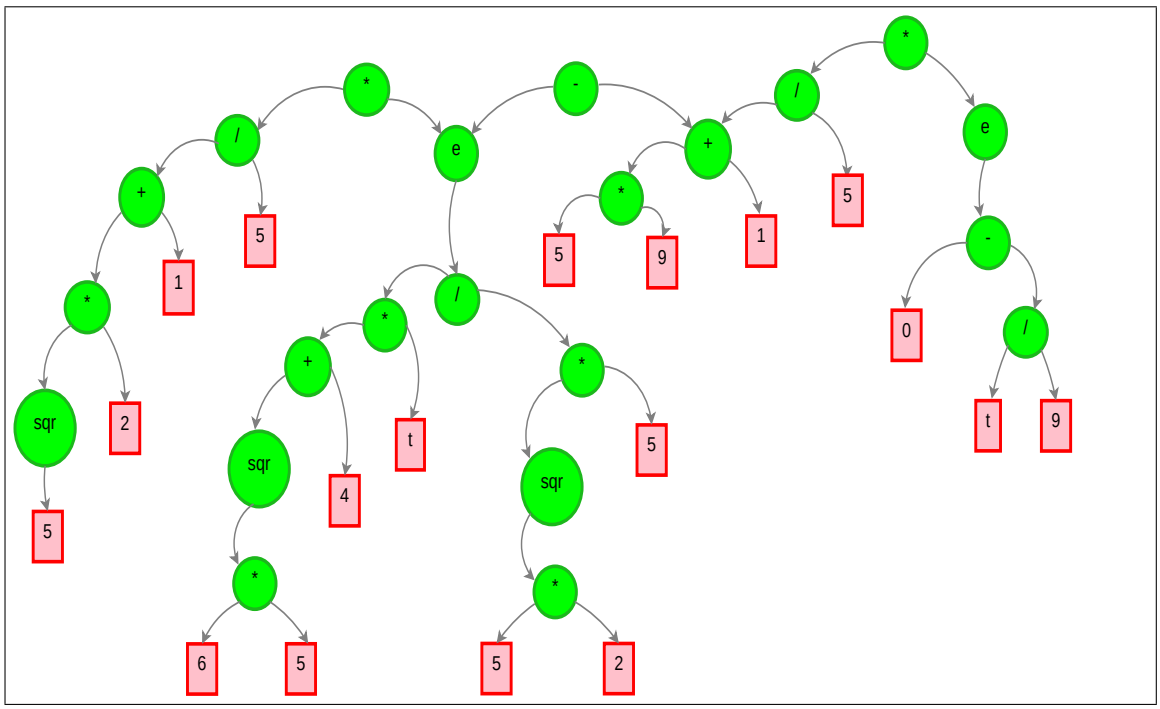

Figure 4.17: Tours of ant and its parse tree for T(t).

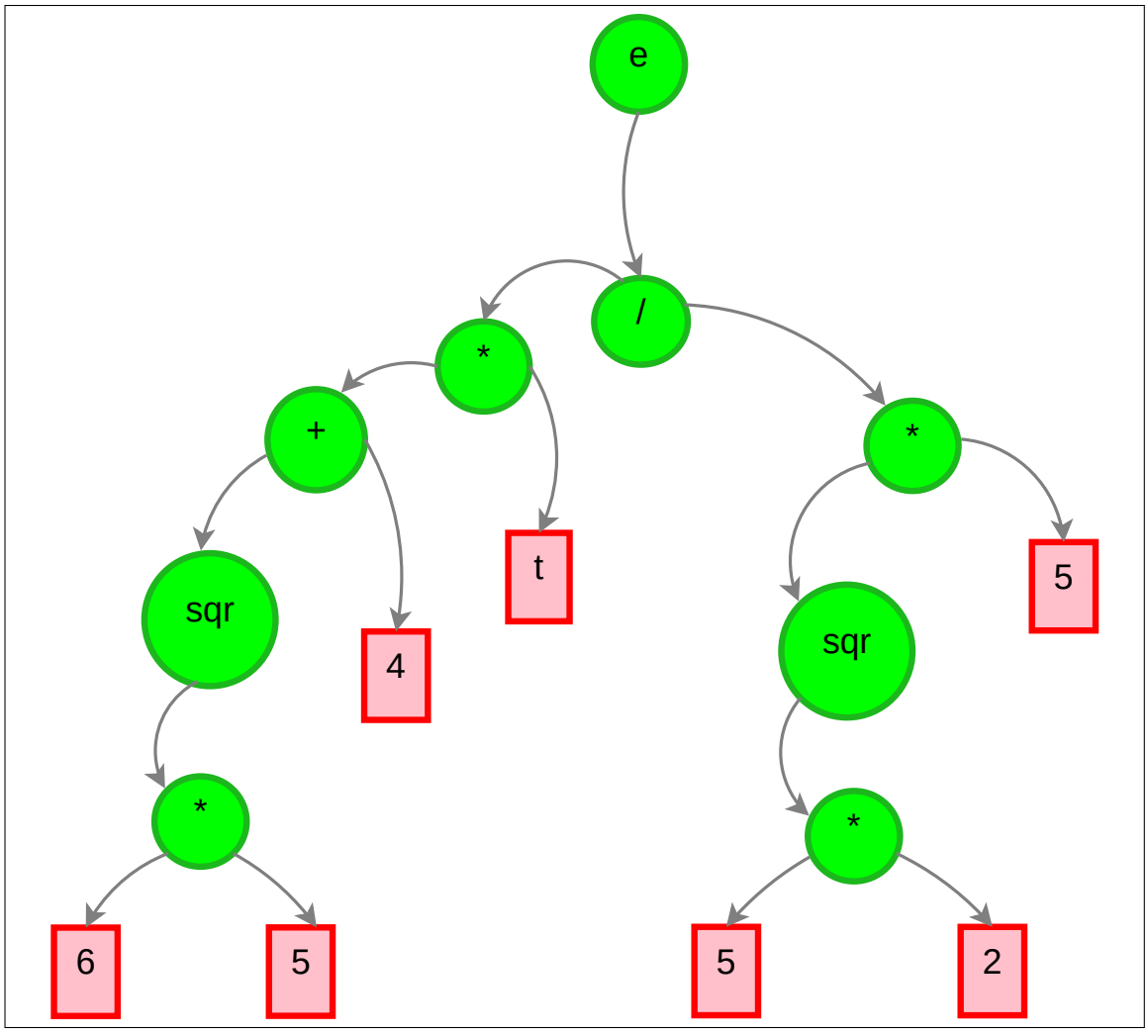

Figure 4.18: Tours of ant and its parse tree for V(t).

4.4 Takagi-Sugeno fuzzy modelling of S-type microbial growth model for ethanol fermentation process and optimal control

The development of mathematical models in the field of predictive microbiology to describe and predict the microbial evolution in foods are very vital. Single species microbial growth, whether in a (liquid) food product, normally passes three distinct phases. In the first phase called lag phase, the microbial cells adapt to their new environment and do not multiply. The total number of microbial cells remains constant during this phase. During the next phase or the exponential growth phase, the microbial cells multiply exponentially. Finally, the microbial cells cease multiplying and their total number remains constant at the maximum population density. This third final phase is called the stationary phase.

The S-type microbial growth model below was given in the paper by Van Impe et al., (Van Impe et al., 2006):

$$\dot{N}(t) = \left(\frac{Q(t)}{1+Q(t)}\right).\mu_{max}.S(t).N(t) \quad \text{with } N(t=0) = N_0,$$

$$\dot{Q}(t) = \mu_{max}.Q(t) \quad \text{with } Q(t=0) = Q_0,$$

$$\dot{S}(t) = -\left(\frac{Q(t)}{1+Q(t)}\right).\mu_{max}.\frac{S(t)}{Y_{\frac{N}{S}}}.N(t) \quad \text{with } S(t=0) = N_0.$$

The first differential equation describes the evolution of the microbial load in time. It consists of the adjustment function which describes the lag phase by means of a variable representing the physiological state of the cells $Q(t)$, as well as the inhibition function which is a linear function of the substrate concentration $S(t)$. The second differential equation, describes the evolution of $Q(t)$, which increases exponentially, whereas the third differential equation represents the evolution of the substrate concentration $S(t)$. $Y_{\frac{N}{S}}$ refers as the yield constant.

The T-S fuzzy model can be derived from the above nonlinear system using sector nonlinearity approach (Kawamoto et al., 1993). Consider the linear dynamical fuzzy system (Wu et al., 2005) that can be expressed in the form: R^i: If x_j is M_{ji}, $i = 1, 2, 3, 4$ and $j = 1, 2, 3$, then

$$\dot{x}(t) = A_i x(t) + B_i u(t), \qquad x(0) = 0,\ t \in [0,\ t_f], \tag{4.4}$$

where

$$\dot{x}(t) = \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \end{bmatrix} = \begin{bmatrix} \dot{N}(t) \\ \dot{Q}(t) \\ \dot{S}(t) \end{bmatrix},$$

$$A_i = \begin{bmatrix} z_1 & 0 & 0 \\ 0 & \mu_{max} & 0 \\ 0 & 0 & z_2 \end{bmatrix}, \; B_i = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix},$$

$$z_1 = \mu_{max} \frac{Q(t)}{1+Q(t)} S(t), \qquad z_2 = \; -\mu_{max} \tfrac{Q(t)}{1+Q(t)} \tfrac{N(t)}{Y_{\frac{n}{s}}} \quad ,$$

R^i denotes the i^{th} rule of the fuzzy model, M_{ji} is membership function, $x(t) \in \mathbb{R}^n$ is a generalized state space vector, $u(t) \in \mathbb{R}^m$ is a control variable and it takes value in some Euclidean space, and $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ are coefficient matrices associated with $x(t)$ and $u(t)$ respectively, x_0 is given initial state vector and $m \leq n$. Consider the system of differential equation for given below

$$\dot{K}_i(t) + K_i(t) A_i + A_i^T K_i(t) + Q - K_i(t) B_i R^{-1} B_i^T K_i(t) \;=\; 0 \qquad (4.5)$$

in each rule of the fuzzy model

$$\dot{k}_{ij}(t) = \phi_{ij}(k_{ij}(t)), \qquad (k_{ij})(t_f) = A_{ij}, \; (i, \, j \;=\; 1, 2, ..n).$$

Consider the optimal control problem:

$$J \;=\; E\left\{ \frac{1}{2} x^T(t_f) F_i^T S F_i x(t_f) + \frac{1}{2} \int_0^{t_f} \left[x^T(t) Q x(t) + u^T(t) R u(t) \right] dt \right\},$$

subject to the linear T-S fuzzy system R^i: If x_j is M_{ji} , $i = 1, 2, 3, 4$ and $j = 1, 2, 3$, then following the eq.(4.4). The values of x_1, x_2 and x_3 are taken as $x_1 \in [0.0, \, 0.5]$, $x_2 \in [0.0, \, 0.5]$ and $x_3 \in [0.0, \, 0.5]$. The value of μ_{max} is given as 9.006. The minimum and maximum values of z_1 and z_2 are calculated as follows:

$$max \; z_1(t) = 0.3333, min \; z_1(t) = 0.0,$$

$$max \; z_2(t) = 0.0, min \; z_2(t) = -0.16667.$$

Table 4.9: Results comparison between the ACP and exact solutions.

t	ACP k_{11}	Exact k_{11}
1.0	3.3693	3.3692
1.1	3.0553	3.0552
1.2	2.7615	2.7614
1.3	2.4867	2.4866
1.4	2.2296	2.2295
1.5	1.9890	1.9889
1.6	1.7640	1.7639
1.7	1.5535	1.5535
1.8	1.3566	1.3566
1.9	1.1723	1.1723
2.0	1.0000	1.0000

Therefore the membership functions can be computed

Model Rule 1: If $z_1(t)$ is Positive and $z_2(t)$ is Big, Then $\dot{x}(t) = A_1 x(t) + Bu$.

Model Rule 2: If $z_1(t)$ is Positive and $z_2(t)$ is Small, Then $\dot{x}(t) = A_2 x(t) + Bu$.

Model Rule 3: If $z_1(t)$ is Negative and $z_2(t)$ is Small, Then $\dot{x}(t) = A_3 x(t) + Bu$.

Model Rule 4: If $z_1(t)$ is Negative and $z_2(t)$ is Big, Then $\dot{x}(t) = A_4 x(t) + Bu$.

Here

$$S = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, A_1 = \begin{bmatrix} 0.3333 & 0 & 0 \\ 0 & 9.006 & 0 \\ 0 & 0 & 0 \end{bmatrix}, A_2 = \begin{bmatrix} 0.3333 & 0 & 0 \\ 0 & 9.006 & 0 \\ 0 & 0 & 0.16667 \end{bmatrix},$$

$$A_3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 9.006 & 0 \\ 0 & 0 & 0 \end{bmatrix}, A_4 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 9.006 & 0 \\ 0 & 0 & 0.16667 \end{bmatrix}, B_i = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix},$$

$$R = 0, Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

The numerical implementation could be adapted by taking t_f = 2 for solving the related MRDE of the above linear system. The appropriate matrices are substituted in (4.5), the MRDE is transformed into system of differential equation in k_{11} and k_{12}. The numerical solutions of MRDE are calculated and displayed in Table 4.9. Similarly the solution of the above system with the matrix A_2, A_3 and A_4 can be found out using ACP.

By generating the graph randomly, 80 generations with 50-100 number of ants each, are sent out through the graph with $\rho = 0.5$ and $\beta = 1$. In Figure 4.19, the trial solutions are

Table 4.10: Comparison results for k_{11}(t) between the ACP and GP.

k_{11}(t)	Average No. Generations	Average No. Ants	Average Time(secs)
ACP	45	80	120.53
GP	234	-	-

plotted. These trial solutions are compared with the exact solution given as:

$$k_{11}(t) = \frac{8333}{3333} e^{\frac{-3333}{5000}t + \frac{3333}{2500}} - \frac{5000}{3333}\,.$$

At 13^{th} generation, the trial solution computed the fitness values was 0.4444 at $t = 2$ and the expression was:

$$Tours := k_{11}(t) = e(2-t);$$

$$Expressions := k_{11}(t) = e^{2-t}.$$

After the global pheromone update, the ACP method gives an expression with the fitness function equals to 0.2172 its functional form is given as:

$$Tours := k_{11}(t) = 6/5 * e(2-t) - 1/5;$$

$$Expressions := k_{11}(t) = \frac{6}{5}e^{2-t} - \frac{1}{5}$$

at 28^{th} generation. Later at 43^{rd} generation, the ACP predicted an expression with a fitness function less than 0.0707 with its functional form given as:

$$Tours := k_{11}(t) = 7/5 * e(2-t) - 2/5;$$

$$Expressions := k_{11}(t) = \frac{7}{5}e^{2-t} - \frac{2}{5}.$$

Only after reaching 70^{th} generation, the final solution is obtained. The fitness function is $4.4e^{-09}$ and the expression was given as:

$$Tours := k_{11}(t) = 5/2 * e((4 - 2 * t)/3) - 3/2;$$

$$Expressions := k_{11}(t) = \frac{5}{2}e^{4-2t} - \frac{3}{2}.$$

The parse trees for the solutions k_{11} is shown in Figure 4.20. The comparison between the ACP and GP method are given in Table 4.10.

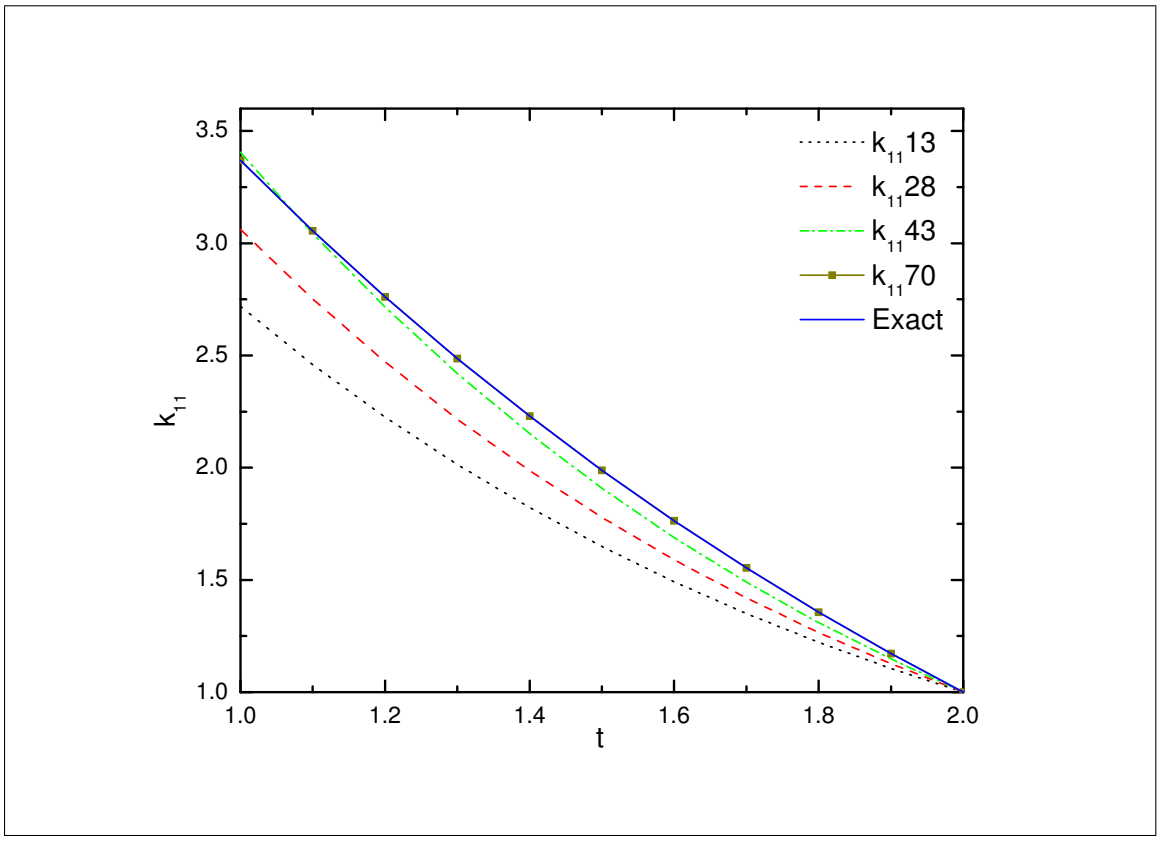

Figure 4.19: Candidate solutions

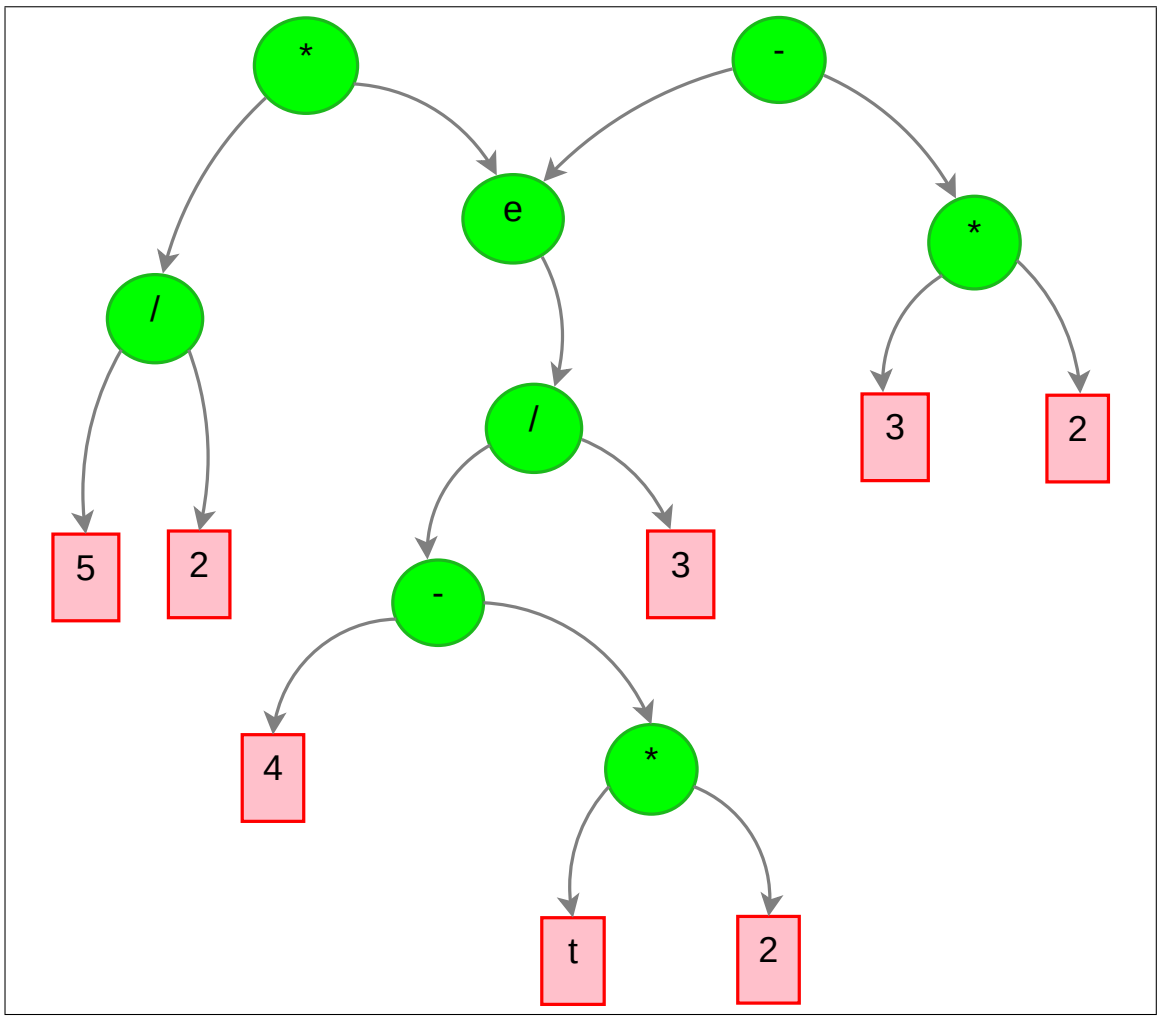

Figure 4.20: Tours of ant and its parse tree.

CHAPTER 5

SOLUTION OF MATRIX RICCATI DIFFERENTIAL EQUATION OF OPTIMAL FUZZY CONTROLLER DESIGN WITH SIMULINK

In this chapter, we implement the Simulink approach to compute the solution of MRDE. Simulink is a graphical extension in MATLAB and is best for modeling and simulation of dynamic systems and this includes the nonlinear system. Another advantage is its capability to take on initial conditions for dynamical systems. Simulink runs based on systems drawn as a block of diagrams which can be translated either as system of ordinary differential equations, transfer functions, signal routing etc. Therefore its a very useful tools for everyone. The Simulink approach suggests an alternative method to solve the MRDE and nonlinear fuzzy modelling problems discussed in this present work.

5.1 Nonlinear singular system with cross term

In this section, we are dealing with the optimal control problem where a quadratic performance index is required to be minimized. Consider the optimal control problem, where

$$J = \frac{1}{2}\left(x^T(t)\,Qx(t) + u^T(t)\,Ru(t) + 2u^T(t)\,Hx(t)\right)dt,$$

is minimized, subject to the linear singular fuzzy system R_i :

If $z_1(t)$ is M_{i1} and $z_2(t)$ is M_{i2} then

$$E_i\dot{x}(t) = A_i(x)\,x(t) + B_iu(t), \qquad x(0) = x_0,\ i = 1, 2, .., r$$

where

$$\dot{x}(t) = \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix}, x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}, E_i = \begin{bmatrix} 3 & 0 \\ 0 & 0 \end{bmatrix},$$

Table 5.1: T-S Fuzzy Model Implication

Implication	Premise	Consequence	Truth Value
Rule 1	$M_1(z_1) = 0.5483$, $N_1(z_2) = 0.2498$	$\dot{x}_1 = 0.25$, $\dot{x}_2 = 8.5075$	$0.5483 \bigwedge 0.2498 = 0.2498$
Rule 2	$M_1(z_1) = 0.5483$, $N_2(z_2) = 0.7502$	$\dot{x}_1 = 0.25$, $\dot{x}_2 = 7.2575$	$0.5483 \bigwedge 0.7502 = 0.5483$
Rule 3	$M_2(z_1) = 0.4517$, $N_1(z_2) = 0.2498$	$\dot{x}_1 = 0.25$, $\dot{x}_2 = 2.0725$	$0.4517 \bigwedge 0.2498 = 0.2498$
Rule 4	$M_1(z_1) = 0.4517$, $N_1(z_2) = 0.7502$	$\dot{x}_1 = 0.25$, $\dot{x}_2 = 0.8225$	$0.4517 \bigwedge 0.7502 = 0.4517$

$$A_i(x) = \begin{bmatrix} 0 & 1 \\ z_1(t) & z_2(t) \end{bmatrix}, H = \begin{bmatrix} 1 & 0 \end{bmatrix}, B_i = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, R = 1, Q = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

are substituted in the equation given below

$$E_i^T \dot{K}_i E_i + E_i^T K_i A_i + A_i^T K_i E_i + Q - \left(H^T + E_i^T K_i B_i\right) R^{-1} \left(H + B_i^T K_i E_i\right) = 0,$$

where $z_1(t) = x_1(t)$ and $z_2(t) = x_2(t)$. Let $x_1 \in [0.5, 3.5]$ and $x_2 \in [-1, 4]$. The minimum and maximum values of z_1 and z_2 can be calculated, the membership functions can be obtained. Then, the nonlinear system is represented by the following fuzzy model.

Model Rule 1: If $z_1(t)$ is Positive and $z_2(t)$ is Big, Then $\dot{x}(t) = A_1 x(t) + Bu$.

Model Rule 2: If $z_1(t)$ is Positive and $z_2(t)$ is Small, Then $\dot{x}(t) = A_2 x(t) + Bu$.

Model Rule 3: If $z_1(t)$ is Negative and $z_2(t)$ is Small, Then $\dot{x}(t) = A_3 x(t) + Bu$.

Model Rule 4: If $z_1(t)$ is Negative and $z_2(t)$ is Big, Then $\dot{x}(t) = A_4 x(t) + Bu$.

Here

$$A_1(x) = \begin{bmatrix} 0 & 1 \\ 3.5 & 4 \end{bmatrix}, A_2(x) = \begin{bmatrix} 0 & 1 \\ 3.5 & -1 \end{bmatrix}, A_3(x) = \begin{bmatrix} 0 & 1 \\ 0.5 & 4 \end{bmatrix},$$

$$A_4(x) = \begin{bmatrix} 0 & 1 \\ 0.5 & -1 \end{bmatrix}.$$

If $z_1 = x_1 = 2.145$ and $z_2 = x_2 = 0.25$, the T-S fuzzy modelling implication can be derived as in Table 5.1. Now the final values for $\dot{x}_1$ and $\dot{x}_2$ in the T-S fuzzy defuzzification process, can be calculated as:

$$\dot{x}_1 = \frac{(0.2498.0.25)+(0.5483.0.25)+(0.2498.0.25)+(0.4517.0.25)}{(0.2498+0.5483+0.2498+0.4517)} = 0.25,$$

$$\dot{x}_2 = \frac{(0.2498.8.5075)+(0.5483.7.2575)+(0.2498.2.0725)+(0.4517.0.8225)}{(0.2498+0.5483+0.2498+0.4517)} = 4.6639.$$

The results of $\dot{x}_1$ and $\dot{x}_2$ from the T-S fuzzy approximation are either close or similar to the original system where $\dot{x}_1$ = 0.2490 and $\dot{x}_2$ = 4.6630. The Simulink model shown in

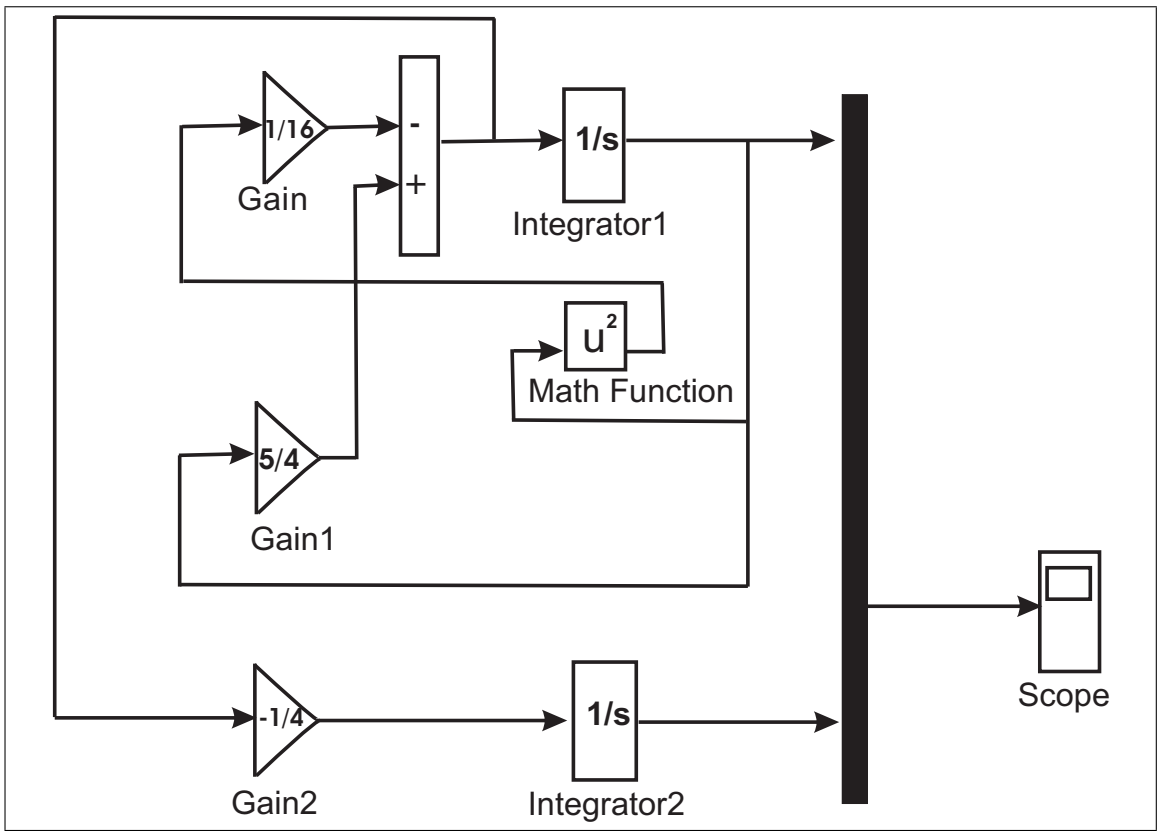

Figure 5.1: Simulink Model.

Figure 5.1 represents the systems of differential equations in k_{11} and k_{12}, with the terminal conditions $k_{11}(2) = 1.0$ and $k_{12}(2) = 0.0$.

These numerical solutions of MRDE are shown in Table 5.2. Similarly the solution of the above system with the matrix A_2, A_3 and A_4 can be solved using simulink. The solution of MRDE of optimal fuzzy controller design for nonlinear singular system with cross term has been obtained by using Simulink. The results are similar to the exact solution. A numerical example is given to illustrate the proposed method. The computational work of the optimal solutions are done in Matlab on PC, CPU 2.0GHz.

5.2 Fuzzy modelling of microbial type growth model for ethanol fermentation process and the optimal control using Simulink

5.2.1 S-type microbial growth model

The theoretical details for this S-type microbial growth model has been described in the previous chapter. Consider the optimal control problem,

Table 5.2: Solutions of MRDE.

	Exact		Simulink	
t	$k_{11}(t)$	$k_{12}(t)$	$k_{11}(t)$	$k_{12}(t)$
0.0	0.0860	0.2285	0.01231	0.0860
0.2	0.1103	0.2224	0.1103	0.2224
0.4	0.1415	0.2146	0.02760	0.2146
0.6	0.1813	0.2047	0.1813	0.2047
0.8	0.2322	0.1920	0.2321	0.1920
1.0	0.2971	0.1757	0.2971	0.1757
1.2	0.3799	0.1550	0.3799	0.1550
1.4	0.4852	0.1287	0.4852	0.1287
1.6	0.6187	0.0953	0.6187	0.0953
1.8	0.7875	0.0531	0.7875	0.0531
2.0	1.00000	0.00000	1.00000	0.0000

$$J = \frac{1}{2}x^T(t_f)\,Sx(t_f) + \frac{1}{2}\int_0^{t_f}\left[x^T(t)\,Qx(t) + u^T(t)\,Ru(t)\right]dt$$

is minimized with subject to the linear T-S fuzzy system R^i: If x_j is M_{ji}, $i = 1, ..., 4$ and $j = 1, 2, 3$. Then the values of x_1, x_2 and x_3 are taken as $x_1 \in [1.68e^{-06}, 0.478]$, x_2 is fixed and $x_3 \in [0.306, 0.542]$. The value of μ_{max} is given in (Baranyi & Roberts, 1994) (i.e. μ_{max} = 9.006, whereas $Y_{N/S} = 19.5147$). The minimum and maximum values of z_1 and z_2 can be calculated and the membership functions can be obtained. Then, the nonlinear system is represented by the following fuzzy model:

Model Rule 1: If $z_1(t)$ is Positive and $z_2(t)$ is Big, Then $\dot{x}(t) = A_1x(t) + Bu$.

Model Rule 2: If $z_1(t)$ is Positive and $z_2(t)$ is Small, Then $\dot{x}(t) = A_2x(t) + Bu$.

Model Rule 3: If $z_1(t)$ is Negative and $z_2(t)$ is Small, Then $\dot{x}(t) = A_3x(t) + Bu$.

Model Rule 4: If $z_1(t)$ is Negative and $z_2(t)$ is Big, Then $\dot{x}(t) = A_4x(t) + Bu$.

Here

$$A_1 = \begin{bmatrix} 2.65 & 0 & 0 \\ 0 & 9.01 & 0 \\ 0 & 0 & 6.64e-04 \end{bmatrix}, A_2 = \begin{bmatrix} 2.65 & 0 & 0 \\ 0 & 9.01 & 0 \\ 0 & 0 & 4.27e-07 \end{bmatrix},$$

$$A_3 = \begin{bmatrix} -1.47e-02 & 0 & 0 \\ 0 & 9.01 & 0 \\ 0 & 0 & 6.64e-04 \end{bmatrix}, A_4 = \begin{bmatrix} -1.47e-02 & 0 & 0 \\ 0 & 9.006 & 0 \\ 0 & 0 & -4.2e-07 \end{bmatrix},$$

Table 5.3: T-S fuzzy model implication

Implication	Premise	Consequence	Truth Value
Rule 1	$M_1(z_1) = 0.10$ $N_1(z_2) = 2.19e^{-06}$	$\dot{x}_1 = 2.65e^{-05}$ $\dot{x}_2 = 0.90$ $\dot{x}_3 = 2e^{-04}$	$M_1(z_1) \wedge N_1(z_2) = N_1(z_2)$
Rule 2	$M_1(z_1) = 0.10$ $N_2(z_2) = 1.0$	$\dot{x}_1 = 2.65e^{-05}$ $\dot{x}_2 = 0.90$ $\dot{x}_3 = -1.30e^{-07}$	$M_1(z_1) \wedge N_2(z_2) = M_1(z_1)$
Rule 3	$M_2(z_1) = 0.90$ $N_1(z_2) = 2.19e^{-06}$	$\dot{x}_1 = -1.47e^{-07}$ $\dot{x}_2 = 0.90$ $\dot{x}_3 = 2.06e^{-04}$	$M_2(z_1) \wedge N_1(z_2) = N_1(z_2)$
Rule 4	$M_2(z_1) = 0.90$ $N_2(z_2) = 1.0$	$\dot{x}_1 = -1.47e^{-07}$ $\dot{x}_2 = 0.90$ $\dot{x}_3 = -1.30e^{-07}$	$M_2(z_1) \wedge N_2(z_2) = M_2(z_1)$

$$B_i = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, R = 0, Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

If $x_1 = 1.0e^{-05}$, $x_2 = 0.1$ and $x_3 = 0.31$, the T-S fuzzy modelling implication can be derived as in Table 5.3. Now the final values for $\dot{x}_1$ and $\dot{x}_3$, in T-S fuzzy defuzzification process, can be calculated as:

$$\dot{x}_1 = \frac{(0.2498 \cdot 0.25) + (0.5483 \cdot 0.25) + (0.2498 \cdot 0.25) + (0.4517 \cdot 0.25)}{(0.2498 + 0.5483 + 0.2498 + 0.4517)} = 0.25,$$

$$\dot{x}_2 = \frac{(0.2498 \cdot 8.5075) + (0.5483 \cdot 7.2575) + (0.2498 \cdot 2.0725) + (0.4517 \cdot 0.8225)}{(0.2498 + 0.5483 + 0.2498 + 0.4517)} = 4.6639$$

Comparing the values of $\dot{x}_1 = 2.54e^{-06}$ and $\dot{x}_3 = -1.30e^{-07}$ of the original system, the T-S fuzzy approximation is more or less similar. The numerical implementation could be adapted by taking $t_f = 2$ for solving the related MRDE of the above linear system. The appropriate matrices are substituted in the equation given below,

$$\dot{K}_{ii} + K_i A_i + A_i^T K_i + Q - K_i B_i R^{-1} B_i^T K_i = 0$$

and the MRDE is transformed into system of differential equation in k_{11}, k_{12}, k_{13}, k_{22}, k_{23} and k_{33}. The equidistant points in the interval $[0,\ 2]$ are taken as input vector. The Simulink model is shown in Figure 5.2 represents the systems of differential equations with the terminal conditions $k_{11} = 1.0$ and $k_{12} = k_{13} = k_{22} = k_{23} = k_{33} = 0.0$. The numerical solutions of MRDE are calculated and displayed in Table 5.4. In the similar way, the MRDE can be solved for the matrices A_2, A_3 and A_4.

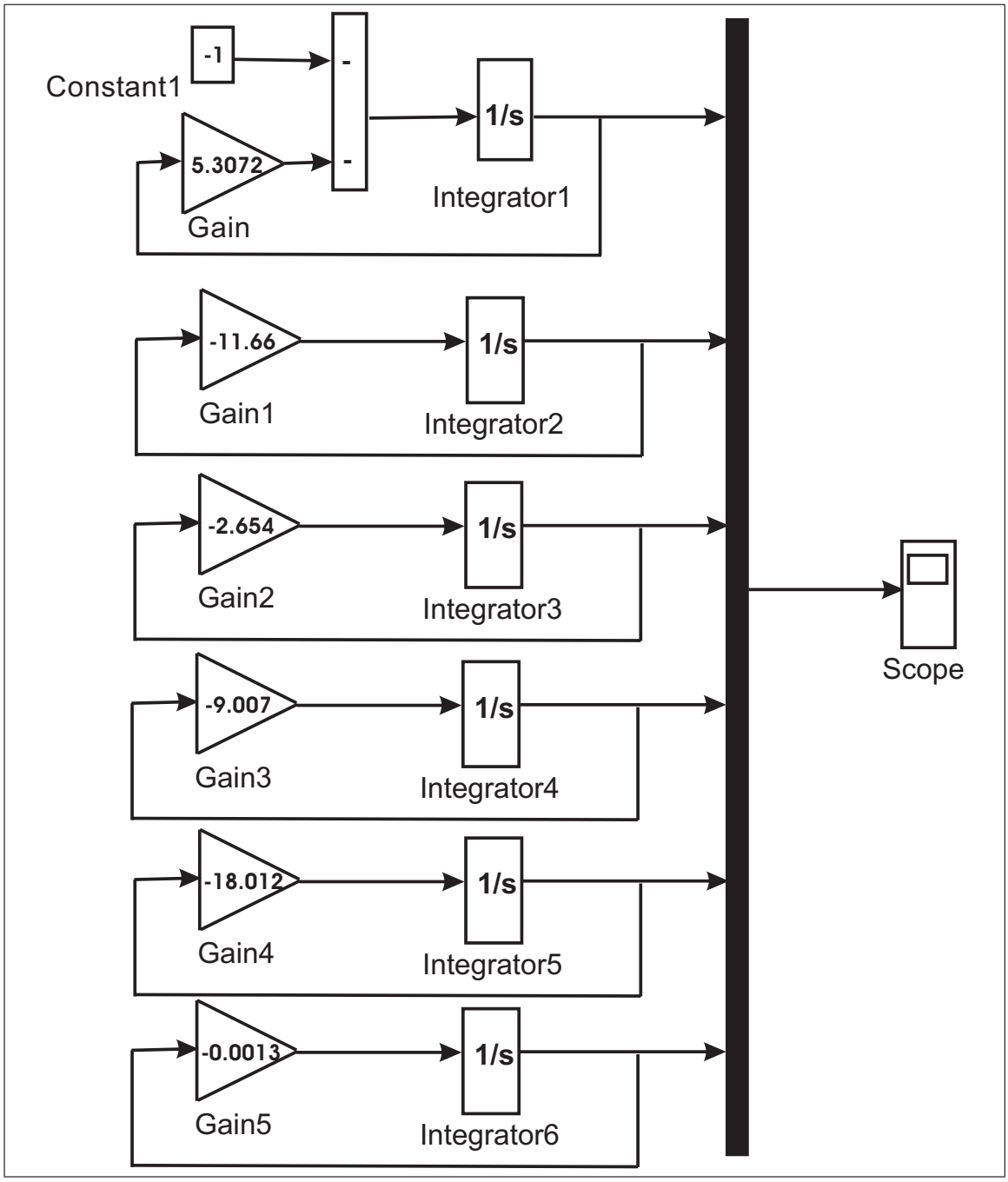

Figure 5.2: Simulink Model

Table 5.4: Solutions for MRDE

t	Exact k_{11}	Simulink k_{11}
0	48389	48389
0.24	13690	13693
0.44	4736	4737
0.64	1639	1639
0.72	1072	1072
0.84	568	567
1.04	197	196
1.24	68	68
1.44	23	23
1.84	3	3
2	1	1

5.2.2 P-type microbial growth model

The incapability of the reference growth model (Baranyi & Roberts, 1994) to explain complicated and realistic situation (e.g., co-cultural growth, growth in structured media) due to the lacks of mechanistic base in the model itself have urged other researchers to extend and improve the microbial predictive growth model. Van Impe et al. (2006) reported in their work where they had proposed two novel class of predictive growth models. In this work, the P-type has been applied together with the T-S fuzzy system. The P-type microbial growth model is given below as

$$\dot{N}(t) = \left(\frac{Q(t)}{1+Q(t)}\right).\mu_{max}.\left(1-\frac{P(t)}{K_p}\right).N(t) \quad \text{with } N(t=0) = N_0,$$

$$\dot{Q}(t) = \mu_{max}.Q(t) \quad \text{with } Q(t=0) = Q_0,$$

$$\dot{S}(t) = -\left(\frac{Q(t)}{1+Q(t)}\right).\mu_{max}.\left(1-\frac{P(t)}{K_p}\right).N(t) \quad \text{with } S(t=0) = N_0.$$

The first differential equation describes the evolution of the microbial load $N(t)$ in time. It consists of the adjustment function which describes the lag phase by means of a variable representing the physiological state of the cells $Q(t)$, as well as the inhibition function which is a linear function of the toxic product concentration $P(t)$. The second differential equation, describes the evolution of $Q(t)$, which increase exponentially, whereas the third differential equation represents the evolution of the toxic product concentration $P(t)$. K_P refers as the concentration of the product at which growth ceases whereas $1/K_P$ described as the sensitivity of the microbial cell towards $P(t)$.

For the P-type microbial growth, to deal with the optimal control problem and to minimize with subject to the linear T-S fuzzy system, the procedure is almost similar to the one

described for the S-type microbial growth. For this case, the value of μ_{max} and K_P is given in (Van Impe et al., 2006) (i.e., μ_{max}= 9.006, K_P = 7.5022). The minimum and maximum values z_1, z_2, z_3 and z_4 can be calculated and the membership functions can be obtained. Then, the nonlinear system is represented by the following fuzzy model.

Model Rule 1: If $z_1(t)$ is Positive , $z_2(t)$ is Big, $z_3(t)$ is Positive, $z_4(t)$ is Big Then $\dot{x}(t) = A_1 x(t) + Bu$.

Model Rule 2: If $z_1(t)$ is Positive, $z_2(t)$ is Big, $z_3(t)$ is Positive, $z_4(t)$ is Small Then $\dot{x}(t) = A_2 x(t) + Bu$.

: : : : : : : :

Model Rule 16: If $z_1(t)$ is Negative and $z_2(t)$ is "Small", $z_3(t)$ is Negative, $z_4(t)$ is Big, Then $\dot{x}(t) = A_{16} x(t) + Bu$.

Here

$$S = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, R = 0, Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$A_1(x) = \begin{bmatrix} 4.896 & 0 & 5.37e^{-10} \\ 0 & 9.01 & 0 \\ 0.018 & 0 & 0.3119 \end{bmatrix}, A_2(x) = \begin{bmatrix} 4.896 & 0 & -0.3119 \\ 0 & 9.01 & 0 \\ 0.018 & 0 & 0.3119 \end{bmatrix},$$

$$A_3(x) = \begin{bmatrix} 4.896 & 0 & 5.37e^{-10} \\ 0 & 9.01 & 0 \\ -4.896 & 0 & 0.3119 \end{bmatrix}, A_4(x) = \begin{bmatrix} 4.896 & 0 & 5.37e^{-10} \\ 0 & 9.01 & 0 \\ 0.018 & 0 & -5.37e^{-10} \end{bmatrix},$$

$$A_5(x) = \begin{bmatrix} -0.018 & 0 & 5.37e^{-10} \\ 0 & 9.01 & 0 \\ 0.018 & 0 & 0.3119 \end{bmatrix}, A_6(x) = \begin{bmatrix} -0.018 & 0 & -0.3119 \\ 0 & 9.01 & 0 \\ 0.018 & 0 & 0.3119 \end{bmatrix},$$

$$A_7(x) = \begin{bmatrix} -0.018 & 0 & -0.3119 \\ 0 & 9.01 & 0 \\ 0.018 & 0 & 0.3119 \end{bmatrix}, A_8(x) = \begin{bmatrix} -0.018 & 0 & 5.37e^{-10} \\ 0 & 9.01 & 0 \\ 0.018 & 0 & -5.37e^{-10} \end{bmatrix},$$

$$A_9(x) = \begin{bmatrix} 4.896 & 0 & -0.3119 \\ 0 & 9.01 & 0 \\ -4.896 & 0 & 0.3119 \end{bmatrix}, A_{10}(x) = \begin{bmatrix} 4.896 & 0 & -0.3119 \\ 0 & 9.01 & 0 \\ -4.896 & 0 & -5.37e^{-10} \end{bmatrix},$$

$$A_{11}(x) = \begin{bmatrix} 4.896 & 0 & -0.3119 \\ 0 & 9.01 & 0 \\ 0.018 & 0 & -5.37e^{-10} \end{bmatrix}, A_{12}(x) = \begin{bmatrix} 4.896 & 0 & 5.37e^{-10} \\ 0 & 9.01 & 0 \\ -4.896 & 0 & -5.37e^{-10} \end{bmatrix},$$

$$A_{13}(x) = \begin{bmatrix} -0.0185 & 0 & -0.3119 \\ 0 & 9.01 & 0 \\ -4.896 & 0 & 0.3119 \end{bmatrix}, A_{14}(x) = \begin{bmatrix} -0.0185 & 0 & -0.3119 \\ 0 & 9.01 & 0 \\ -4.896 & 0 & -5.37e^{-10} \end{bmatrix},$$

$$A_{15}(x) = \begin{bmatrix} -0.0185 & 0 & -0.3119 \\ 0 & 9.01 & 0 \\ -4.896 & 0 & -5.37e^{-10} \end{bmatrix}, A_{16}(x) = \begin{bmatrix} -1.47e^{-02} & 0 & 0 \\ 0 & 9.01 & 0 \\ 0 & 0 & -4.2e^{-07} \end{bmatrix}.$$

If $x_1 = 1.74e^{-07}$, $x_2 = 7.0e^{-04}$ and $x_3 = 1.0e^{-09}$, the T-S fuzzy modelling implication can be derived as in Tables 5.5 and 5.6. Now the final values for and in T-S fuzzy defuzzification process, can be calculated and is shown in Table 5.7.

$$\dot{x}_1 = {}^{8.44e^{-09}}/_{1.02} = 8.3e^{-09}, \dot{x}_3 = {}^{-8.44e^{-09}}/_{1.02} = -8.3e^{-09}.$$

Comparing the values of $\dot{x}_1 = 1.46e^{-10}$ and $\dot{x}_3 = -1.46e^{-10}$ of the original system, the T-S fuzzy approximation is more or less similar. The numerical implementation could be adapted by taking $t_f = 1$ for solving the related MRDE of the above linear system. The appropriate matrices are substituted in equation given below:

$$\dot{K}_{ii} + K_i A_i + A_i^T K_i + Q - K_i B_i R^{-1} B_i^T K_i = 0.$$

The MRDE is transformed into system of differential equation in k_{11}, k_{12}, k_{13}, k_{22}, k_{23} and k_{33}. The equidistant points in the interval [0,1] are taken as input vector. The simulink model is shown in Fig. 5.3 represents the systems of differential equations with the terminal conditions $k_{11} = 1.0$ and $k_{12} = k_{13} = k_{22} = k_{23} = k_{33} = 0.0$. The numerical solutions of the MRDE are computed and shown in Table 5.8. Similarly, the MRDE can be solved for the matrices A_1, A_2, A_3,..., and A_{16}.

Table 5.5: T-S fuzzy model implication: Rule 1 - Rule 8.

Implication	Premise	Consequence	Truth Value
Rule 1	$M_1(z_1) = 4.848e^{-03}$ $N_1(z_2) = 6.348e^{-04}$ $R_1(z_3) = 0.9952$ $S_1(z_4) = 0.9994$	$\dot{x}_1 = 8.519e^{-07}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = 3.361e^{-09}$	$M_1(z_1) \wedge N_1(z_2) \wedge R_1(z_3) \wedge S_1(z_4) = N_1(z_2)$
Rule 2	$M_1(z_1) = 4.848e^{-03}$ $N_1(z_2) = 6.348e^{-04}$ $R_1(z_3) = 0.9952$ $S_2(z_4) = 6.348e^{-04}$	$\dot{x}_1 = 8.519e^{-07}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = 3.361e^{-09}$	$M_1(z_1) \wedge N_1(z_2) \wedge R_1(z_3) \wedge S_2(z_4) = N_1(z_2)$
Rule 3	$M_1(z_1) = 4.848e^{-03}$ $N_1(z_2) = 6.348e^{-04}$ $R_2(z_3) = 4.848e^{-03}$ $S_1(z_4) = 0.9994$	$\dot{x}_1 = 8.519e^{-07}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = -8.519e^{-07}$	$M_1(z_1) \wedge N_1(z_2) \wedge R_2(z_3) \wedge S_1(z_4) = N_1(z_2)$
Rule 4	$M_1(z_1) = 4.848e^{-03}$ $N_2(z_2) = 0.9994$ $R_1(z_3) = 0.9952$ $S_1(z_4) = 0.9994$	$\dot{x}_1 = 8.519e^{-07}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = 3.049e^{-09}$	$M_1(z_1) \wedge N_2(z_2) \wedge R_1(z_3) \wedge S_1(z_4) = M_1(z_1)$
Rule 5	$M_2(z_1) = 0.9952$ $N_1(z_2) = 6.348e^{-04}$ $R_1(z_3) = 0.9952$ $S_1(z_4) = 0.9994$	$\dot{x}_1 = -3.049e^{-09}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = 3.361e^{-09}$	$M_2(z_1) \wedge N_1(z_2) \wedge R_1(z_3) \wedge S_1(z_4) = N_1(z_2)$
Rule 6	$M_2(z_1) = 0.9952$ $N_1(z_2) = 6.348e^{-04}$ $R_1(z_3) = 0.9952$ $S_2(z_4) = 6.348e^{-04}$	$\dot{x}_1 = -3.049e^{-09}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = 3.361e^{-09}$	$M_2(z_1) \wedge N_1(z_2) \wedge R_1(z_3) \wedge S_2(z_4) = N_1(z_2)$
Rule 7	$M_2(z_1) = 0.9952$ $N_1(z_2) = 6.348e^{-04}$ $R_2(z_3) = 4.848e^{-03}$ $S_1(z_4) = 0.9994$	$\dot{x}_1 = -3.049e^{-09}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = -8.561e^{-07}$	$M_2(z_1) \wedge N_1(z_2) \wedge R_2(z_3) \wedge S_1(z_4) = N_1(z_2)$
Rule 8	$M_2(z_1) = 0.9952$ $N_2(z_2) = 0.9994$ $R_1(z_3) = 0.9952$ $S_1(z_4) = 0.9994$	$\dot{x}_1 = -3.049e^{-09}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = -8.561e^{-07}$	$M_2(z_1) \wedge N_2(z_2) \wedge R_1(z_3) \wedge S_1(z_4) = R_1(z_3)$

Table 5.6: T-S fuzzy model implication: Rule 9 - Rule 16.

Implication	Premise	Consequence	Truth Value
Rule 9	$M_1(z_1) = 4.848e^{-03}$ $N_1(z_2) = 6.348e^{-04}$ $R_2(z_3) = 4.848e^{-03}$ $S_2(z_4) = 6.348e^{-04}$	$\dot{x}_1 = 8.519e^{-07}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = -8.519e^{-07}$	$M_1(z_1) \wedge N_1(z_2) \wedge R_2(z_3) \wedge S_2(z_4) = N_1(z_2)$
Rule 10	$M_1(z_1) = 4.848e^{-03}$ $N_2(z_2) = 0.9994$ $R_2(z_3) = 4.848e^{-03}$ $S_2(z_4) = 6.348e^{-04}$	$\dot{x}_1 = 8.519e^{-07}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = -8.519e^{-07}$	$M_1(z_1) \wedge N_2(z_2) \wedge R_2(z_3) \wedge S_2(z_4) = S_2(z_4)$
Rule 11	$M_1(z_1) = 4.848e^{-03}$ $N_2(z_2) = 0.9994$ $R_1(z_3) = 0.9952$ $S_2(z_4) = 6.348e^{-04}$	$\dot{x}_1 = 8.519e^{-07}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = 3.049e^{-09}$	$M_1(z_1) \wedge N_2(z_2) \wedge R_1(z_3) \wedge S_2(z_4) = S_2(z_4)$
Rule 12	$M_1(z_1) = 4.848e^{-03}$ $N_2(z_2) = 0.9994$ $R_2(z_3) = 4.848e^{-03}$ $S_1(z_4) = 0.9994$	$\dot{x}_1 = 8.519e^{-07}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = -8.519e^{-07}$	$M_1(z_1) \wedge N_2(z_2) \wedge R_2(z_3) \wedge S_1(z_4) = M_1(z_1)$
Rule 13	$M_2(z_1) = 0.9952$ $N_1(z_2) = 6.348e^{-04}$ $R_2(z_3) = 4.848e^{-03}$ $S_2(z_4) = 6.348e^{-04}$	$\dot{x}_1 = -3.360e^{-09}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = -8.519e^{-07}$	$M_2(z_1) \wedge N_1(z_2) \wedge R_2(z_3) \wedge S_2(z_4) = N_1(z_2)$
Rule 14	$M_2(z_1) = 0.9952$ $N_2(z_2) = 0.9994$ $R_2(z_3) = 4.848e^{-03}$ $S_2(z_4) = 6.348e^{-04}$	$\dot{x}_1 = -3.360e^{-09}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = -8.519e^{-07}$	$M_2(z_1) \wedge N_2(z_2) \wedge R_2(z_3) \wedge S_2(z_4) = S_2(z_4)$
Rule 15	$M_2(z_1) = 0.9952$ $N_2(z_2) = 0.9994$ $R_1(z_3) = 0.9952$ $S_2(z_4) = 6.348e^{-04}$	$\dot{x}_1 = -3.360e^{-09}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = 3.049e^{-09}$	$M_2(z_1) \wedge N_2(z_2) \wedge R_1(z_3) \wedge S_2(z_4) = S_2(z_4)$
Rule 16	$M_2(z_1) = 0.9952$ $N_2(z_2) = 0.9994$ $R_2(z_3) = 4.848e^{-03}$ $S_1(z_4) = 0.9994$	$\dot{x}_1 = -3.049e^{-09}$ $\dot{x}_2 = 6.304e^{-03}$ $\dot{x}_3 = -8.519e^{-07}$	$M_2(z_1) \wedge N_2(z_2) \wedge R_2(z_3) \wedge S_1(z_4) = R_2(z_3)$

Table 5.7: Solutions for $\dot{x}_1(t)$and $\dot{x}_3(t)$

j	$h_j(z_j(t))$	$A_j x_1(t)$	$h_j(z_j(t)) \cdot A_j x_1(t)$	$A_j x_3(t)$	$h_j(z_j(t)) \cdot A_j x_1(t)$
1	$6.35e^{-04}$	$8.52e^{-07}$	$5.41e^{-10}$	$3.36e^{-09}$	$2.13e^{-12}$
2	$6.35e^{-04}$	$8.52e^{-07}$	$5.41e^{-10}$	$3.36e^{-09}$	$2.13e^{-12}$
3	$6.35e^{-04}$	$8.52e^{-07}$	$5.41e^{-10}$	$-8.50e^{-07}$	$-5.40e^{-10}$
4	$4.85e^{-03}$	$8.52e^{-07}$	$4.13e^{-09}$	$3.05e^{-09}$	$1.48e^{-11}$
5	$6.35e^{-04}$	$-3.05e^{-09}$	$-1.94e^{-12}$	$3.36e^{-09}$	$2.13e^{-12}$
6	$6.35e^{-04}$	$-3.36e^{-09}$	$-2.13e^{-12}$	$3.36e^{-09}$	$2.13e^{-12}$
7	$6.35e^{-04}$	$-3.05e^{-09}$	$-1.94e^{-12}$	$-8.50e^{-07}$	$-5.40e^{-10}$
8	$9.95e^{-01}$	$-3.05e^{-09}$	$-3.03e^{-09}$	$3.05e^{-09}$	$3.03e^{-09}$
9	$6.35e^{-04}$	$8.52e^{-07}$	$5.41e^{-10}$	$-8.50e^{-07}$	$-5.40e^{-10}$
10	$6.35e^{-04}$	$8.52e^{-07}$	$5.41e^{-10}$	$-8.50e^{-07}$	$-5.40e^{-10}$
11	$6.35e^{-04}$	$8.52e^{-07}$	$5.41e^{-10}$	$3.05e^{-09}$	$1.94e^{-12}$
12	$4.85e^{-03}$	$8.52e^{-07}$	$4.13e^{-09}$	$-8.50e^{-07}$	$-4.10e^{-09}$
13	$6.35e^{-04}$	$-3.36e^{-09}$	$-2.13e^{-12}$	$-8.50e^{-07}$	$-5.40e^{-10}$
14	$6.35e^{-04}$	$-3.36e^{-09}$	$-2.13e^{-12}$	$-8.50e^{-07}$	$-5.40e^{-10}$
15	$6.35e^{-04}$	$-3.36e^{-09}$	$-2.13e^{-12}$	$3.05e^{-09}$	$1.94e^{-12}$
16	$4.85e^{-03}$	$-3.05e^{-09}$	$-1.48e^{-11}$	$-8.50e^{-07}$	$-4.10e^{-09}$
SUM=	1.02		$8.44e^{-09}$		$8.44e^{-09}$

Table 5.8: Solutions for $k_{11}(t)$

t	Simulink	Exact
0	19717	19717
0.32	859	859
0.64	37	37
0.8	8	8
1	1	1

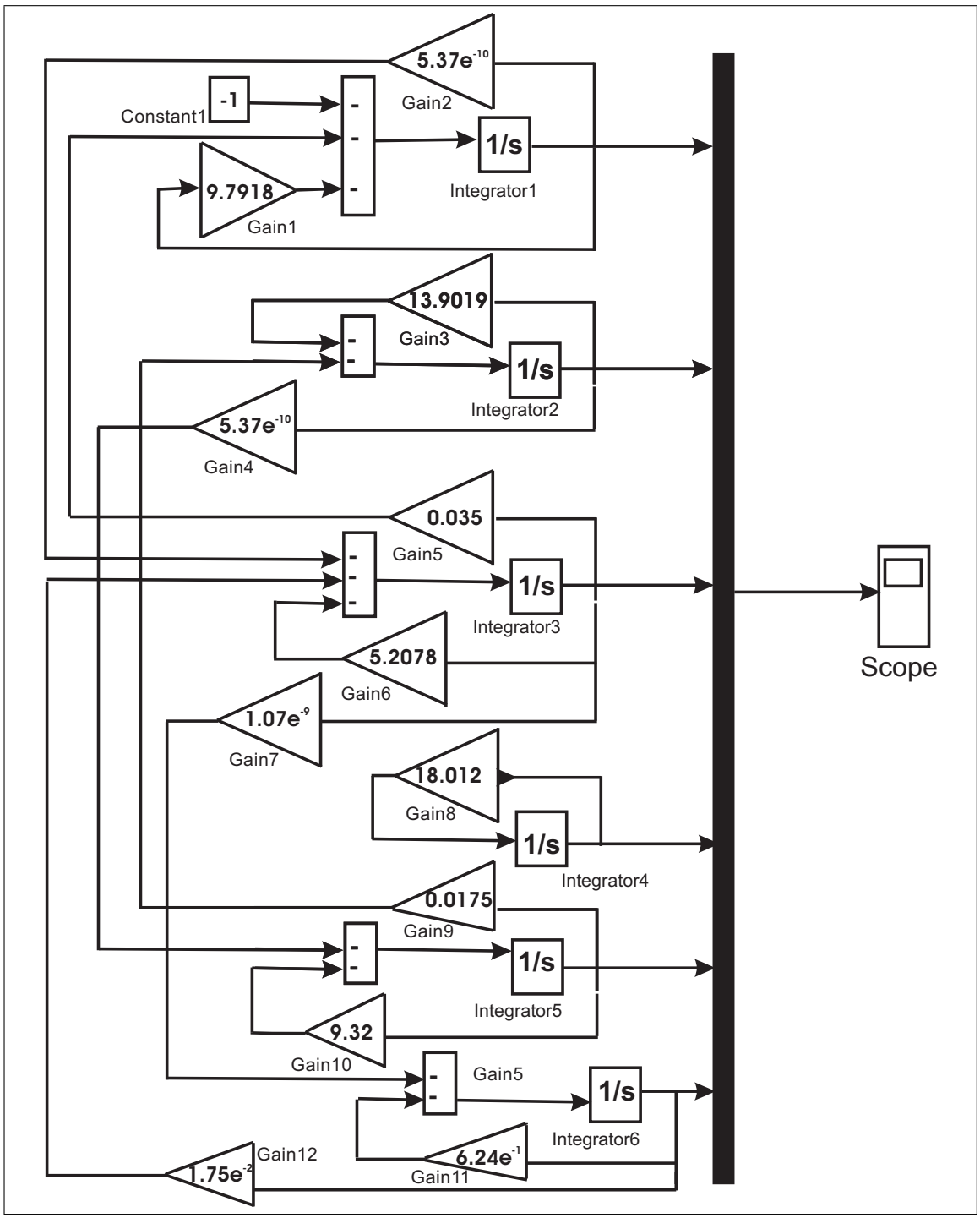

Figure 5.3: Simulink Model

CHAPTER 6

CONCLUSION AND FUTURE DIRECTIONS

In this thesis, a modified ACP have been developed. The function of this modified ACP approach is to generate solutions which is similar or close to the analytical answers, in the form of expressions. This novel approach has shown its validity for solving the ODEs, NLODE, SODE, PDEs and SPDEs. Furthermore, we implemented the ACP for solving the MRDE in linear and nonlinear fuzzy optimal control problem. It is found that the modified ACP can be used to solve complicated differential equations with reasonable computational time. In the ACP method, the average number of generations for finding the solutions increased as the differential equations become more difficult. Comparison between the ACP and the GP method showed that the present ACP gives faster solutions within reasonable range of average number of generations. By manipulating the terminal criteria, the ACP solution predicted either complete agreement or approximately close answer to the exact solution. This approach is very vital for solving complicated differential equations which suggest or predict simpler solutions with very good accuracy. We have also demonstrated conclusively that the Simulink approach provides an alternative method to solve the MRDE and nonlinear fuzzy modelling problems without writing complex coding.

For our future work, we tend to add more terminal symbols and functions in the ACP algorithm in order to test it out with other more complicated differential equations which are used in dynamical and engineering systems. We also hope that this modified ACP approach can be used for other problems such as identifying the structure of unknown molecules especially in the analysis of complex biological samples, generating music or language by moving the artificial ants on a space graph with vertices and edges . As the ant chooses its way, the pheromone is deposited on the edges, simultaneously building up

a melody or a sentence.

PUBLICATIONS UNDER REVIEW

1. M. Z. M. Kamali, N. Kumaresan, and Kuru Ratnavelu, Solution of fuzzy modelling of engineering and scientific problems using Ant Colony Programming, Applied Mathematical Modelling, (under review).

2. M. Z. M. Kamali, N. Kumaresan, and KuruRatnavelu, Modified ant colony programming for solving matrix Riccati differential equation with linear singular fuzzy system: cross term and singular cost, (to be submitted).

3. M. Z. M. Kamali, N. Kumaresan, Koshy Philip and Kuru Ratnavelu, Fuzzy modelling of optimal control ethanol fermentation process using ant colony programming, (to be submitted).

References

Abdul Samath, J., & Selvaraju, N. (2010). Solution of matrix riccati differential equation for nonlinear singular system using neural networks. *Int. J. Comput. Appl.*, *1*, 48-55.

Adair, C., & Briggs, P. (1993). The concept and application of expert systems in the field of microbiological safety. *J Ind. Microbiol.*, *12*, 263-267.

Ast, J. v., Babuska, R., & Schutter, B. D. (2009). Fuzzy ant colony optimization for optimal control. In *American control conference.*

Back, T. (1998). An overview of parameter control methods by self-adaptation in evolutionary algorithms. *Fundam. Inf.*, *35*, 51-66.

Balasubramaniam, P., & Kumar, A. V. A. (2009). Solution of matrix riccati differential equation for nonlinear singular system using genetic programming. *Genet. Program. Evol. M.*, *10*, 71-89.

Balasubramaniam, P., Samath, J. A., & Kumaresan, N. (2007 a). Optimal control for nonlinear singular systems with quadratic performance using neural networks. *Appl. Math. Comput.*, *187*, 1535–1543.

Balasubramaniam, P., Samath, J. A., Kumaresan, N., & Kumar, A. V. A. (2006). Solution of matrix riccati differential equation for the linear quadratic singular system using neural networks. *Appl. Math. Comput.*, *182*, 1832-1839.

Balasubramaniam, P., Samath, J. A., Kumaresan, N., & Kumar, A. V. A. (2007 b). Neuro approach for solving matrix riccati differential equation,. *Neural Parallel Sci. Comput.*, *15*, 125-135.

Baranyi, J., & Roberts, T. A. (1994). A dynamic approach to predicting bacterial growth in food,. *Int. J. Food Microbiol.*, *23*, 277-294.

Baranyi, J., & Roberts, T. A. (1995). Mathematics of predictive food microbiology. *Int. J. Food Microbiol.*, *26*, 199-218.

Bedrossian, N., Bhatt, S., Kang, W., & Ross, I. M. (2009). Zero-propellant maneuver guidance. *IEEE Contr. Syst. Mag.*, *29*, 53-73.

Bellman, R. E. (1957). *Dynamic programming*. Princeton University Press.

Bhatt, S. (2007). *Optimal reorientation of spacecraft using only control moment gyroscopes*

(Unpublished master's thesis). Dept. of Computational & Applied Mathematics, Rice Univ., Houston.

Boltyanskii, V., Gamkrelidze, R., & Pontryagin, L. (1956). Towards a theory of optimal processes. *Reports Acad. Sci. USSR*, *110*.

Boryczka, M. (2002). Ant colony programming for approximation problems. In *Proceedings of the Eleventh International, Symposium on Intelligent Information Systems, Sopot, Poland.*

Boryczka, M. (2005). Eliminating introns in ant colony programming,. *Fund. Inform.*, *68*, 1-19.

Boryczka, M., & Czech, Z. J. (2002). Solving approximation problems by ant colony programming. In *Breaking Papers at the Genetic and Evolutionary Computation Conference (GECCO-2002).*

Boryczka, M., Czech, Z. J., & Wieczorek, W. (2003). Ant colony programming for approximation problems. In *Proceedings of the Genetic and Evolutionary Computation (GECCO-2003).*

Boryczka, M., & Wiezorek, W. (2003). Solving approximation problems using ant colony programming,. In *Proceedings of AIMETH.*

Brown, M. H., Davies, K. W., Billon, C., Adair, C., & McClure, P. J. (1998). Quantitative microbiological risk assessment: principles applied to determining the comparative risk of salmonellosis from chicken products. *J. Food Prot.*, *61*, 1446-1453.

Burgess, G. (1999). Surveillance system division, electronics and surveillance research laboratory. In (chap. Finding Approximate Analytical Solutions to Differential Equations Using Genetic Programming). Department of Defense, Australia.

Cao, J., Li, P., & Liu, H. (2011). Fuzzy controllers, theory and applications. In (chap. Adaptive Fuzzy Modelling and Control for Non-Linear Systems Using Interval Reasoning and Differential Evolution). Lucian Grigorie (Ed.), ISBN: 978-953-307-543-3, InTech (2011).

Cao, S. G., Rees, N. W., & Feng, G. (1996). Fuzzy control of nonlinear continuous-time systems. *Proc. 35th IEEE Conf. Decision and Control*, 592-597.

Chen, F. C., & Liu, C. C. (1994). Adaptive controlling nonlinear continuous-time using multilayer neural networks. *IEEE. Trans. Automat. Control*, *39*, 1306-1310.

Chen, S., Li, X., & Zho, X. (1998). Stochastic linear quadratic regulators with indefinite control weight costs. *SIAM J. Control Optim.*, *36(5)*, 1685-1702.

Dalgaard, P., Buch, P., & Silberg, S. (2000). SSP online-an internet version of the seafood spoilage predictor software. In *Proceedings of 1st International Conference on Simulation in Food and Bio Industries.*

Davey, K. (1991). Applicability of the davey(linear arrhenius) predictive model to the lag phase

of microbial growth. *J. Appl. Bacteriol.*, *70*, 253-257.

Dorigo, M. (1992). *Optimization, learning and natural algorithms (in italian)* (Unpublished doctoral dissertation). Dipartimento di Elettronica, Politecnico di Milano, Italy.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system: Optimization by a colony of cooperating agents. *IEEE T. Syst. Man. Cy. B*, *26(1)*, 29-41.

Fasshauer, G. E. (1999). Solving differential equations with radial basis functions: Multilevel methods and smoothing,. *Adv. Comput. Math.*, *11*, 139-159.

Garcia-Gimeno, R. M., Hervas-Martinez, C., & de Siloniz, I. (2002). Improving artificial neural networks with a pruning methodology and genetic algorithms for their application in microbial growth prediction in foods. *Int. J. Food Microbiol.*, *72*, 19-30.

Ghanbari, G., & Farahi, M. H. (2014). Optimal control of a delayed HIV infection model via fourier series. *Journal of Nonlinear Dynamics*, *2014*.

Gibson, A. M., Bratchell, N., & Roberts, T. A. (1987). The effect of sodium chloride and temperature on the rate and extent of growth of clostridium botulinum type a in pasteurized pork slurry. *J. Appl. Bacteriol.*, *62*, 479-490.

Gibson, A. M., Bratchell, N., & Roberts, T. A. (1988). Predicting microbial growth: grogro responses of salmonellae in a laboratory mdeium as affected by ph, sodium chloride and storage temperature. *Int. J. Food Microbiol.*, *6(2)*, 155-178.

Jamshidi, M. (1980). An overview on the solutions of the algebraic matrix Riccati equation and related problems. *Large Scale Syst.*, *1*, 167-192.

Jenkins, D. F., & Passino, K. M. (1999). An introduction to nonlinear analysis of fuzzy control systems. *J. Intell. Fuzzy Syst.*, *7*, 75-103.

Jones, J. (1993). A real time database/models base/expert system in predictive microbiology. *J. Ind. Microbiol.*, *12*, 268-272.

Joshi, H. R. (2002). Optimal control of an HIV immunology model. *Optim. Control Appl. Meth.*, *23*, 199-213.

Kawamoto, S., Tada, K., Ishigame, A., & Taniguchi, T. (1993). An approach to stability analysis of second order fuzzy systems. *IEEE T. Neural Netw.*, *4*, 919-930.

Keber, C., & Schuster, M. G. (2002). Option valuation with generalized ant programming. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002).*

Kirschner, D., & Webb, G. F. (1998). Immunotherapy of HIV-1 infection. *J. Biol Syst.*, *6(1)*, 71-83.

Koza, J. R. (1992). *Genetic programming: On the programming of computers by means of natural*

selection. MIT Press, Cambridge, MA.

Koza, J. R. (1994). *Genetic programming II: Automatic discovery of reusable programs*. MIT Press, Cambridge, MA,.

Kramer, O. (2010). Evolutionary self-adaptation: a survey of operators and stratergy parameters. *Evol. Intel.*, *3*, 51-65.

Kumaresan, N. (2010). Optimal control for stochastic linear quadratic singular Takagi-Sugeno fuzzy system using ant colony programming. *Neural Parallel Sci. Comput.*, *18*, 89-108.

Kumaresan, N. (2011). Optimal control for stochastic linear quadratic singular periodic neuro Takagi-Sugeno fuzzy system with singular cost using ant colony programming. *Appl. Math. Model.*, *35*, 3797-3808.

Kumaresan, N. (2012). Optimal control for stochastic singular integro-differential Takagi-Sugeno fuzzy system using ant colony programming. *Filomat*, *26:3*, 415-426.

Kumaresan, N., & Balasubramaniam, P. (2008). Optimal control for stochastic nonlinear singular system using neural network. *Comput. Math. Appl.*, *56*, 2145-2154.

Kumaresan, N., & Balasubramaniam, P. (2010). Singular optimal control for stochastic linear quadratic singular system using ant colony programming. *Int. J. Comput. Math.*, *87(14)*, 3311-3327.

Kumaresan, N., & Ratnavelu, K. (2014). Optimal control for stochastic linear quadratic singular neuro Takagi-Sugeno fuzzy system with singular cost using genetic programming. *Appl. Soft. Comput.*, *24*, 1136-1144.

Lagaris, I., Likas, A., & Fotiadis, D. I. (1998). Artificial neural networks for solving ordinary and partial differential equations. *IEEE T. Neural Netw.*, *9*, 987-1000.

Leite, C. R. M., Sizilio, G. R. A., Neto, A. D. D., Valentim, R. A. M., & Guerreiro, A. M. (2011). A fuzzy model for processing and monitoring vital signs in ICU patients. *Biomed. Eng. Online*, *10*, 1-17.

Lopez, S., Prieto, M., Dijkstra, J., Dhanoa, M. S., & France, J. (2004). Statistical evaluation of mathematical models for microbial growth. *Int. J. Food Microbiol.*, *96*, 289-300.

Miguel, A. M., Carlos, A. P., & Sanchez, E. (2006). A genetic-fuzzy system approach to control a model of the HIV infection dynamics. In *IEEE International Conference on Fuzzy Systems.*

Miller, W. T., Sutton, R., & Werbos, P. (1990). *Neural networks for control*. Cambridge, MA, MIT Press.

Neumeyer, K., Ross, T., & McMeekin, T. A. (1997). Development of pseudomonas predictor. *Aust. J. Dairy Technol.*, *52*, 120-122.

Nicolai, B. M., & Baerdemaeker, J. D. (1996). Chefcad: A software package for food recipe design and analysis. In *Proceedings of the Software and Food Safety Conference; Leatherhead Food RA: Leatherhead U. K.*

Oltean, M. (2005). Evolving evolutionary algorithms using linear genetic programming. *Evol. Comput.*, *13*, 387-410.

Oysal, Y., Becerikli, Y., & Konar, A. F. (2006). Modified descend curvature based fixed form fuzzy optimal control of nonlinear dynamical systems,. *Comput. Chem. Eng.*, *30*, 878-888.

Parisini, T., & Zoppoli, R. (1998). Neural aapproximation for infinite horizon optimal control of nonlinear stochastic systems. *IEEE Trans. Neural Netw.*, *9*, 1388-1408.

Pin, C., Gonzalo, F., Ordonez, J. A., & Baranyi, J. (2002). Analysing the lag-growth rate relationship of yersinia enterocolitica. *Int. J. Food Microbiol.*, *73*, 197-201.

Polycarpou, M. M. (1996). Stable adaptive neural control scheme for nonlinear systems. *IEEE Trans. Automat. Contrl.*, *41*, 447-451.

Rice, J. R. (1976). The algorithm selection problem. *Adv. Comput.*, *15*, 65-118.

Roshanfeki, M., Farah, M. H., & Rahbarian, R. (2014). A different approach of optimal control on an HIV immunology model. *Ain Shams Eng. J.*, *5*, 213-219.

Roux, O., & Fonlupt, C. (2002). Ant programming: Or how to use ants for automatic programming. In *Proceedings of ANTS00, Brussels, Belgium.*

Rovithakis, G. A., & Christodoulou, M. A. (1994). Adaptive control of unknown plants using dynamical neural networks. *IEEE Trans. Systems, Man and Cybernetics*, *24*, 400-412.

Sadegh, N. (1993). A perceptron network for functional identification and control of nonlinear systems. *IEEE Trans. Neural Networks*, *4*, 982-988.

Shirakawa, S., Ogino, S., & Nagao, T. (2011). Automatic construction of programs using dynamic ant programming,. *Ant Colony Optimization-Methods and Applications*, 75-88.

Takagi, T., & Sugeno, M. (1985). Fuzzy identification of systems and its applications to modeling and control. *IEEE Trans. Syst. Man Cyber.*, *15*, 116-132.

Tsoulos, I. G., & Lagaris, I. E. (2006). Solving differential equations with genetic programming. *Genet Program Evolvable Mach*, *7*, 33-54.

Van Impe, J. F., Poschet, F. P., Geeraerd, A. H., & Vereecken, K. M. (2005). Towards a novel class of predictive micromicr growth models. *Int. J. Food Microbiol.*, *100*, 97-105.

Van Impe, J. F., Poschet, F. P., Nicolai, B. M., & Geeraerd, A. H. (2006). S and P-type models a novel class of predictive microbial growth models. In *13th World Congress of Food Science and Technology,.*

Vincent Antony Kumar, A., & Balasubramaniam, P. (2007). Optimal control for linear singular system using genetic programming. *Appl. Math. Comput.*, *192(1)*, 78-89.

Wang, L. X. (1998). Stable and optial fuzzy control of linearsystems. *IEEE Trans. Fuzzy Syst.*, *6(1)*, 137-143.

Wijtzes, T., Riet, K. A. V., in't Veld, J., & Zwietering, M. H. (1998). A decision support system for the prediction of microbial food safety and food quality. *Int. J. Food Microbiol.*, *42*, 79-90.

Wu, S. J., Chiang, H. H., Lin, H. T., & Lee, T. T. (2005). Neural network based optimal fuzzy controller design for nonlinear systems. *Fuzzy Set. Syst.*, *154*, 182-207.

Ying, H. (1998). Sufficient conditions on uniform approximation of multivariate functions by general takagi-sugeno fuzzy systems with linear rule consequence. *IEEE Trans. Syst. Man Cyber.*, *28*, 515-521.

Zadeh, L. A. (1965). Fuzzy sets. *Inf. Control.*, *8*, 338-353.

Zarei, H., Kamyad, A. V., & Heydari, A. A. (2012). Fuzzy modeling and control of HIV infection. *Comput Math Methods Med.*, 1-17.

Zhou, Y., Liang, Y., & Wu, J. (2014). An optimal stratergy for HIV multitherapy. *J. Comput. Appl. Math.*, *263*, 326-337.

Zwietering, M., Wijtzes, T., Wit, J., & Riet, K. A. V. (1992). A decision support system for prediction of the microbial spoilage in foods. *J. Food Prot.*, *55*, 973-979.