# APPLICATION OF FRACTIONAL CALCULUS TO ENGINEERING: A NEW COMPUTATIONAL APPROACH


**By**

**RAJA MUHAMMAD ASIF ZAHOOR**
Reg. No. 10-FET/PHDEE/F-07

**A dissertation submitted to I.I.U. in partial fulfillment of the requirements for the degree of**

**DOCTOR OF PHILOSOPHY**

**Department of Electronic Engineering**
**Faculty of Engineering and Technology**
**INTERNATIONAL ISLAMIC UNIVERSITY**
**ISLAMABAD**
**2011**

DEDICATED TO

My Grandmother Maqbool Jan
Parents
Wife, Childerns Ali & Momina
Brothers and Sisters

# CERTIFICATE OF APPROVAL

Title of Thesis: Applications of Fractional Calculus to Engineering: A new Computational Approach

Name of Student: Raja Muhammad Asif Zahoor

Registration No: 10-FET/PHDEE/F-07

Accepted by the Department of Electronic Engineering, INTERNATIONAL ISLAMIC UNIVERSITY, ISLAMABAD, in partial fulfillment of the requirements for the Doctor of Philosophy Degree in **Electronics Engineering**.

**Viva Voce Commitee**

**Prof. Dr. Ghulam Yasin Chohan** _____
Dean, Facility of Engineering & Technology
International Islamic University, Islamabad

**Mr. Suheel Abdullah Malik (**Chairman**)** _____
Chairman, Department of Electronics Engineering
International Islamic University, Islamabad

**Dr. Muhammad Arif (**External Examiner-I**)** _____
Pakistan Institute of Engineering and Applied Science
Nilore, Islamabad

**Dr. Muhammad Aamir Saleem (**External Examiner-II**)** _____
Associate Professor, Associate Dean
Department of Electrical Engineering
Univerisity of Lahore, Sihala Triangle, Islamabad

**Dr. Ihsan ul Haq (**Internal Examiner**)** _____
Department of Electronics Engineering
International Islamic Univeristy, Islamabad

**Prof. Dr. Ijaz Mansoor Qureshi (**Supervisor**)** _____
Department of Electrical Engineering
Air University, Islamabad

Thursday, 22th Sep, 2011

# ABSTRACT

In this dissertation, a new heuristic computational intelligence technique has been developed for the solution for fractional order systems in engineering. These systems are provided with generic ordinary linear and nonlinear differential equations involving integer and non-integer order derivatives.

The design scheme consists of two parts, firstly, the strength of feed-forward artificial neural network (ANN) is exploited for approximate mathematical modeling and secondly, finding the optimal weights for ANN. The exponential function is used as an activation function due to availability of its fractional derivative. The linear combination of these networks defines an unsupervised error for the system. The error is reduced by selection of appropriate unknown weights, obtained by training the networks using heuristic techniques. The stochastic techniques applied are based on nature inspired heuristics like Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) algorithm. Such global search techniques are hybridized with efficient local search techniques for rapid convergence. The local optimizers used are Simulating Annealing (SA) and Pattern Search (PS) techniques.

The methodology is validated by applying to a number of linear and nonlinear fraction differential equations with known solutions. The well known nonlinear fractional system in engineering based on Riccati differential equations and Bagley-Torvik Equations are also solved with the scheme.

The comparative studies are carried out for training of weights for ANN networks with SA, PS, GA, PSO, GA hybrid with SA (GA-SA), GA hybrid with PS (GA-PS), PSO hybrid with SA (PSO-SA) and PSO hybrid with PS (PSO-PS) algorithms. It is found that the GA-SA, GA-PS, PSO-SA and PSO-PS hybrid approaches are the best stochastic optimizers. The comparison of results is made with available exact solution, approximate analytic solution and standard numerical solvers. It is found that in most of the cases the design scheme has produced the results in good agreement with state of art numerical solvers. The advantage of our approach over such solvers is that it provides the solution on continuous time inputs with finite interval instead of predefine discrete grid of inputs. The other perk up of the scheme in its simplicity of the concept, ease in use, efficiency, and effectiveness.

# LIST OF PUBLICATIONS


[1] **R. M. A. Zahoor**, J. A. Khan, and I. M. Qureshi, "Evolutionary Computation Technique for Solution of Riccati Differential Equation of Arbitrary Order," *World Acadamy Science Engineering Technology*, vol. 58, pp 303-309, 2009.

[2] **M. A. Z. Raja**, J. A. Khan, and I. M. Qureshi, "Evolutionary computational intelligence in solving the Fractional differential Equations," *Lecture notes in Computer Science*, vol. 5990, part 1, Springer-Verlog, ACIIDS Hue City, Vietnam pp 231-240, 2010.

[3] **M. A. Z. Raja**, J. A. Khan, and I. M. Qureshi, "A new Stochastic Approach for Solution of Riccati Differential Equation of Fractional Order" *Ann Math Artif Intell*., vol. 60, No. (3-4), pp. 229-250, 2010. ( ISI Impact Factor 0.893, JSR: 2009)

[4] **M. A. Z. Raja**, J. A. Khan, and I. M. Qureshi, "Solution of fractional order system of Bagley-Torvik equation using Evolutionary computational intelligence," *Mathematical Problems in Engineering*, vol. 2011, Article ID. 765075, 18 pages, 2011. (ISI-Impact factor 0.687, JSR:2010)

[5] **M. A. Z. Raja**, J. A. Khan, and I. M. Qureshi, "Heuristic computational approach using swarm intelligence in solving fractional differential equations," *GECCO* (Companion) 2010: pp. 2023-2026 2010.

[6] **R. M. A. Zahoor**, J. A. Khan, and I. M. Qureshi, "A Modified Least Mean Square Algorithm Using Fractional Derivative and its Application to System Identification" *European Journal of Scientific Research*, Vol. 35 No. 1 pp 14-21, 2009.

[7] **M. A. Z. Raja**, J. A. Khan, and I. M. Qureshi, "Swarm Intelligent optimized neural networks for solving fractional differential equations" *International Journal of Innovative Computing, Information and Control*, vol. 7, no. 11, pp. 6301-6318 Nov. 2011 ( ISI Impact factor 2.93, JSR:2009).

[8] **M. A. Z. Raja**, J. A. Khan, and I. M. Qureshi, "Swarm Intelligence Optimized neural network for solving Fractional Order Systems of Bagley-Tervik equation" Accepted for publications in *intelligent engineering systems for Electrical Engineering and Communication.*

[9] **M. A. Z. Raja,** J. A. Khan, S. I. Ahmad, and I. M. Qureshi, "Numeriacal Treatment of Painleve equation I using neural networks and stochastic solvers", Accepted as a book chapter in Studies in Computational intelligence, Springer, book series.

[10] J. A. Khan, **M. A. Z. Raja**, and I. M. Qureshi, "Stochastic Computational Approach for Complex Non-linear Ordinary Differential Equations," *Chin. Phys. Lett. v*ol. 28, no. 2, pp. 020206-09, 2011. (ISI-Impact factor 1.077, JSR:2010).

[11] J. A. Khan, **M. A. Z. Raja**, and I. M. Qureshi, "Swarm Intelligence for the problems of Non-linear differential Equations and its application to Well Known Wessinger's Equation", *European Journal of Scientific Research* vol. 34, no. 4, pp. 514-525, 2009.

[12] J. A. Khan, **M. A. Z. Raja**, and I. M. Qureshi, "Evolutionary Computing Approach for the solution of initial value Problems in ordinary Differential equations" *World Academic of Science Engineering and Technology*, vol. 55, pp 578-581, 2009.

[13] J. A. Khan, **M. A. Z. Raja**, and I. M. Qureshi, "Swarm Intelligence for the Solution of Problems in Differential Equations," icecs, pp.141-147, 2009 *Second International Conference on Environmental and Computer Science*, 2009.

[14] J. A. Khan, **M. A. Z. Raja** and I. M. Qureshi, "Hybrid Evolutionary Computational Approach: Application to van der Pol Oscillator," Accepted in International journal of physical sciences (*ISI-Impact factor 0.54, JSR: 2010*).

[15] J. A. Khan, **M. A. Z. Raja**, I. M. Qureshi, "An Application of Evolutionary computational Technique to Non-Linear Singular System arising in Polytrophic and Isothermal sphere," Accepted for publications in Numerical methods in engineering journal, 2011.

[16] J. A. Khan, **M. A. Z. Raja**, I. M. Qureshi, "Artificial Neural Network Optimized by Hybrid Computational Intelligence Approach for the Numerical Treatment of the Model of Heart," Accepted for publications in *African Journal of Bioteachnology* (ISI-Impact factor 0.556 JSR:2010).

[17] J. A. Khan, **M. A. Z. Raja** and I. M. Qureshi, "Novel Approach for van der Pol Oscillator on the continuous Time Domain", acceptedmanuscript id: 2011-0892, *Chin. Phys. Lett.* (ISI-Impact factor 1.077, JSR:2010).

[18] A. Jalil, A. Manzar, **R. A. Zahoor**, I. M. Qureshi, "Rotation-Invariant Features for Texture Image Classification," IEEE International Conference on Engineering of Intelligent System, pp. 42-45, 2006

## LIST OF SUBMISSIONS

[1] **M. A. Z. Raja**, J. A. Khan, S. I. Ahmad, and I. M. Qureshi, "Hybrid neural networks: Application to Painlevé I Transcedent," Submitted in *Chin. Phys. Lett.*

[2] **M. A. Z. Raja**, J. A. Khan, S. I. Ahmad, and I. M. Qureshi, "Stochastic numerical techniques for boundary value problem of Pantograph functional differential equations," submitted in *IEEE Trans. on Evolutionary Computations*.

[3] **M. A. Z. Raja**, S. I. Ahmad, "Stochastic numerical treatment for solving Troesh's Problem," submitted in *Journal of Computational Physics*

[4] **M. A. Z. Raja**, "Stochatic numerical techniques for solving Troesh's Problem," submitted in *Information Sciences Journal*

[5] J. A. Khan, **M. A. Z. Raja** and I. M. Qureshi, "Numerical treatment of Nonlinear Emden-Fowler Equation using Stochastic Technique," AMAI-D-11-00028, *Annals of Mathematics and Artificial Intelligence*.

[6] J. A. Khan, **M. A. Z. Raja** and I. M. Qureshi, "Numerical treatment for solving nonlinear Emden-Fowler equation using computational intelligence," NON/391934/PAP, *Nonlineartiy Journal*.

[7] J. A. Khan, **M. A. Z. Raja** and I. M. Qureshi, "Continuous Time Domain Treatment of Nonlinear Van Der Pol Oscillator via Heuristic Computational Techniques," submitted in *Mathematical Models and Method in Applied Sciences*.

The material presented in this dissertation based on first eight publications.

# ACKNOWLEDGEMENTS

*I am truly thankful to almighty Allah for his ultimate blessing and guidance at each step of this work through the blessing of Eman, health, supporting family, brilliant teachers, co-operative friends and colleagues. Peace and prayer for marvelous human and the torch bearer of wisdom Muhammad (Peace be upon him). Who's objective was enlightenment of the whole world.*

*I offer my special and sincere thanks to my supervisor Dr. Ijaz Mansoor Qureshi whose continuous and motivating guidance, beneficial suggestions and patronized supervision remained with me throughout the research work. Despite his hyper dimensional commitments, he never let me relax.*

*I owe to express my thanks for fruitful advices and critical reviews of my research work to Dr. Siraj-ul-Islam Ahmad. I found him eager to help me in my research. I am grateful to Dr. Eshan-ul-Haq for his encouragement and inspiration from time to time. I am indebted to Dr. Anwaar Manzer for his encouragement and kind support. I am also thankful to my friend Junaid Ali for his suitable suggestions and participation in long discussions throughout the research work.*

*I appreciate gratefully to my mother and father who remain desirous for my best state of Emaan and health. I am also thankful to my brothers Tariq and Atif for providing me the moral and financial support during the course of studies. I am also thankful to my sisters for their prayer and good wishes that enentually was the source of inspiration throughtout my studies. Last but not the least I thank my wife Dr. Adiqa who proved to be embodiment of endurance and sacrifice throughout untiring research years, who loves to see me flourishing and progressing. I am also thankful to my daughter Momina and son Ali whose little innocent gestures were an intangible source of inspiration to me throughout.*

*I will always be greatful to my employer organization, National Engineering and Scientific Commission NESCOM, for providing the scholarship for PhD studies.*

***(Raja Muhammad Asif Zahoor)***

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| ADM | Adomian Decomposition Mathod |
| ACO | Ant colony optimization |
| ANN | Artificial Neural Network |
| CW | Cholisky Wavelets |
| DE-NN | Differential Equation Neural Networks |
| DE | Differential Evolution |
| DSM | Direct Search Method |
| FDE | Fractional Differential Equation |
| FDE-NN | Fractional Differential Equation Neural Network |
| FOS | Fractional Order System |
| GA | Genetic Algorithm |
| GA-PS | Genetic Algorithm Hybridized with Pattern Search |
| GA-SA | Genetic Algorithm Hybridized Simulating Annealing |
| GL | Grünwald-Letnikov |
| HAM | Homotopy analysis method |
| HPM | Homotomy Perturbation Method |
| PS | Pattern Search |
| PMA | Podlubny Matrix Approach |
| PSO | Particle Swarm Optimization |
| PSO-PS | Particle Swarm Optimization Hybridized with Pattern Search |
| PSO-SA | Particle Swarm Optimization Hybridized with Simulating Annealing |
| VIM | Variational Iteration Method |

# CHAPTER 1

# INTRODUCTION

## 1.1 PROBLEM STATEMENT

Application of fractional order systems has significant importance in the field of engineering and applied science. Generally, a fractional order system is provided by a fractional differential equation, a differential equation which contains integer and non-integer order derivatives. Theory of fractional order calculus is almost as old as that of classical calculus. However, most of the extensive applications of the fractional order calculus are found to be in the last three decades. In fact, many scientific areas are currently paying attention to the fractional calculus concepts and its adoption is well known in viscoelasticity and damping, diffusion and wave propagation, electromagnetism, chaos and fractals, heat transfer, biology, electronics, signal processing, robotics, system identification, traffic systems, genetic algorithms, percolation, modeling and identification, telecommunications, chemistry, irreversibility, physics, control systems as well as economy and finance.

Linear and nonlinear ordinary differential equations have been solved with well known modern and classical numerical techniques like Runge Kutta (RK), Adomian Decomposition methods (ADM), Taylor Series methods, Homotopy analysis method (HAM), Homotopy perturbation method (HPM) and Adams method etc. Recently, the capabilities of many numerical approaches have strengthed to such extent that these are able to solve fractional differential equations. They provide the approximate analytic solution in the form of known special function with esily computable mathematical terms. Some of the famous approaches are ADM, HAM, HPM, and fractional Adam methods. However, comparatively smaller work has been carried out

for the solution of fractional order systems, which motivates the analysts and researchers of numerical computing in this field. We have investigated the heuristic approaches to provide an effective, viable and less computationally expensive solution to these fractional order systems.

The mathematical models based on Artificial Neural Network (ANN) have been used for solution of problems associated with differential equations due to its universal capability of modeling. However, the problems solved so far using ANN have differential equations with only integer order derivatives. There has been no work regarding FDEs solution with such algorithms. In this work we have developed a scheme for solving the differential equations, which contain integer, as well as, non-integer order derivatives. In this regard, the major issues which have to be addressed are given below:

- Unfortunately, the fractional derivatives for commonly used activation funcitons such as log-sigmoid are not available. So, investigation is required for the selection of activation functions for neural network.
- The formulation of problem specific objective function has to be carried out by linear combination of the neural networks that define unsupervised error such that its optimization results in an appropriate model for fractional differential equation.
- To choose appropriate methodology for training of the weights of neural networks that can optimize the problem specific objective function.
- The reliability and effectiveness of the stochastic methodologies need investigation by Monte-Carlo simulations and its statistical analysis.

## 1.2 CONTRIBUTION OF THE DISSERTATION

This dissertation presents new developed stochastic techniques for finding the solution to the fractional order systems in engineering. The designed scheme consists of approximate mathematical modeling of the fractional differential equations (FDE) using artificial neural networks that define the unsupervised error. The error is

optimized by the use of appropriate unknown weights for neural networks which are trained by the use of initial population based on computational intelligence techniques. The detail is provided as follows

1. *Development of New Neural Network Mathematical Model for Fractional Order System and its Optimization with Heuristic Computational Techniques*

   The successful mathematical models of FD equations have been constructed with the help of AN networks. In these networks exponential function was used as an activation function due to its strength in approximating any function and availability of its fractional derivative in term of easily computable terms. The accuracy of the modeling is subjected to selection of appropriate weights for the networks, which are highly stochastic by nature. Therefore, evolutionary computation algorithms, as well as, particle swarm optimization algorithm hybridized with rapid local search techniques have been used for training of weights. In this way, the proposed design scheme is formulated.

2. *The Application of Proposed Scheme to Generic Linear and Non-Linear Systems*

   Design approach has been tested by solving number of linear and nonlinear systems based on differential equations of fractional order for which the exact solution is known. It is found that the results are in a good agreement with exact solutions.

3. *The Application of Proposed Scheme to specific Systems*

   The designed scheme was also applied to well-known fractional order systems in applied sciences and engineering. These fractional order systems are known in the literature as Besset problem involving linear FD equations, Nonlinear Riccati FD equations, and Bagley Torvik linear and nonlinear FD equations. The results of the scheme are in good agreement with state of art numerical solvers.

## 1.3 ORGANIZATION OF THE DISSERTATION

Chapter 2 summarizes the overview of fractional calculus. It starts with a short survey on the history of fractional calculus and then provides the basic definitions for important relations. It also provides introduction to fractional differential equations along with their applications to engineering. A short survey on numerical treatment of fractional differential equations is also given.

Chapter 3 covers the designed methodology for modeling of fractional order systems. It also provides the formulation of fitness function for the equations and introduces the evolutionary computation and swarm intelligence algorithm. Moreover, some necessary details of algorithms are also given for further clarity.

Chapter 4 illustrates the applications of design scheme based on evolutionary computational techniques i.e. Genetic Algorithm (GA). It starts with linear differential equation of fractional order and ends up with famous engineering applications of Riccati and Bagley-Torvik equations.

Chapter 5 demonstrates some the applications of designed methodology using Particle Swarm Optimization algorithm. In this chapter, the scheme is not only applied to linear and nonlinear fractional differential equations but also the statistical analysis and comparison of the results with other techniques are provided.

Chapter 6 summarizes and concludes the dissertation. Some of the future directions and recommendations are also highlighted at the end.

# CHAPTER 2

# FRACTIONAL CALCULUS: AN OVERVIEW

In this chapter, an overview of the historical development of fractional calculus is presented. Some basic definitions of fractional integration and differentiation are given with examples of some functions. The important analytical relations are introduced to be used later in the thesis. Introduction to fractional differential equations with its important applications in engineering and technologies, and numerical treatment for the solution of differential equation of fractional order are also provided.

## 2.1 SHORT HISTORY

It is utterly wrong to categorize the fractional calculus as a young scientific field. In fact, the birth of fractional calculus takes place almost at the same time as that of classical calculus itself. The first references to the subject term of fractional order derivative were made in the $17^{th}$ century. It is interesting that today's mathematical topics related to the fractional calculus are far from being the "calculus of fractions" as one might assume by the notation itself. As an alternative, the field of fractional calculus would be understood today better as "integration and differentiation to an arbitrary order". In recent years, the available literature on fractional calculus in the form of books is published ([1], [2], [3]) and it contains a lot of information about its survey on historical development. Moreover, a number of research articles ([4], [5], [6], [7]) have addressed the various aspects of its history. Therefore, at the beginning of this document a short summary is provided for the history of fractional calculus based on the material provided in above mentioned books and articles.

The origin of classical calculus is linked with Leibniz and Newton and actually, it is also a historical controversy e. g [8], the birth of fractional calculus can be associated with Leibniz. His comments to the questions asked by L'Hospital about the meaning of non-integer (real) order derivatives, especially the case $1/2$, are the earliest remarks on the subject. His response is available in a letter to L'Hospital dated 3rd August, 1695 [9], and produces the famous statement that "one day very useful consequences will be drawn from this paradox" because there are little paradoxes without worth. Leibniz also made correspondence about the topic of non-integer order derivatives with Johann Bernoulli December 1695 and J. Wallis in 1697 ([10], [11]). The topic of non-integer order derivative continued on even after the death of Leibniz in 1716. Leonhard Euler made his comments on fractional order derivative in the paper [12] in 1783. He worked on progressions of numbers and introduced first time the generalization of factorials to Gamma function. A little more than fifty year after the death of Leibniz, J. L. Lagrange in 1772 [13] indirectly contributed to the development of exponents law for differential operators of integer order, which can be transferred to arbitrary order under certain conditions.

Perhaps, P. S. Laplace has provided the first detailed definition for fractional derivative in his book in 1812 [14]. It states that fractional derivative can be defined for functions with representation by an integral, in modern notation it can be written as $\int y(t)t^{-x}dt$. After few years, work on fractional derivative is also carried out by S. F. Lacroix. He worked on generalizing the integer order derivative of function $y(t) = t^m$ to fractional order, where $m$ is some natural number. In modern notations integer order nth derivative derived by Lacroix can be given as

$$\frac{d^n}{dt^n}t^m = \frac{m!}{(m-n)!}t^{m-n}, \quad m \geq n,$$

Generalized factorial used by Lacroix as given by Euler's Gamma function in order to formulate the fractional order derivative and gets

$$\frac{d^{\nu}}{dt^{\nu}}t^{m} = \frac{\Gamma(m+1)}{\Gamma(m-\nu+1)}t^{m-\nu}, \quad m \geq \nu,$$

Lacroix also provided the $1/2$ order derivative for function $y(t) = t$ and it is given as

$$\frac{d^{1/2}}{dt^{1/2}}t = \frac{\Gamma(2)}{\Gamma(3/2)}t^{1/2},$$

J. B. J. Fourier in 1822 provides more applicable definition for fractional calculus in his book [15]. Fractional calculus was talked about right from the commencing of classical calculus, but in 1823 first time fractional operations were used in solving of specific physical application, named as tautochrone problem. It was Niels Henrik Abel who describes the problem along with its solution in detail in 1823.

After Abel's integral equations, applications in physics, the first detailed study on fractional calculus were provided by J. Liouville ([16], [17], [18]). His definitions for fractional derivative for a function are given in the form of an infinite series with limitations to its convergence and applicability to limited class of functions only. In many applications to physical, geometrical and mechanical problems, Liouville definition of fractional derivative is used. Riemann used complementary function for defining of fractional derivative. In 1847 G. F. B. Riemann carried out his work for a generalization of a Taylor series, and deduced his definition for fractional derivative.

In such a way, the commonly used definition for fractional integration has been developed. It is called as Riemann-Liouville fractional integer (see also section 2.3.4). It is worth mentioning that Riemann could not develop the representation without involving problematic complementary function, as pointed out by A. Cayley in [19]. However, it is noted that the collective work of Riemann were printed about ten years after his death.

Number of other definitions for fractional derivative or integrals is available in the literature. To name a few, Grünwald-Letnikov provided the fractional derivative by use of finite differences. H. Weyl [20] in 1917 has introduced fractional integral by use of periodic functions and their Fourier transform. Marchaud [21] in 1927 has

given the integral version for the fractional derivative of Grünwald-Letnikov. Leibniz's formula for analytic functions has been developed for fractional derivative of Riemann-Liouville type by Watanabe [22] in 1931. M. Riesz ([23], [24]) has developed the integral known as Riesz Potential, closely related to Riemann-Liouville and Weyl fractional integral. B. S. Nagy [25] in 1940 has developed an inequality for trigonometric sums and approximated the fractional integrals by trigonometric polynomials. Erdélyi and Kober in ([26], [27]) have introduced the modification in Riemann Liouville fractional integral by using Mellin Transform. A large number of additional results on fractional calculus were reported in the twentieth century, but it is worth mentioning that the definition of fractional derivative provided by M. Caputo in 1960 is the most frequently used in the applications.

The field of fractional calculus has developed to such an extent by the second half of twentieth century that the first conference exclusively on the theory and application of fractional calculus was held in 1974 at New Haven [28]. The first book on fractional calculus by Oldham and Spanier [29] appeared in 1974. The first issue of journal related solely to the fractional calculus "Fractional Calculus & Applied Analysis" was available in 1998. In recent history a number of books and survey papers have been published on the subject of fractional calculus. The most popular in books in this regard by Miller and Ross [1], Samko et al. [2] and Podlubny [30]. Furthermore application specific books in fractional dynamics are also published like Barlow [31], Ben-Avraham and Havlin [32], Carpinteri and Mainardi [33], Metzler and Klafter [34], Ozaktas et al [35], Monje et al [36] and West et al [37] etc. Beside those a large number of special issues on fractional calculus and its applications to the physical systems are also available in the reputed journals ([38], [39], [40]). Moreover, a number of special issues of journals have been launched recently exclusively for fractional order calculus.

In short, fractional calculus from its birth till today, has found wide applications for many complex problems in the field of science and engineering.

## 2.2 BASIC DEFINITIONS

In this section, basic theory of some special functions and definitions of fractional integral and derivative is given, which will be used in the next chapters. The fractional integral and derivative have been expressed in the literature in a variety of ways, including Riemann-Liouville, Caputo, Erdélyi-Kober, Hadamard, Grünwald-Letnikov and Riesz type etc. Equivalence of these definitions for some function has been given in standard fractional calculus reference books ([1], [2], [29]). All these definitions have their own importance and advantages in different types of mathematical problems. Riemann-Liouville and Caputo definitions of fractional derivative are used in our study. Firstly we introduce the Gamma and Mittag-leffler functions, and then other definitions are presented which play significant role to develop a theory for differential of arbitrary order, as well as, in the theory of FDE equations.

### 2.2.1 GAMMA FUNCTION

Probably, one of the fundamental functions of the fractional calculus is Euler's Gamma function, which is basically provide the generalization of factorial, $n!$ and permits $n$ to take as real and even complex values.

The definition of the gamma function is given by the integral [30]

$$\Gamma(t) = \int_0^\infty e^{-x} x^{t-1} dx \tag{2.2.1}$$

The gamma function can also be defined by the limit representation as

$$\Gamma(t) = \lim_{n\to\infty} \frac{n! n^x}{x(x+1)\cdots(x+n)}, \quad \mathrm{Re}(x) > 0. \tag{2.2.2}$$

Two basic properties of gamma function are given as

$$\Gamma(t+1) = t\Gamma(t) = n(n-1)! = n!, \tag{2.2.3}$$

and the second important property is that it has simple pole at the points $x = -n, (n = 0,1,2,...,)$.

### 2.2.2 MITTAG-LEFFLER FUNCTION

G. M. Mittag-Leffler has developed a function, for one-parameter generalization of exponential function, which is represented by [41]

$$E_{\alpha}(x)=\sum_{n=0}^{\infty}\frac{x^{n}}{\Gamma(\alpha x+1)}. \tag{2.2.4}$$

It was introduced in [42] and also studied by A. Wiman [43]. Mittag-leffler function (MLF) of two parameters was in fact developed by Agarwal [44] using series expansion [41] and is defined as

$$E_{\alpha,\beta}(x)=\sum_{n=0}^{\infty}\frac{x^{n}}{\Gamma(\alpha x+\beta)},\quad (\alpha>0,\beta>0). \tag{2.2.5}$$

For $\beta=1,$ the above relation is reduced to MLF function in one parameter. It can be obtained from the definition (2.2.5) that

$$E_{1,1}(x)=e^{x},\quad E_{1,2}(x)=\frac{e^{x}-1}{x},\quad E_{1,3}(x)=\frac{e^{x}-1-x}{x^{2}}, \tag{2.2.6}$$

The particular cases of MLF can be represented by hyperbolic sine and cosine functions

$$E_{2,1}(x^{2})=\cosh(x),\quad E_{2,2}(x^{2})=\frac{\sinh(x)}{x}. \tag{2.2.7}$$

Similarly the relation of MLF with error function is given as

$$E_{1/2,1}(x)=\sum_{n=0}^{\infty}\frac{x^{n}}{\Gamma(\frac{n}{2}+1)}=e^{x^{2}}erfc(-x), \tag{2.2.8}$$

where the error function is defined by

$$erfc(x)=\frac{2}{\sqrt{\pi}}\int_{x}^{\infty}e^{-x^{2}}dx. \tag{2.2.9}$$

Another important particular case of the MLF used extensively in solving fractional order systems is given as

$$\varepsilon_x(\nu, a) = t^{\nu} \sum_{n=0}^{\infty} \frac{(ax)^n}{\Gamma(\nu + n + 1)} = t^{\nu} E_{1,\nu+1}(ax), \tag{2.2.10}$$

where $\nu$ is representing a fraction and $a$ is a constant.

### 2.2.3 RIEMANN LIOUVILLE FRACTIONAL INTEGRAL AND DERIVATIVE

The definition of Riemann-Liouville fractional integral of order $\nu < 0$ is given as

$$(I^{\nu} f)(t) = \frac{1}{\Gamma(\alpha)} \int_0^t (t-\tau)^{\nu-1} f(\tau) d\tau, \qquad (I^0 f)(t) = f(t) \tag{2.2.11}$$

where $I^{\nu}$ is the $\nu$ order fractional integral. Accordingly, the fractional derivative of order $\nu > 0$ is normally given as

$$(D^{\nu} f)(t) = (\frac{d}{dt})^n (I^{n-\nu} f)(t) \quad (n-1 < \nu \le n), \tag{2.2.12}$$

where $D^{\nu}$ is the fractional derivative and $n$ is an integer. One of the important properties associated with it is that the Riemann Liouville fractional derivative is an inverse of the integral of the same order.

Let us apply the definition of Riemann Liouville fractional derivative to power function given as

$$f(t) = (t-a)^{\alpha}, \tag{2.2.13}$$

where $\alpha$ is an arbitrary real number and $a$ is some constant. It can be evaluated and provided as [30]

$$D^{\nu}(t-a)^{\alpha} = \frac{\Gamma(1+\alpha)}{\Gamma(1+\alpha+\nu)}(t-a)^{\alpha-\nu}, \tag{2.2.14}$$

Similarly, in case of exponential function, $e^{\lambda t}$, it can be evaluated as

$$D^{\nu}(e^{\lambda t}) = t^{-\nu} E_{1,1-\nu}(\lambda). \tag{2.2.15}$$

### 2.2.4 CAPUTO FRACTIONAL DERIVATIVE

The definition of derivative provided by Riemann-Liouville has certain limitations when it is used for modeling of real-world phenomena associated with fractional differential equations. Therefore, we introduce a modified definition of fractional differential operator $D^{\nu}$ given by Caputo [45], [46].

$$(D^{\nu} f)(t) = I^{n-\nu} \frac{d^n}{dt^n} f(t) = \frac{1}{\Gamma(n-\nu)} \int_0^t (t-\tau)^{n-\nu-1} f^{(n)}(\tau) d\tau \quad (n-1 < \nu \le n), \tag{2.2.16}$$

where $I^{\nu}$ is given in (2.2.11). The usual property of the Caputo integral operator is

$$(I^{\nu} D^{\nu} f)(t) = f(t) - \sum_{k=0}^{n-1} f^{k}(0) \frac{t^k}{k!} \quad (n-1 < \nu \le n), \tag{2.2.17}$$

The Caputo fractional derivative is obtained by computing an ordinary derivative followed by the fractional integral, whereas the Riemann-Liouville is obtained in the reverse order. The use of Caputo fractional derivative allows the traditional homogeneous, as well as, inhomogeneous initial and boundary conditions occurring often in general application. However, both the formulations, Riemann Liouville and Caputo, coincide for homogeneous initial conditions.

Laplace transform operator is used to understand the difference between Riemann-Liouville and Caputo derivatives in terms of initial conditions accompanied by fractional differential equations. The formula for Laplace transform of Riemann-Liouville fractional derivative contains the initial condition with limiting values for its fractional derivative. Problems with such type of initial conditions are solved successfully [47], however its solution is useless, because there is no physical interpretation for such initial conditions. On the contrary, such problems do not arise by taking Laplace transform of Caputo fractional derivative.

### 2.2.5 GRÜNWALD-LETNIKOV FRACTIONAL DERIVATIVE

In this section, we describe an approach of Grünwald-Letnikov fractional derivative by taking a continuous function $y = f(t)$. Using the definition of first order derivative

$$\frac{d}{dt} f(t) = \lim_{h \to 0} \frac{f(t) - f(t-h)}{h}$$

The n[th] order derivative can be represented as

$$\frac{d^n}{dt^n} f(t) = \lim_{h \to 0} \frac{1}{h^n} \sum_{m=0}^{n} (-1)^m \binom{n}{m} f(t - mh), \tag{2.2.18}$$

where, in usual notation of binomial coefficient

$$\binom{n}{m} = \frac{n(n-1)(n-2)\ldots(n-m+1)}{m!}$$

Now using the arbitrary value of order $\nu$ the equation (2.2.18) can be written as

$$D^{\nu} f(t) = \lim_{\substack{h \to 0 \\ nh = t}} h^{-p} \sum_{m=0}^{n} (-1)^{\nu} \binom{\nu}{m} f(t - mh), \tag{2.2.19}$$

which represents the derivative of order $r$ if $\nu = r$ and the r-fold integral if $\nu = -r$. By assumption the $y = f(t)$ and its integer order derivative are continuous in the closed interval $[a,t]$ and $r$ is an integer number satisfying the condition $r > \nu - 1$, then the formula (2.2.18) can be formulated as

$$D^{\nu} f(t) = \sum_{k=0}^{r} \frac{(t-a)^{-\nu+k}}{\Gamma(-\nu+k+1)} \frac{d^k}{dt^k} f(a) + \frac{1}{\Gamma(-\nu+k+1)} \int_a^t (t-\tau)^{r-\nu} \frac{d^{r+1}}{dt^{r+1}} f(\tau) d\tau. \tag{2.2.20}$$

From the mathematical perspective, the class of the functions for which this definition is applicable, is very narrow due to constraints on function for $(r+1)$-time continuously differential functions. Moreover, understanding this fact is important in order to apply for the methods of the fractional calculus in applications, especially

because of the fact that the Riemann-Liouville definition weakens the conditions on the function $y = f(t)$ and requires the integrable condition on $y = f(t)$ only.

### 2.2.6 OTHER DEFINITIONS

In this section, we have presented the other definitions of fractional integrals and derivatives used in the literature. All these definitions have their own advantages and limitations. Let us start with the definition of Erdélyi-Kober type fractional integrals and derivative. Let us take $(a,b)$ as a finite or infinite interval on $\Re^+$, also let $\Re(\nu) > 0, \sigma > 0,$ and $\eta$ be the complex number, then for $a = -\infty$ and $b = \infty$, the Erdélyi-Kober type right-sided fractional integral is defined by

$$I_{-}^{\nu} f(t) = \frac{\sigma t^{\sigma\eta}}{\Gamma(\nu)} \int_{t}^{\infty} \frac{\tau^{\sigma(1-\nu-\eta)-1} f(\tau) d\tau}{(\tau^{\sigma} - t^{\sigma})^{(1-\nu)}} \quad (t > 0), \tag{2.2.21}$$

and similarly the right side fractional derivative corresponding to Erdélyi-Kober type fractional derivative is defined as

$$D_{-}^{\nu} f(t) = t^{-\sigma\eta} (-\frac{1}{\sigma x^{\sigma-1}} D)^{n} t^{\sigma(n-\eta-\nu)} (I_{-}^{n-\nu} f(t)), \tag{2.2.22}$$

where $n = \Re(\nu) + 1$ with $D = d/dt$. Such types of definitions are given in Samko et al [2], and Butzer et al [48].

The left-sided fractional integral and derivative, respectively, introduced by Hadamard [49] can be defined as

$$I_{+}^{\nu} f(t) = \frac{1}{\Gamma(\nu)} \int_{0}^{t} (\log \frac{t}{\tau})^{\nu-1} \frac{f(\tau) d\tau}{\tau} \quad (t > 0), \tag{2.2.23}$$

$$D_{+}^{\nu} f(t) = (t \frac{d}{dx})^{n} \frac{1}{\Gamma(n-\nu)} \int_{0}^{t} (\log \frac{t}{\tau})^{n-\nu+1} \frac{f(\tau) d\tau}{\tau} \quad (t > 0). \tag{2.2.24}$$

There are many other definitions available in the literature, but we will briefly provide the definitions mostly used in engineering applications like Riesz Fractional Integro-

Differentiation. The definition of Riesz fractional integration is realized in the form of Riesz potential and given as the Fourier convolution form

$$(I^{\nu} f)(\mathbf{t}) = \int_{R^n} k_{\nu}(\mathbf{t}-\boldsymbol{\tau}) f(\boldsymbol{\tau}) d\boldsymbol{\tau} \quad (\Re(\nu) > 0), \tag{2.2.25}$$

where $\Re^n$ be the n-dimensional Euclidean space, sufficiently good functions $f(\mathbf{t}) = f(t_1, t_2, ..., t_n)$ and Riesz kernel function $k_{\nu}(\mathbf{t})$ is given as

$$k_{\nu}(\mathbf{t}) \approx \frac{1}{\gamma_n(\nu)} \begin{cases} |\mathbf{t}|^{\nu-n}, & \nu - n \neq 0,2,4,...., \\ |\mathbf{t}|^{\nu-n} \log(\frac{1}{|\mathbf{t}|}), & \nu - n = 0,2,4,...., \end{cases} \tag{2.2.26}$$

where the constant $\gamma_n(\nu)$ can be written as

$$\gamma_n(\nu) \approx \begin{cases} 2^{\nu} \pi^{n/2} \Gamma(\nu/2)[\Gamma(n-\nu)/2]^{-1}, & \nu - n \neq 0,2,4,...., \\ (-1)^{(n-\nu)/2} 2^{\nu-1} \pi^{n/2} \Gamma(1+(\nu-n)/2)\Gamma(\nu/2), & \nu - n = 0,2,4,...., \end{cases} \tag{2.2.27}$$

The equivalence of all these definition applied on certain function can be easily verified and is given in standard reference books of fractional calculus. However, it is necessary to mention that we have confined our studies to Riemann-Liouvelli, Caputo and Grünwald-Letnikov fractional derivative formulations.

## 2.3 FRACTIONAL DIFFERENTIAL EQUATIONS

In order to abbreviate the Ordinary differential equations involving arbitrary non-integer order derivatives the notation of fractional differential equation is used in the literature. The interest of the researcher for FDE equations as a separate topic has arisen some 40-50 years ago. It is mainly due to its ability to make mathematical models of complex phenomena with ease. These equations can hold non-local relationships in space and time through power-law memory kernels. Moreover, since FDEs have extensive applications in engineering and science, research in such areas has developed significant potential all around the world.

The solution for Cauchy type problems involving the Riemann-Liouville differential operators has attracted several researchers specially the mathematicians. The

problems involving this kind of FDE equations have been systematically and rigorously studied and it can be found in the Ref. ([50], [51], [52]) for review. Besides, many applied researchers have used FDE equations models of real process however the literature on the subject does not seem to provide the results on far-reaching consequences. It is due to the fact that for fractional derivative operators many useful properties are unknown or not fully developed, such as, a clear physical and geometrical interpretation, chain rules, product rules, and so on.

The strength of classical calculus is well known to provide a powerful tool for describing and modeling of important dynamic processes in wide fields of the applied sciences. However, there are many complex systems in nature involving anomalous dynamics and have a macroscopic complicated behaviors, for which the dynamics cannot be characterized fully by classical calculus such as pollution diffusion in atmosphere, the transportation studies for chemical contaminants in water around rocks, the dynamics of polymers a kind of viscoelastic materials, cellular diffusion practices, network traffic, the effect on the profitability in financial markets stocks through speculations, transmission of signals during strong magnetic and many more.

Recently, fractional calculus operators have been utilized rather extensively for finding particular solutions of homogeneous as well as non-homogeneous linear ordinary and partial differential equations associated with many celebrated equations of mathematical physics such as the Gauss hypergeometric equation and Bessel equation. The ordinary and partial differential equations of higher orders, and many extensions and relatives have presented large number of scattered results in a unified manner [53]. Few years ago many interesting applications of so called Fractional Fourier Transform (FFT) have been published. This FFT transform has been widely applied mainly in Optics and Signal Processing theory. The excellent book by Ozaktas et al [35] is a good introduction in this respect (see also, [54], [55], [56], [57])

## 2.4 APPLICATION OF FRACTIONAL CALCULUS IN ENGINEERING

In this section, a short survey has been provided on applications of fractional order systems. In recent years, it has been a productive field for research in engineering and applied sciences. Fractional calculus concepts are used in many scientific areas of applied sciences and and its acceptance is well known in electromagnetism, viscoelasticity and damping, diffusion and wave propagation, chaos and fractals, heat transfer, biology, electronics, signal processing, robotics, system identification, traffic systems, genetic algorithms, percolation, modeling and identification, telecommunications, chemistry, irreversibility, physics, control systems as well as social sciences like economy and finance. However, we have presented here a brief overview on few of them.

### 2.4.1 VISCOELASTICITY

Let us start with the Hooke's law about the relationships between stress and strain for solids and for Newtonian fluids, which are as

$$\sigma(t) = E\varepsilon(t), \qquad \sigma(t) = \eta \frac{d\varepsilon(t)}{dt}, \tag{2.4.1}$$

where $\eta$ and $E$ are constants. These mathematical models are applicable for ideal solid material and for an ideal fluid. In-fact, in real word neither of such materials exists. Its real material properties lie between ideal solid and ideal fluids on the basis of sorting of materials on its firmness. Moreover, it can be stated that the zero$^{th}$ and first order derivative of strain is proportional to the stress for solid and stress for fluid, respectively. Naturally, a question arises, in ref [58], that whether "intermediate" material stress is proportional to "intermediate" strain derivative (non-integer).

The models linear viscoelesticity has been developed using integer-order derivatives and it is provided in book [30]. Hooke's elastic element is shown as a spring, while the Newton viscous element is represented as a dashpot. Similarly, generic linear

model of viscoelasticity can be derived from by replacing integer-order derivative with fractional derivatives. It can be given as

$$\sum_{i=0}^{n} a_i D^{\nu_i} \sigma(t) = \sum_{i=0}^{m} b_i D^{\mu_i} \varepsilon(t), \qquad 2.4.2$$

where $a$ and $b$ are constant coefficient, $\nu$ and $\mu$ are non-integer order of differential equations. It is the generalization of basic laws of deformation of viscoelesatic materials in term of complicated differential equations, but it is used in formulating and solving many applied problems. Moreover, it is possible to achieve the best results for $n = m$ and $\nu_i = \mu_i, (i = 0,1,2,...,)$. It is worth mentioning that many authors made significant contributions to the development of fractional-order models of viscoeleasticity and their applications like M. Caputo ( [60], [61], [62] ), A. Fenauder [63], Ch. Friedrich ( [64], [65] ), B. Gross [66], F. Mainardi ( [67], [68] ) and others. However, today's intensive development of this field and its advanced state comparing to other fields is undoubtedly due to series of works by R. L. Bagley and Torvik ( [69], [70], [71] ) in which the advantages of fractional calculus approach were presented with ultimate clarity using both theoretical and experimental arguments, and also due to the need for a better description of the properties of materials used in industry.

### 2.4.2 BODE'S ANALYSIS OF FEEDBACK AMPLIFIERS

Studies on feedback amplifier design [72] were first published by H. W. Bode in 1945. He has taken the system characterized with the frequency response given by

$$Z(\omega) = \frac{a}{b}(i\omega)^{-n} \qquad (2.4.3)$$

and its corresponding transfer function

$$g_z(s) = \frac{a}{b} s^{-n}, \qquad (2.4.4)$$

where $a$ and $b$ are constant, and $n$ is the number representing stages for feedback amplifier. In his investigation Bode takes $n$ as an arbitrary real number and found that optimal number of stages is non-integer in a feedback amplifier. In short, Bode carried out a frequency domain analysis for the performance of non-integer order system with fractional order transfer function as given in equation (2.4.4).

### 2.4.3 FRACTIONAL LEAST MEAN SQUARE ALGORITHM

In Least Mean Sqare (LMS) in the literature summarized by the following equations [73]

$$\begin{aligned} y(n) &= \hat{\mathbf{w}}^H(n)\mathbf{u}(n) \\ e(n) &= d(n) - y(n) \\ \hat{\mathbf{w}}(n+1) &= \hat{\mathbf{w}}(n) + \mu e(n)\mathbf{u}(n) \end{aligned} \tag{2.4.5}$$

Where $y(n)$ is filter response, $d(n)$ is the desired response, $e(n)$ is the error signal, $u(n)$ is input signal, $\hat{w}(n)$ is tape weights and $\mu$ is step size. In deriving the fractional LMS algorithm, we have to use fractional derivatives in addition to first derivative. The weight adaptation equation of k$^{th}$ element for FLMS algorithm can be written by

$$\hat{w}_k(n) = \hat{w}_k(n-1) + \mu_1 e(n)u(n-k) + \mu_2 e(n)u(n-k)\frac{\hat{w}_k^{1-\nu}(n)}{\Gamma(2-\nu)} \tag{2.4.6}$$

where $\nu$ is the real number, $0 < \nu < 1$.

To speed up the tedious computations of the non-linear equation in cyclic single step iteration, assuming $\hat{w}_k^{1-\nu}(n) \cong \hat{w}_k^{1-\nu}(n-1)$ then

$$\hat{w}_k(n) \cong \hat{w}_k(n-1) + \mu_1 e(n)u(n-k) + \mu_2 e(n)u(n-k)\frac{\hat{w}_k^{1-\nu}(n-1)}{\Gamma(2-\nu)} \tag{2.4.7}$$

for $k = 0, 1, 2, \ldots, M-1$, where $M$ is the number of taps.

The FLMS algorithm is formulated by using eqation (2.4.5), and weights updation using equation (2.4.7).

It can be used in digital signal problems extensively like as standard system identification problem [73]. The desire response $d(n)$ for for the input vector $\boldsymbol{u}(n)$, of the unknown system is given as

$$d(n) = \boldsymbol{w}_0^H(n)\boldsymbol{u}(n) + \eta(n) \tag{2.4.8}$$

where the tap-weight vector $\boldsymbol{w}_0(n)$ represents the "target" to be tracked by the filter, and $\eta(n)$ is unknown white Gaussian noise with zero mean. The tap-weight vector $\hat{\boldsymbol{w}}(n)$ is estimated using standard LMS and FLMS algorithm [74] are given in Table 2.1.

**Table 2.1** Comparison of Convergence Analysis of LMS & FLMS Algorithms

| **Step size** | **Algorithm Applied** | **Order of derivative** | **Number of iterations** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | *5 taps with* $\Delta w$ | | | *7 taps with* $\Delta w$ | | | *9 taps with* $\Delta w$ | | |
| | | | 0.5 | 0.1 | 0.004 | 0.5 | 0.1 | 0.004 | 0.5 | 0.1 | 0.004 |
| 0.0005 | LMS | 1 | 239 | 651 | 1000+ | 345 | 681 | 1000+ | 355 | 732 | 1000+ |
| | FLMS | 0.5 | 91 | 193 | 459 | 104 | 206 | 445 | 149 | 271 | 524 |
| 0.001 | LMS | 1 | 106 | 302 | 719 | 157 | 365 | 698 | 204 | 358 | 745 |
| | FLMS | 0.5 | 55 | 101 | 201 | 61 | 132 | 286 | 68 | 209 | 479 |
| 0.0015 | LMS | 1 | 79 | 196 | 469 | 79 | 185 | 472 | 89 | 252 | 474 |
| | FLMS | 0.5 | 22 | 58 | 114 | 30 | 76 | 175 | 39 | 140 | 291 |

It can be inferred from the results shown in Table 2.1 that the rate of convergence of the FLMS algorithm is superior to that of conventional LMS algorithm

### 2.4.4 RICCATI DIFFERENTIAL EQUATION

In this section, an introductory literature about Riccati fractional order system provided by differential equations of arbitrary order is presented. Its historical background, introduction to the subject terms, and fundamental theories can be found in books ( [75], [76] ). The Riccati differential equations has substantial importance in classical, as well as, modern science and engineering applications, like stochastic realization theory, random processes, optimal filtering, controls, robust stabilization,

calculus of variations, network synthesis, diffusion problems and areas such as financial mathematics ( [75], [76], [77], [78], [79] ).

The generic non-linear quadratic Riccati differential equation can be written as

$$\frac{d^{\nu} y(t)}{dt^{\nu}} = p(t) + q(t)y(t) + r(t)y^2(t), \quad 0 < t \le T, \tag{2.4.9}$$

with initial and boundary conditions given as $y(0) = 0, \quad y'(0) = 0,$

where $\nu$ is the order of the equations, $p(t), q(t)$, and $r(t)$ are known functions, $T$ is the constant used for representing the span of inputs within the close interval $[0, T]$.

The general response equation (2.4.9) contains a parameter $\nu$, the order of the fractional derivative that can be varied to obtain various responses. In the case of $\nu = n$, where $n$ is some integer, the equation (2.4.9) is reduced to the classical Riccati differential equation. It has significant role in many fields of science and technology. For example, it is well-known that the one-dimensional static Schrödinger equation is closely related to a Riccati differential equation [80]. Solitary wave solution of a nonlinear partial differential equation can be represented as a polynomial in two elementary functions satisfying a projective Riccati equation [81].

In case the order $\nu$ of equation (2.4.9) is non-integer, then it is a Riccati differential equation of fractional order. The value of order $\nu = 0.5$ has a special importance. This is because in classical fractional calculus many of the model equations have been developed by using this particular order of the derivative [82]. However in modern applications much more generic values of the order $\nu$ appear in the equations ( [83], [84] ). Therefore, the problem to develop the analytical and numerical method to solve the Riccati differential equation of arbitrary order, has attracted much attention and has been studied by many authors. In this regard an approximate analytical solution of the equation is derived using piecewise variation iteration method [85]. Recently other numerical methods have been extended to solve such equation, e. g. Adomian decomposition method [86], He's variational iteration method [87], and Fractional Adams-Moulton method [88].

### 2.4.5 BAGLEY-TORVIK EQUATION

The Bagley-Torvik equation is originally formulated in the studies on behavior of real materials by use of fractional calculus ([70], [88]). It has raised its importance since then in many engineering and applied sciences applications. In particular, the equation with 1/2-order derivative or 3/2-order derivative can model the frequency-dependent damping materials quite satisfactorily. It can also describe motion of real physical systems, the modeling of the motion of a rigid plate immersed in a Newtonian fluid and a gas in a fluid, respectively ([30], [90]). Fractional dynamic systems have found many applications in various problems such as viscoelasticity, heat conduction, electrode–electrolyte polarization, electromagnetic waves, diffusion wave, control theory, signal processing and so on ([69], [71], [91], [92], [93], [94], [95]).

The generic form of Bagley-Torvik equation can be written as

$$A\frac{d^2y(t)}{dt^2} + B\frac{d^{3/2}y(t)}{dt^{3/2}} + C[y(t)]^n = f(t), \quad 0 < t \le T \tag{2.4.10}$$

with initial and boundary conditions given as $y(0) = 0, \quad y'(0) = 0,$

where $n$ is the nonlinear operator of the equations, $A, B$ and $C$ are constant coefficients, and $T$ is the constant representing the span of inputs within the close interval $[0, T]$.

The general response expression (2.4.10) contains parameters that can be varied to obtain various responses. In the case of $n = 1, A = M,$ the mass of thin rigid plate, $C = K,$ the stiffness of the spring, $B = 2S\sqrt{\mu\rho}$ where $S$ is area of plate immersed in Newtonian fluid, $\mu$ is viscosity and $\rho$ is the fluid density, then the equation (2.4.10) represent the motion of a large thin plate in a Newtonian fluid [30]. Similarly, linearly damped fractional oscillators with the damping term having a fractional derivative of order $\nu = 1.5,$ it can be represented by Bagley-Torvik equation ([30], [96]).

The problem to develop the numerical solvers to find the solution of Bagley-Torvik fractional differential equation has attracted much attention and has been studied by

many authors. In this regard, an approximate analytical solution of the equation was derived using Adomian decomposition method ([97], [98]). He's variational iteration method [99], Taylor collocation method [100]. Diethelm transformed the equation into first order coupled fractional differential equation and solved the problem with Adams predictor and corrector approach [101]. Podlubny used successive approximation method to solve the equation, and recently applied the Matrix Approach to Discretization of Fractional Derivatives for same problem ([30], [102]). However, there has been no advancement to apply the stochastic numerical solvers to find the solution of the equation.

### 2.4.6 FRACTIONAL-ORDER CHUA-HARTLEY SYSTEM

The classical Chua circuit is presented by the following non-linear system of integer order differential equations

$$\begin{aligned} \frac{dx(t)}{dt} &= -a_1 x + a_1 y - b_1 f(t), \\ \frac{dy(t)}{dt} &= a_2 x - a_2 y - b_2 z, \\ \frac{dx(t)}{dt} &= -a_3 y, \end{aligned} \tag{2.4.11}$$

where *f(t)* is a characteristic of piecewise-linear resistor and

$$a_1 = \frac{1}{RC_1}, \quad a_2 = \frac{1}{rC_2}, \quad b_1 = \frac{1}{C_1}, \quad b_2 = \frac{1}{RC_2}.$$

Chua's circuit is involved extensively in study of chaos, multi-stable behavior, and bifurcations. There are a number of article available on the subject and it is one of the most widely considered circuit today [103]. Chua's circuit can be depicted by the closed-loop control systems. It has Chua's resistor in its feedback with

$$G_c(s) = \frac{1}{s}, \quad G_s(s) = \frac{a_1(s^2 + a_2 s + a_3 b_2)}{s^2 + a_2 s + a_3 b_2 - a_1 a_2}. \tag{2.4.12}$$

The system provided in Eq. (2.4.11) can be described by the following system of differential equations involving three equations with one containing two fractional derivatives [104]:

$$\begin{aligned} D^{\nu}x(t) &= \alpha D^{\nu-1}(y(t)-x(t)) - \frac{2a}{7}(4x(t)-x^3(t)), \\ \frac{dy(t)}{dt} &= x(t)-y(t)-z(t), \\ \frac{dz(t)}{dt} &= -\frac{100}{7}y(t), \end{aligned} \tag{2.4.13}$$

by taking the values of coefficients as

$$a_1 = \alpha, \quad a_2 = 1, \quad a_3 = \frac{100}{7}, \quad b_1 = \frac{2\alpha}{7}, \quad b_2 = 1.$$

### 2.4.7 FRACTIONAL DIFFUSION EQUATIONS

One of the most important applications for fractional order derivatives is the modeling of diffusion in a particular type of porous medium in fractal media. The resultant equation is related to the fractal dimension of the porous material. The transfer processes in fractal are described with the following equation suggested by many authors [105]:

$$D^{1/d-1}J(t) = LX(t), \tag{2.4.14}$$

where $J(t)$ is macroscopic flow across the fractal interface, $X(t)$ is the driving force, $L$ is a constant, and $d$ is the fractal dimension. It is significant that the fractional diffusion equation has been related to a dynamical process in fractal dimension of the fractal, which serves as a model of a porous material. The fractional order diffusion equation suggested by R. Metzler et al [106] is an example of second type of fractional diffusion equation given by

$$D^{2/d_w}P(r,t) = \frac{1}{r^{d_s-1}}\frac{\partial}{\partial r}\left(r^{d_s-1}\frac{\partial P(r,t)}{\partial r}\right), \tag{2.4.15}$$

where $d_w$ and $d_s$ depend on the fractal dimension of the media. Moreover, another example of the second type is the fractional diffusion equation in the form deduced by R. R. Nigmatullin ([107], [108]). In simple case of spatially one-dimensional diffusion, Nigmatullin's equation is given as

$$D^{\nu}u(x,t) = \frac{d^2u(x,t)}{dx^2}. \qquad 2.4.16$$

Since the order $\nu$ of the derivative in the above equation can be of arbitrary real order, including $\nu = 1$ and $\nu = 2$, it is known as fractional diffusion wave equation. Now for $\nu = 1$ the Eq. (2.4.12) becomes classical diffusion equations, and for $\nu = 2$ it is known as classical wave equation. For $0 < \nu < 1$ and $0 < \nu < 2$ the above equation is so-called ultraslow diffusion and intermediate processes, respectively ([109], [110]). The fractional diffusion-wave equation was intensively used in application by F. Mainardi ([68], [110], [111]), A. N. Kochubei [112] and A. M. A. El-Sayed [113].

### 2.4.8 OTHER APPLICATIONS

It is well known that a close connection exists between fractional differential equations and the dynamics of many complex systems, like anomalous processes, fractional Brownian motion, continuous time random walk method, fractal media and many others. Therefore, it is not possible to provide the complete list of its application. However, some important references are mentioned for selected subject terms: Probability and Statistical modeling ([114], [115]); Fractional controls in Robotics ([116], [117], [118]); Fractional Capacitor Theory ([119], [120]); Mechanical system and its analysis ([121], [122], [123]); Electro-analytic Chemistry ([124], [125]); Fractional order physics ([126], [127]); Electronics and Electromagnetism ([128], [129]); Control Theory ([130], [131], [132]); Fractional Multi pole [133]; Intelligent systems for transportation ([134], [135]); Oscillators [136]; Thermal systems ([137], [138]); Living systems [139]; Electrolyte processes [140]; Economics ([141], [142]); chaos and fractals ([143], [144]).

## 2.5 NUMERICAL TREATMENT OF FRACTIONAL DIFFERENTIAL EQUATIONS

In this section, a brief description on some numerical methods has been given for the solution of differential equations of fractional order. It starts with brief introduction and brief historical survey of numerical methods. A brief description is also provided for the numerical methods used later in the thesis for comparison of the results.

### 2.5.1 INTRODUCTION

The numerical methods have been developed successfully by many researchers for linear, nonlinear, ordinary and partial differential equations of fractional order. The numerical methods are developed on the ideas for integer order case but due to the non-local characteristic of fractional derivatives they will differ in important aspects and exhibit problems not known in the classical cases. Before starting the narration of methods it is given in the previous sections that in general there exist many differential approaches to convert the classical calculus to the fractional case, which lead to the number of definition of derivative of fractional order. Such kind of behavior is more critical whenever analyzing the numerical methods.

It is common to develop methods for fractional differential equations by taking the techniques for classical differential equations involving integer order derivatives and then to generalize the concepts in a suitable manner. The reasonable way to refer these methods is then to give them the same name extended with the adjective ‘fractional’ to the underlying classical algorithms. However, classical numerical solvers can be extended in more than one ways due to which it is possible for fractional calculus literature that two different algorithm may be denoted by the same name. Certainly, it can be the source of potential confusion and reader must be careful in this regard.

### 2.5.2 BRIEF HISTORY

Before moving towards numerical techniques, a brief historical survey is given in this section. Most of the authors discussed the numerical method integral equation of Abel Volterra involving Riemann-Liouville fractional integral ([145], [146]). However, such investigation into the FDE equations began only recently.

Shkhanukov [147] applied successfully the one-parameter family of difference scheme to FDEs equations and proved its stability and convergence. Blank [148] applied the collocation method to ordinary FDEs with Caputo derivatives. Diethelm ([149], [150]) suggested an algorithm for differential equations of fractional orders based on Hadamard finite part integral used for interpretation of Riemann-Liouville fractional derivative. Diethelm et al [151] have converted the nonlinear FDEs to second kind of Volterra integral equation and solved it using trapezoidal quadrature formula. Podlubny ([30], [152]) numerical methods for FDEs are based on successive approximation of Riemann-Liouville fractional derivatives. He has applied his approach successfully to the Cauchy type-problems. Bagley and Torvik [153] first consider the problem of behavior of viscoelastic materials in which dynamics of certain gases dissolved in a fluid are mathematically modeled using FDEs.

Diethelm and Ford [154] carried out the numerical treatment of Bagley-Torvik equation by transforming the equation into a system of fractional differential equations of order $1/2$. They provide consistent and stable solution using fractional linear multistep methods and Adams predictor-corrector approach [155]. Edwards et al. [156] provide the solution for multi-term FDEs equation by reduceing it to the system of FDE equations of order at most unity. Ford and Simpson [157] solved the Cauchy type FDEs problems using fixed memory principle as given by Podlubny [30]. Lubich ([158], [159]) used discrete convolution quadrature for approximating Riemann-Liouville fractional integral. He applied his approach on ordinary and partial differential equations of fractional order. Kilbas and Marzan [160] used the corresponding routine of the known numerical mathematical libraries called NAG and

IMSL to obtain approximate solutions for FDE with Cauchy type problem and the Cauchy problem. The ADM, HAM, He's VIM methods are applied to find out approximate numerical solution of non-linear FDE equations. In these methods, the solutions of FDE equations are provided in the form of rapidly convergent infinite series with easily computable components. It can be concluded that the development of numerical techniques for FDE equations is a growing field of investigations for the researchers.

### 2.5.3 GRÜNWALD-LETNIKOV FRACTIONAL BACKWARD DIFFERENCE METHOD

The general form of fractional differential equation can be given as

$$D^{\nu} y(t) = f(t, y(t)), \quad D^{k} y(0) = b_k \quad (k = 0,1,2,...n-1), \tag{2.5.1}$$

where $\nu > 0, \nu \in \Re$ and $n = \lceil \nu \rceil$. The solution $y(t)$ of the equation is determined in the interval $(0, T)$ with a step size of h on input timings $t_0, t_1, t_2, ....t_n$ where $N = T/h$, $t_0 = 0$ and $t_n = T$, $y_m$ and $f_m$ are denoted by $y(t_m)$ and $f(t_m, y_m)$, respectively. Using the definition of Grünwald-Letnikov differential operation the formula of for solving FDEs equations can be given as [29]

$$\frac{1}{h^{\nu}} \sum_{k=0}^{m} (-1)^k \binom{\nu}{k} y(t_m - kh) = f(t_m, y(t_m)), \quad m = 0,1,2,..., N. \tag{2.5.2}$$

Using the expression for coefficient as

$$\omega_k = (-1)^k \binom{\nu}{k} = (1 - \frac{\nu + 1}{k}) \omega_{k-1}, \tag{2.5.3}$$

with $\omega_0 = 1$, and $k = 1,2,3,..., m$, then equation (2.5.2) can be written as

$$y_m = h^{\nu} f(t_m, y_m) - \sum_{k=1}^{m} \omega_k y(t_m - kh), \quad m = 1,2,..., N. \tag{2.5.4}$$

One can compute $y_1, y_2, ..., y_{m-1}, y_m$ by using $y_0 = 0$ from the initial conditions.

### 2.5.4 PODLUBNY'S MATRIX APPROACH

Podlubny provides his techniques for solving FDEs equations ([30], [102]) with homogeneous initial conditions using matrix approach. The formula for obtaining $y_m$ for all nodes simultaneously in a matrix notation is given as

$$\frac{1}{h^{\nu}}\begin{pmatrix} \omega_0 & & & \\ \omega_1 & \omega_0 & & \\ \vdots & \vdots & \ddots & \\ \omega_N & \omega_{N-1} & \dots & \omega_0 \end{pmatrix}\begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_N \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ \vdots \\ f_N \end{pmatrix} \Leftrightarrow \mathbf{B}_N^{\nu}\mathbf{Y}_N = \mathbf{F}_N, \tag{2.5.5}$$

where $y_0 = 1$ is known due to the initial condition, the solution of FDE equation can be obtained by solving simultaneously the system of equations instead of using recurrence formula. Podlubny described his plausible and elegant approach for linear k-term FDEs with homogeneous or inhomogeneous initial conditions of either Riemann-Liouville or Caputo type. However, it is necessary for any inhomogeneous initial condition to transform into homogenous ones using appropriate conversion.

### 2.5.5 DIETHELM'S FRACTIONAL BACKWARD DIFFERENCES BASED ON QUADRATURE

The brief description of Diethelm's fractional backward difference method is provided here, however its detail descriptions can be found in from ([149], [161]). Diethelm's approach can be provided with the help of Grünwald-Letnikov operator using Caputo definition and it is given as

$$y_m = h^{\nu} f(t_m, y_m) - \sum_{k=1}^{m} \hat{\omega}_{km} y(t_m - kh) + h^{\nu} D^{\nu} T_{n-1}[y;0](t_m) \quad m = 1,2,...,N. \tag{2.5.6}$$

where the weight $\hat{\omega}_{km}$ can be written as

$$\hat{\omega}_{km} = \begin{cases} (-1)^k \dfrac{\Gamma(\nu)}{\Gamma(k)\Gamma(\nu-k+1)} + \dfrac{k^{-\nu}}{1-\nu} & k = m \geq 0, \\ (-1)^k \dfrac{\Gamma(\nu+1)}{\Gamma(\nu+1)\Gamma(\nu-k+1)} + \dfrac{k^{-\nu}}{1-\nu} & 0 \leq k \leq m-1, \end{cases} \tag{2.5.7}$$

and $D^{\nu}T_{n-1}[y;0](t_m)$ with $b_k, k = 1,2,...,n$ for Riemann-Liouville initial conditions is

$$D^{\nu}T_{n-1}[y;0](t_m) = \sum_{k=0}^{n-1} \frac{b_k t_m^{k-\nu}}{\Gamma(k+1-\nu)} \tag{2.5.8}$$

### 2.5.6 LUBICH'S FRACTIONAL BACKWARD DIFFERENCE METHODS

Lubich has developed his fractional backward difference method on the basis of fractional linear multistep techniques ([158], [159],[162]). Such approaches have already been implemented numerically on Volterra integral equations [163].

Lubich's fractional backward difference method of order $p \in \{1,2...,6\}$ for FDEs equations of Caputo type can be given by the formula

$$y_m = h^{\nu} f(t_m, y_m) - \sum_{i=0}^{m-1} \omega_{m-j}.y(t_j) - \sum_{j=0}^{s} w_{m,j} y(t_j) + D^{\nu}T_{n-1}[y;0](t_m), \tag{2.5.9}$$

for $m = 1,2...,N$, where $\nu > 0$, $n = \lceil \nu \rceil$, the convolution weights $\omega_m$ in the form of generating function are given as

$$\omega^{\nu}(x) = \left( \sum_{k=1}^{p} \frac{1}{k} (1-x)^k \right)^{\nu}, \tag{2.5.10}$$

and the starting weights $w_{m,j}$ are provided by the solution of linear system of quations

$$\sum_{j=0}^{s} w_{m-j} j^{\gamma} = \frac{\Gamma(1+\gamma)}{\Gamma(1+\gamma-\nu)} m^{\lambda-\nu} - \sum_{j=0}^{m} \omega_{m-j} j^{\gamma}, \quad \gamma \in A, \tag{2.5.11}$$

with

$$A = \{\gamma = k + j\nu; k, j \in \mathrm{N}, \gamma \leq p-1\}, \quad card\ A = s+1., \tag{2.5.12}$$

The restriction of order $p \in \{1,2...,6\}$ is made due to the fact that backward difference methods are stable only up to order $p = 6$.

### 2.5.7 ADOMIAN DECOMPOSITION METHOD

This is an analytic technique developed on Abel-Volerra integral equation using infinite sum of basis solution given as [164]

$$y(t) = \sum_{i=0}^{\infty} y_i(t) = g(t) + \frac{1}{\Gamma(\nu)} \int_0^t (t-\tau)^{\nu-1} \sum_{i=0}^{\infty} {}_f A_i(\tau) d\tau, \tag{2.5.13}$$

where ${}_f A_i(\tau)$ are the Adomian polynomials. The decomposition method is defined explicitly as

$$y_0(t) = g(x), \quad y_{i+1}(t) = \frac{1}{\Gamma(\nu)} \int_0^t (t-\tau)^{\nu-1} {}_f A_i(\tau) d\tau, \quad i = 0,1,2,..., \tag{2.5.14}$$

and the Adomian polynomials ${}_f A_i(\tau)$ is provided by

$${}_f A_i(\tau) = \left[ \frac{1}{i!} \frac{d^i}{d\lambda^i} f\left( x, \sum_{j=0}^{i} \lambda^j y_j \right) \right]_{\lambda=0}. \tag{2.5.15}$$

The function $y_j$, $j = 0,1,2,...i$ is written for $y_j(\tau)$ for clarity. Although if it is not possible to use infinite scheme, even then a finite expansion can be obtained for corresponding differentiability properties of $f(t, y(t))$. Such finite expansion contains again all necessary information on the lower order terms. The ADM method is widely used for solving various initial and boundary value problems associeated with ordinary and partial fractional differential equations. Some of recent applications of ADM algorithm included the solution obtained for Rayleigh-Stokes problem given by fractional model of second grade fluid [165], solving fractional advection-dispersion equation [166], and solution of discrete hybrid equation associated with constructed solitary [167].

### 2.5.8 ADAMS METHODS

There are many methods developed for fractional order system in engineering, however, we have closed our discussion on the subject matter by providing the simple description of Adams method. It can be used for solving highly nonlinear fractional differential equations.

The formula for one step fractional variant of one-step Adams-Moulton method is ([168], [169]):

$$y_{k+1} = \sum_{j=0}^{n-1} \frac{t_{k+1}^{j}}{j!} b_j + \frac{1}{\Gamma(\nu)} \left( \sum_{j=0}^{k} a_{j,k+1} f(t_j, y_j) + a_{k+1,k+1} f(t_{k+1}, y_{k+1}^{p}) \right), \tag{2.5.16}$$

where $a_{k+1,k+1}$, is given by

$$a_{k+1,k+1} = \frac{(t_{k+1} - t_k)^{\nu}}{\nu(\nu+1)}, \tag{2.5.17}$$

and in case of equispaced nodes the relation for $a_{j,k+1}$ is given as

$$a_{j,k+1} = \begin{cases} \dfrac{h^{\nu}}{\nu(\nu+1)} \left( k^{\nu+1} - (k-\nu)(k+1)^{\nu} \right) & \text{if } j = 0, \\ \dfrac{h^{\nu}}{\nu(\nu+1)} \left( (k-j+2)^{\nu+1} + (k-j)^{\nu+1} - 2(k-j+1)^{\nu+1} \right) & \text{if } 1 \le j \le k, \\ \dfrac{h^{\nu}}{\nu(\nu+1)} & \text{if } j = 0 \end{cases} \tag{2.5.18}$$

Now the remaining problem is to determine the predictor formula that is required for the calculation of value $y_{k+1}^{p}$. For the said purpose the generalization of one-step Adams-Bashforth method is the used by replacement of the integral in Abel-Volterra integral equation with product rectangle rule

$$\int_{0}^{t_{k+1}} (t_{k+1} - \tau)^{\nu-1} g(\tau) d\tau \approx \sum_{j=0}^{k} b_{j,k+1} g(t_j), \tag{2.5.19}$$

Thus, the predictor $y_{k+1}^{p}$ is given by the fractional Adams-Bashforth method as

$$y_{k+1}^{P} = \sum_{j=0}^{n-1} \frac{t_{k+1}^{j}}{j!} b_j + \frac{1}{\Gamma(\nu)} \sum_{j=0}^{k} b_{j,k+1} f(t_j, y_j), \tag{2.5.20}$$

where

$$b_{j,k+1} = \frac{h^{\nu}}{\nu}\left((k+1-j)^{\nu} - (k-j)^{\nu}\right). \tag{2.5.21}$$

It completes the algorithm for fractional Adams-Bashforth-Moulton method described in (2.5.16) and (2.5.20). The weights are provided in expresions (2.5.17), (2.5.18) and (2.5.21).

## 2.6 SUMMARY OF CHAPTER

This chapter presents the literature review for fractional calculus and its applicatons to engineering systems. The chapter starts with the description of brief history of fractional calculus from the origin in 1665 till its present status. The important relations used in fractional calculus like Mittag-Leffler function, gamma function and few basic definitions of fractional integrals and derivatives are presented, which are provided by Riemann Liouville, Caputo and Grünwald-Letnikov. The description of Fractional order system in engineering as fractional differential equations are also presented and few applications are included like viscoeleasticity, Bode's analysis of feedback amplifiers, fractional LMS algorithm, Riccati and Bagley-Torvik fractional system, fractional Chua-Hartley system and fractional diffusion equations. Chapter ends with the narration of deterministic numerical treatment of fractional differential equations such as Grunwald-Letnikov fractional backward difference method, Podlubny's matrix approach, Diethelm and Lubish fractional backward difference methods, Adomian decomposition method and Adams methods.

# CHAPTER 3

# DESIGN METHODOLOGY FOR SOLVING FRACTIONAL ORDER SYSTEMS

In this chapter, the detailed design methodology for fractional order system represented with fractional differential equations is provided. The proposed scheme consists of two parts. In the first part, the well known strength of artificial neural networks is exploited to model the fractional differential equations. In the second part, the heuristic learning methodologies are given for training of weights of neural networks. These are based on global and local search optimizers.

## 3.1 NEURAL NETWORK MATHEMATICAL MODELING

In this section, mathematical modeling for fractional order system (FOS) given by differential equations of arbitrary order with feed forward artificial neural network is presented. It is based on the existing neural network modeling for the differential equation of integer order. The possible extension to the existing designed methodology is developed, so that it is applicable to the equations containing non-integer order derivatives as well.

The generic form of FOS containing single and multi-terms fractional derivatives can be written as

$$D^{\nu} y(t) = f(t, y(t), D^{n} y(t)), \qquad 0 \le t \le T \tag{3.1.1}$$

with initial conditions as follows.

$$D^{k} y(0) = c_k, \quad k = 0,1,2,\ldots,N-1, \tag{3.1.2}$$

and boundary condition at $t = t_b$ for $0 \le t_b \le T$ is written as

$$D^k y(t_b) = b_k, \quad k = 0,1,2,\ldots,N-1, \tag{3.1.3}$$

where $D$ is the derivative operator, $\nu$ is the order of fractional derivative term, $(\nu,\mu) > 0, (\nu,\mu) \in \Re, N = \lceil \nu \rceil, n \in \mathrm{N}$, $y(t)$ is the solution of the equation, $T$ is the constant representing the span of inputs within the close interval $[0, T]$, $c_k$ and $b_k$ are the constants giving initial and boundary conditions, respectively.

### 3.1.1 INTEGER ORDER CASE

It is well known that an arbitrary continuous function and its derivatives on a compact set can be arbitrarily approximated by multiple inputs, single output, single hidden layer feed forward neural networks with a linear output layer having no bias. The solution $y(t)$ of the differential equation along with its 2nd and $n^{th}$ order derivative, $d^n y / dt^n$, can be approximated by the following continuous mapping as in neural network methodology ([170], [171], [172]):

$$\hat{y}(t) = \sum_{i=1}^{m} \alpha_i f(w_i t + \beta_i) \tag{3.1.4}$$

$$\frac{d\hat{y}}{dt} = \sum_{i=1}^{m} \alpha_i \frac{d}{dt} f(w_i t + \beta_i) \tag{3.1.5}$$

$$\vdots$$

$$\frac{d^n \hat{y}}{dt^n} = \sum_{i=1}^{m} \alpha_i \frac{d^n}{dt^n} f(w_i t + \beta_i), \tag{3.1.6}$$

where $\alpha_i, w_i$, and $\beta_i$ are bounded real-valued adaptive parameters, $m$ is the number of neurons, and $f$ is the activation function normally taken as log sigmoid function

$$f(x) = \frac{1}{1+e^{-x}}. \tag{3.1.7}$$

The mathematical model for differential equation involving integer order derivatives can be formulated with the linear combinations of the networks (3.1.4) to (3.1.6). It means that the solution $y(t)$ can be approximated with $\hat{y}(t)$ subject to some

appropriate unknown weights. Moreover, the accuracy of the algorithm increase as $\hat{y}(t)$ match closely to $y(t)$ and it depends upon availability of suitable weights for networks. Such kind of neural networks based mathematical modeling for linear and nonlinear differential equation has been used by various authors in wide areas of applied sciences ([171], [172]).

### 3.1.2 FRACTIONAL ORDER CASE

The networks represented by (3.1.4) to (3.1.6) could not be applied to model the differential equations of fractional order due to non-availability of the fractional derivative of the log-sigmoid activation function. The exponential function is a candidate solution to be used as activation function in the neural network modeling for such equation. It has universal function approximation capability, as well as, fractional order derivative is known with terms represented by classical MLF function of two parameters.

The approximate continuous mappings in the form of linear combination of exponential functions can be taken to approximate the solution $y(t)$, its integer order derivative $D^n y(t)$ and fractional derivatives $D^\nu y(t)$ as

$$\hat{y}(t) = \sum_{i=1}^{m} \alpha_i \, e^{w_i t + b_i}, \tag{3.1.8}$$

$$D^n \hat{y}(t) = \sum_{i=1}^{m} \alpha_i w_i^n e^{w_i t + b_i}, \tag{3.1.9}$$

$$D^\nu \hat{y}(t) = \sum_{i=1}^{m} \alpha_i e^{b_i} t^{-\nu} E_{1,1-\nu}(w_i t), \tag{3.1.10}$$

respectively. The linear combination of the networks represented in equations (3.1.8) to (3.1.10) can approximately model the fractional differential equations as given in (3.1.1). The standard ANN architecture has been extended to be applicable to solution for these problems. It is named as fractional differential equation neural network (FDE-NN). A generic form of the FDE-NN network architectures for equations (3.1.1) is represented in Fig. 3.1.

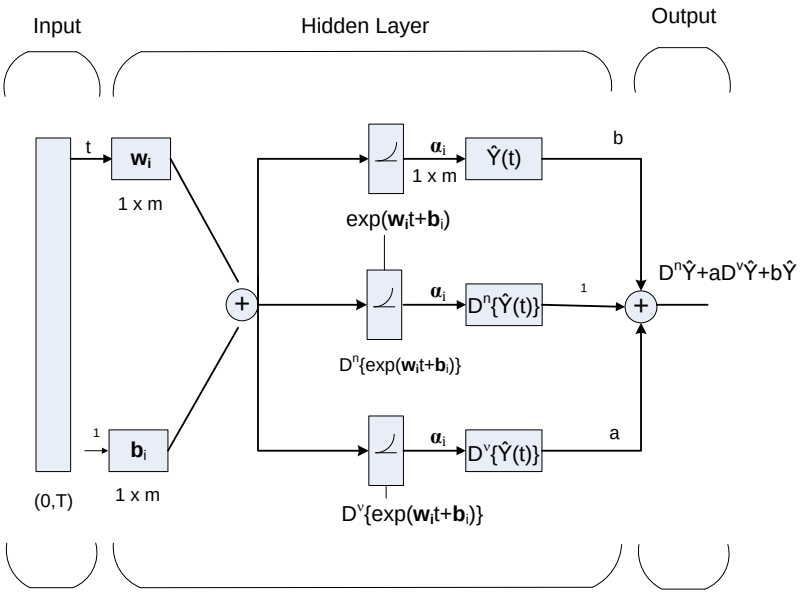



**Fig. 3.1** FDE-NN networks architecture for generic FDE equations.

### 3.1.3 FITNESS FUNCTION

The unsupervised error function $e$ is formulated for defining the fitness function. It is given by the linear combination of FDE-NN networks (3.1.8) to (3.1.10) for fractional order systems as

$$e = \frac{1}{1+e_j} \quad j = 1,2,\ldots\ldots \tag{3.1.11}$$

where $j$ is the cycle count for the algorithm, and the function $e_j$ is defined as the mean of sum of square errors

$$e_j = e_1 + e_2\big|_j \tag{3.1.12}$$

where $e_1$ is error associated with the equation to be modeled. In case of equation (3.1.2), it is given as

$$e_1 = \frac{1}{s}\sum_{i=1}^{s}[D^v\hat{y}(t_i) - f(t_i, \hat{y}(t), D^n\hat{y}(t_i))]^2 \tag{3.1.13}$$

where $s$ is the total number of steps, each step is taken between $(0, T)$. Greater the value of *s*, lesser is the step size and more will be the accuracy of algorithms, but at the cost of increase in computational complexity for the algorithms.

Similarly, $e_2$ is linked with initial and boundary conditions and it is written as

$$e_2 = \frac{1}{N}\sum_{k=0}^{N-1}(D^k y(0) - c_k)^2 + \frac{1}{N}\sum_{k=0}^{N-1}(D^k y(t_b) - b_k)^2 \tag{3.1.14}$$

It is quite evident that subject to the availability of unknown weights for which the function $e_j$ approaches zero, the value of fitness function $e$ approaches its maximum. The solution $y(t)$ of the equation is approximately modeled by $\hat{y}(t)$ as given in (3.1.8).

Similarly, the generic form of fractional order system given by non-linear quadratic Riccati and Bagley-Torvik differential equation of fractional order can be written as

$$\frac{d^{\nu} y(t)}{dt^{\nu}} = p(t) + q(t)y(t) + r(t)y^2(t), \quad 0 < t \le T, \tag{3.1.15}$$

$$AD^2 y(t) + BD^{\nu} y(t) + Cy(t) = g(t), \quad 0 < t \le T \tag{3.1.16}$$

with initial conditions given as

$$D^k y(0) = c_k, \quad k = 0, 1, \tag{3.1.17}$$

whereas boundary condition is written as

$$D^k y(t_0) = b_k, \quad k = 0,1 \quad 0 < t_o \le T \tag{3.1.18}$$

where $D = d/dt$ is derivative operator, $p(t), q(t), r(t)$ and $g(t)$ are known functions of $t$. The linear combination of FDE-NN networks provided in equation (3.1.8) to (3.1.10) can also model (3.1.16), and (3.1.17). The generic FDE-NN networks architecture for Riccati and Bagley-Torvik fractional differential equation is provided in Fig. 3.2 and Fig. 3.3, respectively.

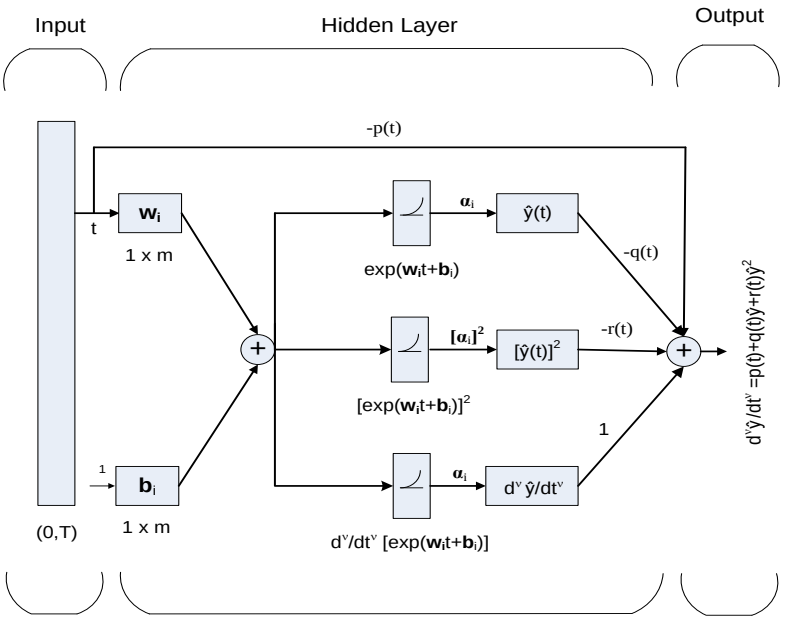



**Fig. 3.2** The FDE-NN networks architecture for nonlinear Quadratic Riccati fractional differential equation.

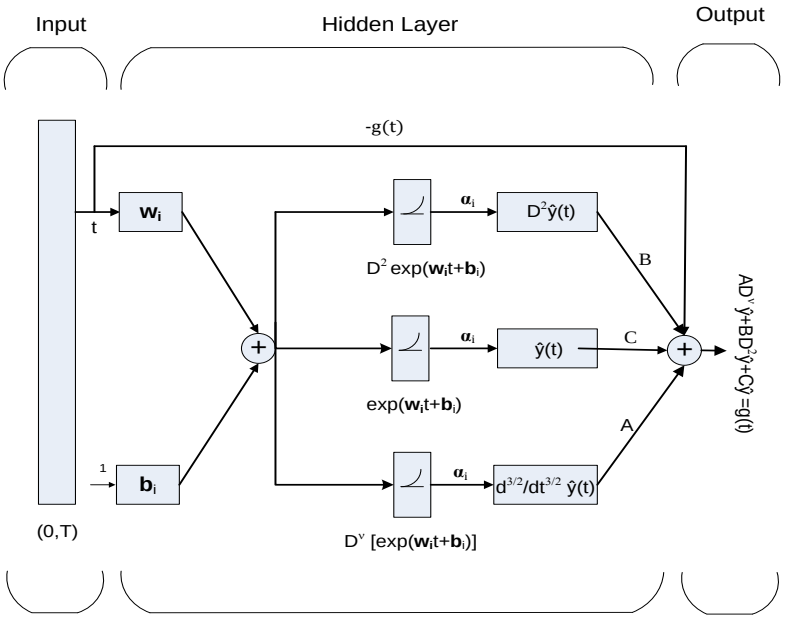



**Fig. 3.3** FDE-NN networks architecture for fractional order system of generic Bagley-Torvik equation.

The fitness function is also same as defined in equations (3.1.11) and (3.1.12) but the expressions $e_1$ for (3.1.15) and (3.1.16) are, respectively, given by

$$e_1 = \frac{1}{s}\sum_{i=1}^{s}\left(D^{\nu}\hat{y}(t_i) - p(t_i) - q(t_i)\hat{y}(t_i) - r(t_i)\hat{y}^2(t_i)\right)^2, \tag{3.1.19}$$

$$e_1 = \frac{1}{s}\sum_{i=1}^{s}\left(AD^2\hat{y}(t_i) + BD^{\nu}\hat{y}(t_i) + C\hat{y}(t_i) - g(t_i)\right)^2, \tag{3.1.20}$$

where $\hat{y}(t), D^2\hat{y}(t)$ and $D^{\nu}\hat{y}(t)$ are the networks given in (3.1.8), (3.1.9) for $n = 2$ and (3.1.10), respectively.

Similarly, $e_2$ is linked with initial and boundary conditions and it is given as

$$e_2 = \frac{1}{2}\sum_{k=0}^{1}[(D^k\hat{y}(0) - c_k)^2 + (D^k\hat{y}(t_0) - b_k)^2]\Big|_j \tag{3.1.21}$$

It is quite evident that the weights, for which the value of fitness function *e* is maximized, are the most desired ones.

## 3.2 COMPUTATIONAL INTELLIGENCE TECHNIQUES

In this section, the introductory material is given for the methodologies used in learning of the unknown weights of networks representing the equation. The well known computation intelligence techniques like Evolutionary Computational and Swarm intelligence techniques aided with efficient local search methods, such as simulating annealing, pattern search and active set algorithm, are used to solve such optimization problem.

### 3.2.1 EVOLUTIONARY COMPUTATIONAL INTELLIGENCE

Evolutionary computation use natural evolution as an optimization mechanism for solving the problems. The main objective of every evolutionary computational technique is to find a good solution to a problem from a large search space of candidate solutions. There are some aggressive methods like simulated annealing, hill climbing etc, while the non-aggressive consist of genetic algorithms (GA). Simulating annealing (SA) is a kind of optimization technique based on annealing of metal [173]. It can be found the minimum using stochastic searching technique from the mean of

the probability. SA algorithm has a strong ability to find the local optimistic result. It can also avoid the problem of local minimum, but its ability of finding the global optimal result is weak. The main concern in the hill climbing is local minimum trap. In general all aggressive search technique stuck in the local minimum. Differential evolution (DE) is very successful for global continuous optimization problems ([174], [175]). It mainly utilizes the differential information to guide its further search. DE algorithms have very good search ability but, it may be trapped in local minima. On the other hand, genetic algorithm does not get stuck in local minima and is a powerful tool for optimization problems [176]. The good source of referencne meterail in evolutionary computational algorithms can be found in books ([177], [178], [179]). However, our studies are confined to evolutionary algorithm based on GA. A brief description of GAs is provided in next section.

#### 3.2.1.1 GENETIC ALGORITHMS

Genetic Algorithm (GA) is global optimization tool based on natural selection and genetic mechanisms. GAs have been developed for the solution of many non-linear problems in engineering and applied sciences, such as a novel non-linear feature selection problem [180], bankruptcy prediction modeling [181], and non-linear mixed integer programming problems [182]. One of the prominent features of GAs is that unlike ANNs, do not get stuck in local minima. GAs incorporate parallel procedure as well as structured strategy for randomly searching high aptitude points. Generally the GA consists of three fundamental operators: selection, crossover and mutation. GA encodes the design parameters into finite bit string to solve the required optimization problem. GA runs iteratively using its operators randomly based upon some fitness function. Finally, it finds and decodes the solution from the last pool of mature strings obtained by ranking of strings, exchanging the portions of strings and changing some bit of the strings. The following basic terms are used in GAs.

***Chromosome or Individual****:* It is a set of genes. Chromosome or individual contains the solution represented in the form of genes.

***Gene**:* It is a part of chromosome and contains only a part of solution

***Population**:* Number of individuals present with same length of chromosome.

***Fitness and Fitness function**:* Fitness is the rank value assigned to an individual. It is based on how far or close an individual is from the solution. The Greater the fitness value, the better is the solution. Fitness function is a function which assigns fitness value to the individual. It is problem specific.

***Crossover:*** The process by which the chromosomes from the parents exchange systematically using probabilistic decision. It means change occurs during reproduction. Therefore, the offspring exhibit some traits of the father and some traits of the mother.

***Mutation**:* The process of changing a random gene in an individual. How often to do mutation, how many genes to change, and how big a change to make are adjustable parameters. Suitable mutation avoids the early maturation.

***Selection**:* Selecting individuals for creating the next generation. The aim in selection is to give the fitter individuals a better chance to survive in the next generation. It is against the nature to kill all unfit genes as they may mutate to something handy. Therefore, in selection there is always a tradeoff for better individual and diversity.

The genetic algorithm process is discussed through GA cycle as given in Fig. 3.4 and the flowchart showing the process of evolutionary algorithm is as shown in Fig. 3.5.

#### 3.2.1.2 ALGORITHM STEPS

Evolutionary computation algorithm can be given in the following steps:

Step 1. ***Initial Population***: Randomly generate bounded real value to form initial population of $P$ number of the individuals or chromosomes. Each individual represents the unknown parameters of neural network. The initial population is scatter enough for better search space of the algorithm. Create a $Q$ number of subpopulations each having $P/Q$ individuals.

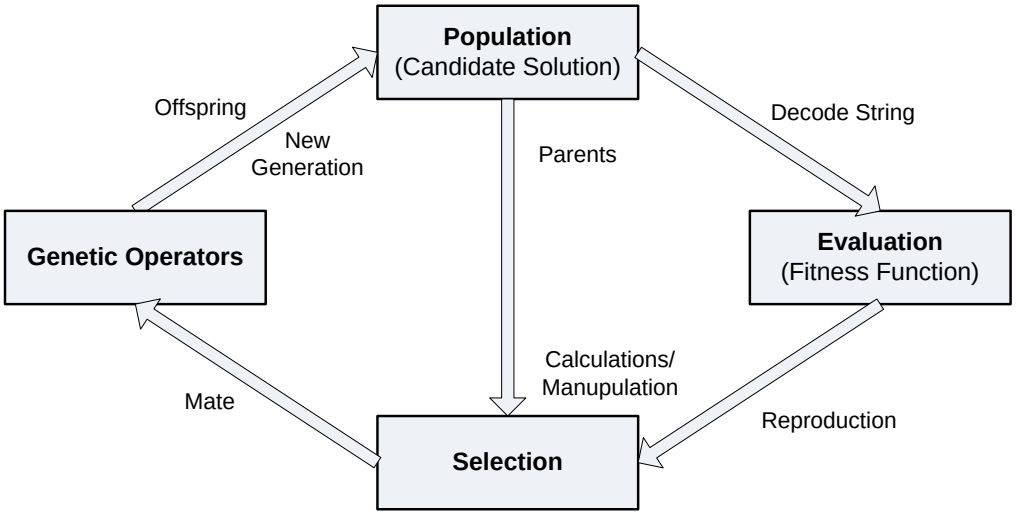



**Fig. 3.4** Genetic Algorithm Cycle.

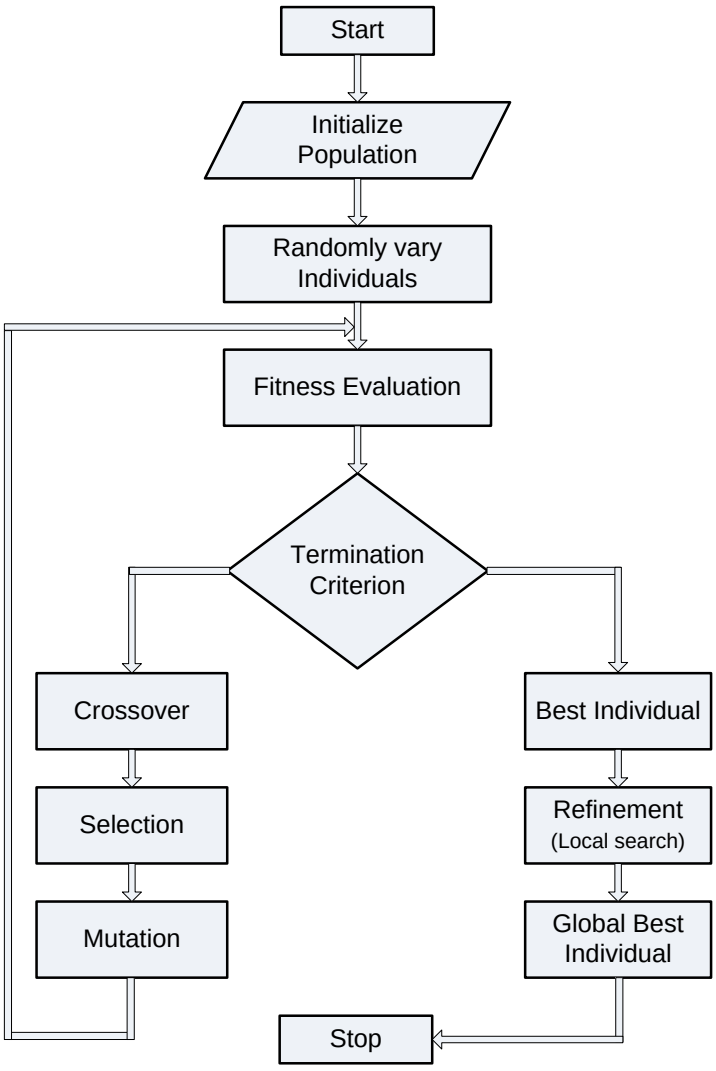



**Fig. 3.5** Flowchart of Evolutionary Algorithm.

Step 2. ***Initialization****:* Initialize parameter setting as follows for algorithm execution.

Set the number of variable in a GA function equivalent to element in the individual. Set the number of generations. Set the fitness limit, start cycle count. Set the value of Elite count and crossover fraction for reproduction. Set Migration in both forward and backward direction. Start generation count etc.

Step 3. ***Fitness evaluation****:* Calculate fitness by using the fitness function given in expressions (3.1.11-14).

Step 4. ***Ranking****:* Ranking is carried out for each individual of the populations on the basis of maximum fitness values.

Step 5. ***Termination Criteria***: Check for termination of the algorithm, which is set as either predefined fitness value i.e. MSE $10^{-08}$ for linear FDEs equations and $10^{-04}$ for non-integer order cases of the equation or Number of cycles is complete. If yes, go to step 8 else continue.

Step 6. ***Reproduction***: Create next generation on the basis of

Crossover: Call for scattered function, Mutation: Call for Adaptive Feasible function, Selection: Call for Stochastic Uniform function and Elitism etc.

Step 7. ***Improvement***: Repeat the procedure from step 3 to step 6 with newly generated population for improvement of results until total number of cycle is complete.

Step 8. ***Refinement***: Further fine tuning of result is made by using local search technique by taking the best fitted individual of step 7 as start point of the algorithm. Call for MATLAB function for such techniques like active-set algorithm, simulating annealing method, pattern search techniques etc. Store in the memory the refined value of fitness along with the best individual for this run of the algorithm.

Step 9. ***Statistical Analysis***: Repeat step 1 to 8 for sufficiently large number of the times to make the statistical analysis of the algorithm. Store these result in memory for detail analysis later.

### 3.2.2 SWARM INTELLIGENCE TECHNIQUES

Swarm intelligence techniques belong to the class of biologically inspired computing algorithms such as artificial neural networks, evolutionary algorithm, artificial immune systems, fractal geometry, and Quantum computing etc. It is basically an amalgamation of computational and collective intelligence of groups that interact with the environment. There are two common swarm intelligence techniques in computational areas: The Ant colony optimization (ACO) introduced by Marco Dorigo in 1993 and Particle swarm optimization (PSO) introduce by Eberhart and Kennedy in 1995 [183]. These techniques are widely used in diverse fields of engineering and applied sciences due to their ability to provide extremely simple high-level algorithmic framework for challenging optimization problems like physically-based erosion model [184], resource contrained scheduling problem [185], and job-shop scheduling algorithm [186]. Comprehensive studies for the method are available in standard reference books ([187], [188]). However, our studies here are limited to Particle swarm optimization algorithm. The main advantages of the PSO algorithm are simple in concept, ease in implementation, robustness in controlling parameters, and computational efficiency compared to other mathematical algorithms and heuristic optimization techniques ([189], [190]). Moreover, PSO can easily be applied to nonlinear and continuous optimization problems and generate high quality solutions in lesser computational time and with more stable convergence characteristics.

#### 3.2.2.1 PARTICLE SWARM OPTIMIZATION ALGORITHM

***Introduction:*** The PSO was introduced by Eberhart and Kennedy in 1995 as an alternative to Genetic Algorithm [183]. It is a stochastic global optimization technique based on the behavior of individuals (i.e., particle or agents) of the swarm. Its roots are in modeling of the movement of particle, inspired by social behavior of bird flocking or fish schooling [183]. It simulates the conduct of bird flocking involving the scenario of a group of birds sharing information among them, a fact that leads to

enhance the efficiency of the group. The PSO algorithm searches in parallel using such groups. Discrete and continuous versions of PSO algorithm have widely been applied to different optimization problems in science and engineering. Few examples include multiuser detection schemes in mobile communications ([191], [192]), sensor networks [192]. Similarly its continuous versions have been used in diverse fields like inventory control [193], multiprocessor scheduling [194], controls ([195], [196]) etc. Some improved versions of PSO can also be seen in the literature ([197], [198]).

***Standard PSO**:* the PSO algorithm is originally initialized with a swarm of particles placed on the search space randomly and is used to search for optimal solution iteratively. In each cycle the position and the velocity of each particle are updated according to its known previous local best position, $\mathbf{Lbest}_i^{n-1}$, and the global best position of all particles, $\mathbf{Gbest}_i^{n-1}$, in the swarm so far. The updating formula for each particle's velocity and position in continuous PSO is given as

$$\mathbf{V}_i^j = \omega \mathbf{V}_i^{j-1} + c_1 \mathbf{r}_1 (\mathbf{Lbest}_i^{j-1} - \mathbf{X}_i^{j-1}) + c_2 \mathbf{r}_2 (\mathbf{Gbest}^{j-1} - \mathbf{X}_i^{j-1}), \tag{3.2.1}$$

$$\mathbf{X}_i^j = \mathbf{X}_i^{j-1} + \mathbf{V}_i^j \tag{3.2.2}$$

where $\omega$ is the inertia weight linearly decreasing over the course of search from *1* and *0*, $\mathbf{r}_1$ and $\mathbf{r}_2$ are random vectors and there elements taken between *0*, and *1*, *i = 1,2,3,…M*, where *M* is the number of particles in a swarm, *j* and *j-1* is the present and previous flight, respectively. The scalar constants $c_1$ and $c_2$ are the local and global social acceleration constants respectively. **X** and **V** are the parameters of position and velocity, its member are restricted to real numbers such that $(X, V) \in \Re^d$ where *d* be the dimension of search space.

Following are the common terms used in PSO.

***Particle:*** In PSO algorithm, each single solution to an optimization problem is considered as a particle in the search space.

***Swarm:*** The exploration of a problem space was made in PSO by a population of particles called a swarm.

***Fitness Function:*** All particles in the swarm have fitness values which are evaluated by the problem specific fitness function.

***Local Best:*** The particle with best fitness value with respect to its previous best position.

***Global Best:*** The particle with best fitness in the swarm is call global best particle.

***Local intelligence:*** The second term on the right hand side in (18) is the difference between local best and position of $i^{th}$ particle. It is called as local intelligence. The more this difference, the more will be the change in velocity.

***Collective intelligence:*** The third term in equation (18) is the difference between global best position and position of $i^{th}$ particle is known as collective intelligence. Generally, more weightage is given to the local intelligence in the initial stages while global intelligence is given more importance in the ending phase of the optimization problem.

***Maximum Velocity****:* The velocity $V \in [-V_{\max}, V_{\max}]$ where $V_{\max}$ is maximum velocity. It is a constant preset defined by users that controls the convergence rate and can prevent the method from growing too fast.

***Termination:*** The termination criterion is determined according to whether the presetting maximum flights or a designated value of the fitness is reached.

The generic flow diagram of PSO algorithm used in this article is given in Fig. 3.6.

#### 3.2.2.2 ALGORITHM IN STEPS

The algorithm is given in the following steps:

Step 1 ***Initialize swarm***: Randomly generate bounded real values to form initial swarm of particles. Each particle represents the unknown parameters of neural network. The initial swarm is scattered enough for better search space for the algorithm.

Step 2 ***Initialization***: Following parameter values assigned for algorithm execution. Set the number of flights. Set the fitness limit and start cycle count. Set the

values of individual best and global best acceleration factors. Set the value of inertia weight ω and maximum velocity $V_{max}$.

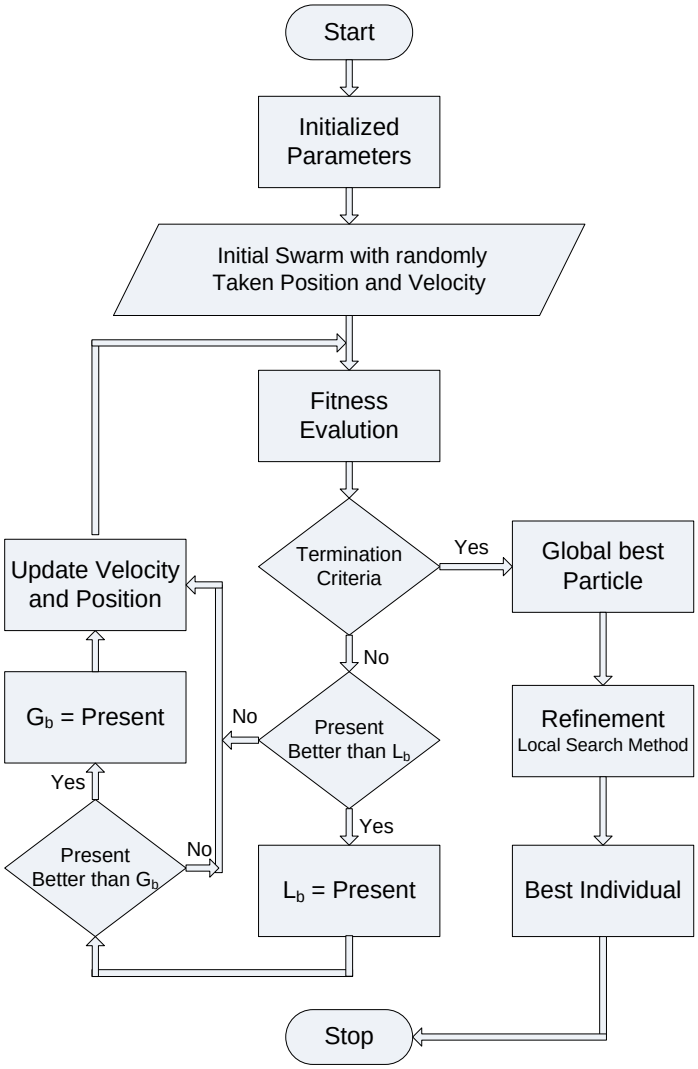



**Fig. 3.6** Generic flow diagram of PSO algorithm.

Step 3 ***Fitness Evaluation***: Calculate fitness by using the fitness function given in expressions (3.1.11) to (3.1.14).

Step 4 ***Ranking****:* Rank each particle of the swarm on the basis of minimum fitness values. Store the best fitness particle.

Step 5 ***Termination Criteria****:* Terminate the algorithm if either predefined fitness value i.e. MSE $10^{-08}$ for linear FDEs and $10^{-04}$ for non-linear FDEs is achieved

or number of maximum flights/cycles is reached. If yes go to step 7 else continue

Step 6 ***Renewal:*** Update the Velocity using equation (3.2.1). Update the position using equation (3.2.2). Repeat the procedure from step 3 to step 6 until total number of flights is reached.

Step 7 ***Storage***: Store the best fitted particle so far and name it as global best particle for this run.

Step 8 ***Refinement:*** MATLAB optimization toolbox is used for SA algorithm and PS techniques for further fine-tuning by taking the best fitted particle as start point of the algorithm. Store the value of fitness along with the best particle for this run. Stop.

### 3.2.3 LOCAL SEARCH METHODS

The rate of convergence of the global search methods is decreased considerably with an increase in number of iterations of the optimization problems. Therefore, in such scenarios very slight improvements in the results have seen. However, if local search method is hybridized at this stage of the optimization problem, it will expedite the procedure substantially. There are number of local search techniques available, but this work presented in this study is limited to SA technique, PS method and Active set algorithm only.

#### 3.2.3.1 SIMULATING ANNEALING

Simulated annealing (SA) technique was originally proposed by Metropolis in 1950s, in which crystallization process model is explained. However, in early 1980s, independent research on the subject has been carried out by Kirkpatrick et. al [199] and Cerny [200]. They noted the similarities between optimization problem for combinatorial and the physical process of annealing and found a correspondence between the different physical states of the material and the candidate solution space

of an optimization problem. It was also observed that the fitness function of an optimization problem links with the free energy of the material.

The annealing is the process in which a solid is heated in a heat bath by increasing the temperature of the bath until the solid is melted into liquid, and then the temperature is declined slowly. In statictical mechanics, SA is introduced to solve complex combinatorial optimization problems based on the annealing process. SA method is a kind of global optimization technique and found the global minimum using stochastic searching technique from the mean of the probability. SA algorithm has a strong ability to find the local optimal result as well. SA approaches can avoid the problem of local minimum, but its ability to find the global optimistic result is relatively weak. The main advantages of the SA techniques lies in its application to hard optimization problems without considering the conditions of differentiability, convexity and continuity, which are required normaly for other optimization methods.

Following is brief description of some of its terminology [201]:

***Temperature:*** The controlling parameter in SA technique that is gradually decreasing as the algorithm proceeds. It is parameters on which the acceptance probability of candidate solution is determined at any step in a search space.

***Annealing Schedule:*** Defines the rate of decrease in temperature as the algorithm proceeds. It is worth mentioning that using slowly decrease the rate of temperature, the better is the performace of algorithm, but at the cost of more computations.

***Reannealing:*** It is the method to control the temperature precisely during cooling a material such that it is brought to an optimal state. After the sufficient number of new points has been accepted, it raises the temperature and starts the algorithm again at the higher temperature. It is due to reannealing for which one can avoid to get stuck at local minima

Moreover, SA scheme is used extensively by researcher in diverse fields of engineering and technologies like the transmission network expansion planning problem [202], operational research problems [203] and unit commitment problem [204].

#### 3.2.3.2 PATTERN SEARCH ALGORITHM

The name, pattern search, was introduced first by Hooke and Jeeves [205], however, the convergent pattern search method was developed using the theory of positive bases by Yu [206]. Pattern search algorithm is a kind of aggressive optimization technique which belongs to the class of direct search methods, or black box method or derivative free techniques [207], [208]. Hence it can be applied for optimization problems which are not continuous and the gradiant information is not already available. It can find the optimum values using stochastic searching technique based on scaled and translated integer lattice. PS techniques have been extensively used by the research optimization problem like Bound Constrained Minimization [209], linearly Constrained Minimization [210] and Globally Convergent Augmented Lagrangian Algorithm [211].

Some of its common terms are discribed as

***Poll*:** Defines the set of mesh point for the search of the algorithm.

***Poll Method***: it specifies the method used to create the mesh points. Normally generalized pattern search algorithms or mesh adaptive direct search algorithms is used for setting the poll.

***Mesh***: It is a parameter which avoids the alogorithm from premature convergence. The comprehensive search space is explored by robust controlling of Mesh size, its acceleration, scaling and rotation characteristics.

#### 3.2.3.3 **Active Set Algorithm**

Active set algorithm is used in constrained optimization problems [212] Normally the objective is to transform the problem into an easier solvable subproblem which is used as the basis of an iterative process. In early methods, the translation of the constrained problem to a basic unconstrained problem is made using a penalty function for constraints. Thus the constrained problem is solved using a chain of parameterized unconstrained optimizations. These schemes are now considered to be inefficient and have been replaced by techniques based on the solution of the Karush-

Kuhn-Tucker (KKT) equations. The KKT equations are necessary conditions for optimization of constrained problems, whereas if the problem is convex programming then the KKT equations are both necessary and sufficient for global candidate solution. These schemes attempt to obtain the Lagrange multipliers directly.

An ASA method has the following structure in general

**Start Point**: Find an appropriate starting point.

**Repeat until**: Optimal condition is satisfied.

*Solve* the equality problem defined approximately by the active set.

*Computation* of Lagrange multipliers for the active set

*Removal* of the constraints subset with negative Lagrange multipliers

*Search* for infeasible constraints

**End repeat**

The AS algorithm has been applied in many particle application of optimization problems like Mathemtical Programing Problem with Equilibrium Constraints [213], Box constrained optimization [214], and sparse linear programming problems [215]. The flow diagram for obtaining desired weights of FDE-NN networks by Genetic algorithm hybridized with pattern search algorithm is given in Fig. 3.7.

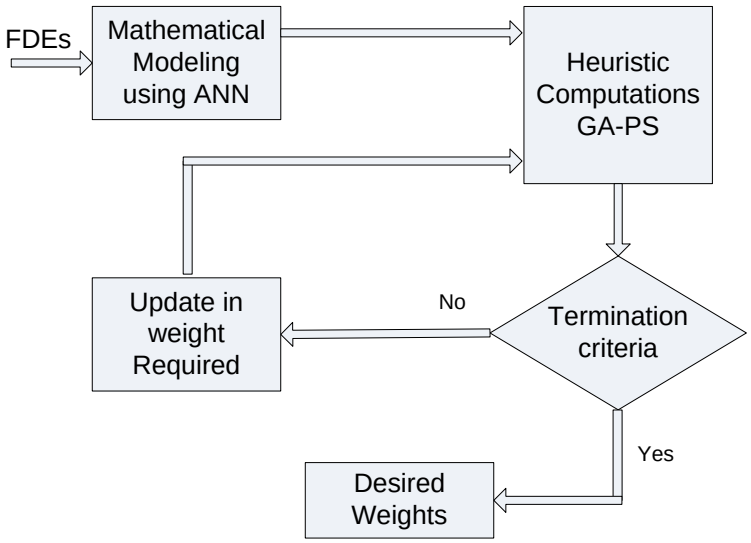



**Fig. 3.7** Procedure for finding desired weights

## 3.3 SUMMARY OF CHAPTER

This chapter presents the design methodology for solving fractional order system using feed-forward ANN supported with evolutionary computing and swarm intelligence techniques. The chapter starts with description of neural network mathematical modeling for FDEs. Neural networks architecture for different FOS are also provided The fitness function is formulated on the basis of unsupervised error that is used as objective function. The computational intelligence algorithms are used to optimize this objective function. Introductory material is provided global search techniques like GA, and PSO algorithms. The definitions of important subject terms used, generic flow diagram, the parameter setting for execution, and implementation procedure in steps are provided for such optimization tools. The chapter ends with brief description of local search methods like simulated annealing, pattern search and active set algorithms.

# CHAPTER 4

# EVOLUTIONARY COMPUTATIONAL INTELLIGENCE TECHNIQUES IN SOLVING FRACTIONAL ORDER SYSTEMS

In this chapter, the evolutionary computation technique based on genetic algorithm is applied for learning of weights for FDE-NN networks representing the differential equations. The proposed methodology was applied to a number of linear and nonlinear fractional order systems given by differential equation of fractional order. A number of problems have been solved and comparison of the results is also made with available exact solutions, as well as, other numerical solvers to validate the applicability of the designed scheme. Moreover, the scheme is also applied to solve fractional Riccati differential equation and well known Bagley-Torvik equation.

## 4.1 FRACTIONAL ORDER SYSTEM REPRESENTED WITH FRACTIONAL DIFFERENTIAL EQUATIONS

Fractional order systems governed with fractional differential equation has attracted so many authors due to its importance in diverse field of applied sciences and engineering ([3], [30]). In this section, the studies have been presented to solve numbers of problems associated with linear, nonlinear, ordinary fractional differential equations by our proposed scheme. Moreover, material mentioned here is taken mostly from a published work in [217].

### 4.1.1 PROBLEM I: LINEAR FRACTIONAL DIFFERENTIAL EQUATION WITH KNOWN EXACT SOLUTION

Consider the linear ordinary fractional differential equation ([164], [216]).

$$D^{\nu} y(t) = t^2 + \frac{2}{\Gamma(3-\nu)} t^{2-\nu} - y(t), \quad y(0) = 0 \quad 0 < \nu \leq 1 \tag{4.1.1}$$

with known exact solution and it is given as $y(t) = t^2$. The value of the fraction order derivative is taken as $\nu = 0.5$ in equation (4.1.1).

The problem is solved with help of FDE-NN networks optimized with GA algorithm. The number of neurons $m = 6$ are taken in FDE-NN networks, then the total number of $18$ unknown parameter ($\alpha_i$, $w_i$ and $b_i$) are to be adapted. These adaptive weights are restricted to real numbers between $(-10, 10)$. The initial population consists of a set of $160$ chromosomes divided into $8$ subpopulations. Each chromosome consists of $18$ genes equivalent to number of unknown weights. Input of the training set is taken from time $t \in (0, 1)$ with a step size of $0.1$. Therefore the error function, $e_j$, is formulated as

$$e_j = \frac{1}{11} \sum_{i=1}^{11} [D^{0.5} \hat{y}(t_i) - t_i^2 - \frac{2}{\Gamma(2.5)} t_i^{1.5} + \hat{y}(t_i)]^2 + [\hat{y}(0)]^2 \Big|_j, \; j = 1,2,3,\ldots, \tag{4.1.2}$$

where $j$ be the number of generation, $\hat{y}(t)$ and $D^{0.5}\hat{y}(t)$ are networks given as in (3.1.8), and (3.1.10), respectively. Our scheme runs iteratively in order to have the maximum fitness, e, for which to find the minimum of the value of function, $e_j$, with stoppage criteria as $2000$ number of generations or the value $e_j \leq 10^{-8}$ whichever comes earlier. One of the unknown weights learned by the algorithm is provided in Table 4.1. Using these weights in equation (3.1.8) one can obtain the solution for any input time $t$ between $0$ and $1$. Comparison of the results with exact solution is made and summarized also in Table 4.1. The results of our proposed scheme are also plotted in Fig. 4.1 for interval (0, 2) with step of $0.2$ along with exact solution. It can be seen from Table 4.1, as well as, Fig. 4.1 that the solution obtained by our algorithm is close to the exact solution. It proves the applicability of our approach for solving FDE equations.

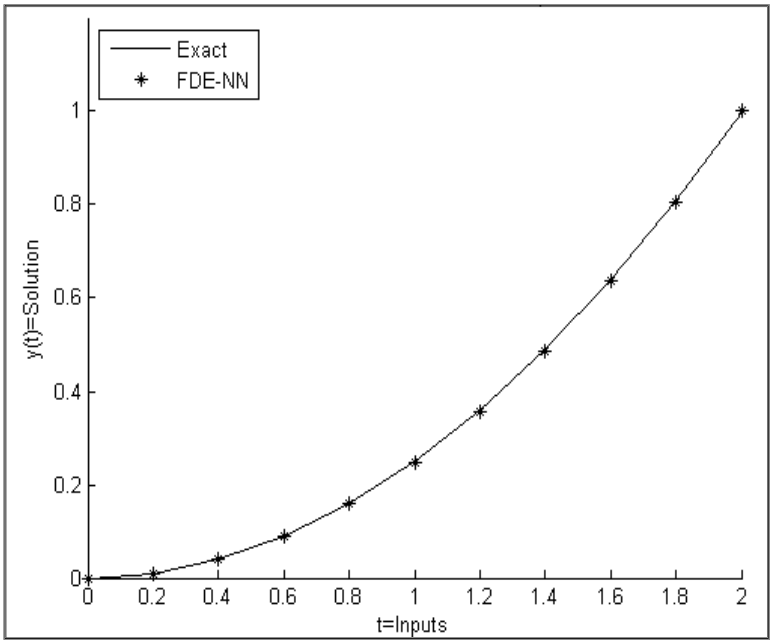



**Fig. 4.1** The result obtained by FDE-NN networks optimized with GA algorithm and exact solutions for t ∈ (0, 2).

**Table 4.1** Weights and Solution for Problem I.

| i | Unknown Weights | | | Time | Exact | FDE-NN | Absolute Error |
|---|---|---|---|---|---|---|---|
| | $w_i$ | $\alpha_i$ | $b_i$ | | | | |
| 1 | 0.1960 | 0.0830 | 0.2050 | 0.0 | 0.0000 | 0.0004 | 3.726e-04 |
| 2 | -1.3561 | 0.6221 | 0.4691 | 0.2 | 0.0400 | 0.0396 | 3.605e-04 |
| 3 | -0.8717 | -0.2375 | 0.9537 | 0.4 | 0.1600 | 0.1596 | 4.328e-04 |
| 4 | 0.9907 | 0.3870 | 0.7539 | 0.6 | 0.3600 | 0.3573 | 2.741e-03 |
| 5 | 0.3077 | -0.0155 | -0.2624 | 0.8 | 0.6400 | 0.6352 | 4.777e-03 |
| 6 | 0.0228 | -1.0413 | 0.2148 | 1.0 | 1.0000 | 1.0004 | 3.519e-04 |

### 4.1.2. PROBLEM II: LINEAR FRACTIONAL DIFFERENTIAL EQUATION WITH UNKNOWN EXACT SOLUTION AND AVAILABLE NUMERICAL SOLUTION

Consider the ordinary fractional differential equation

$$Dy(t) - \tfrac{11}{15} D^{1/2} y(t) + \tfrac{2}{15} y(t) = 0, \quad y(0) = 0, \tag{4.1.3}$$

with analytic solution obtained by direct approach method [1] given as

$$y(t) = a_1 \varepsilon_t(0, a_1^2) - a_2 \varepsilon_t(0, a_2^2) + a_1 \varepsilon_t(-\tfrac{1}{2}, a_1^2) - a_2 \varepsilon_t(-\tfrac{1}{2}, a_2^2), \tag{4.1.4}$$

where $a_1, a_2$ are the zeros of indicial polynomial of (4.1.3) and $\varepsilon_t$ is related to MLF function given in equation (2.2.10). The equation (4.1.3) can be a simplified form of composite fractional relaxation equation [220]. For $\nu = 1/2$ it can represent the unsteady motion of a particle accelerating in a viscous fluid under the action of gravity referred to as Basset problem ([220], [221]).

This problem has been simulated by FDE-NN networks (3.1.8) to (3.1.10) on a similar parameter setting as in previous problem, however, the input of the training set is chosen from time $t \in (0.1, 4)$ with a step size of $0.2$ and adaptive parameters are restricted to the range [-100,100]. The error function, $e_j$, can be formulated as

$$e_j = \tfrac{1}{20}\sum_{i=1}^{20}[D\hat{y}(t_i) - \tfrac{11}{15}D^{1/2}\hat{y}(t_i) + \tfrac{2}{15}\hat{y}(t_i)]^2 + [\hat{y}(0)]^2\Big|_j, \quad j = 1,2,\ldots, \tag{4.1.5}$$

One of the unknown weights learned by FDE-NN algorithm stochastically with value for error, $e_j = 8.85\times10^{-07}$, is given in Table 4.2. Using these weights in expression (3.1.8) one can find the solution to the equation for any input time $t$ between $0$ and $4$.

**Table 4.2** Weights Obtained by FDE-NN Networks Trained with GA Algorithm and Results for FDE in Problem II.

| i | Unknown Weights | | | T | Exact | FDE-NN | Numerical | Absolute Error | |
|---|---|---|---|---|---|---|---|---|---|
| | $w_i$ | $\delta_i$ | $b_i$ | | | | | *FDE-NN* | *Numerical* |
| 1 | 04.8188 | -46.7244 | -29.5226 | 0.5 | 0.1236 | 0.1234 | 0.1203 | 2.70E-04 | 3.34E-3 |
| 2 | 00.5509 | 61.4131 | -10.5329 | 1.0 | 0.1629 | 0.1633 | 0.1571 | 3.91E-04 | 5.83E-3 |
| 3 | -00.4554 | -16.6884 | -05.9407 | 2.0 | 0.2459 | 0.2461 | 0.2335 | 1.01E-04 | 1.25E-2 |
| 4 | -10.7147 | -53.8387 | -08.6134 | 3.0 | 0.3436 | 0.3424 | 0.3214 | 1.21E-03 | 2.22E-2 |
| 5 | -00.7676 | -11.8304 | -05.9951 | 3.5 | 0.3998 | 0.3990 | 0.3712 | 8.02E-04 | 2.86E-2 |
| 6 | 00.2721 | 03.7615 | -03.1966 | 4.0 | 0.4616 | 0.4616 | 0.4254 | 4.30E-05 | 3.61E-2 |

The analytic approximate solution can be determined using expression (4.1.4) for input time $t \in (0.1, 4)$ with a step size of $0.1$. We have also calculated the numerical solution by numerical technique developed by Podlubny based on successive approximation method [30]. The recursive relations are used for computations with

parameters $h = 0.001, n = 0,1,2,\ldots,4000.$ In order to compare the results on given inputs, Podlubny numerical, approximate analytical, FDE-NN solutions were obtained. Results are shown in Fig. 4.2 and given in Table 4.2.

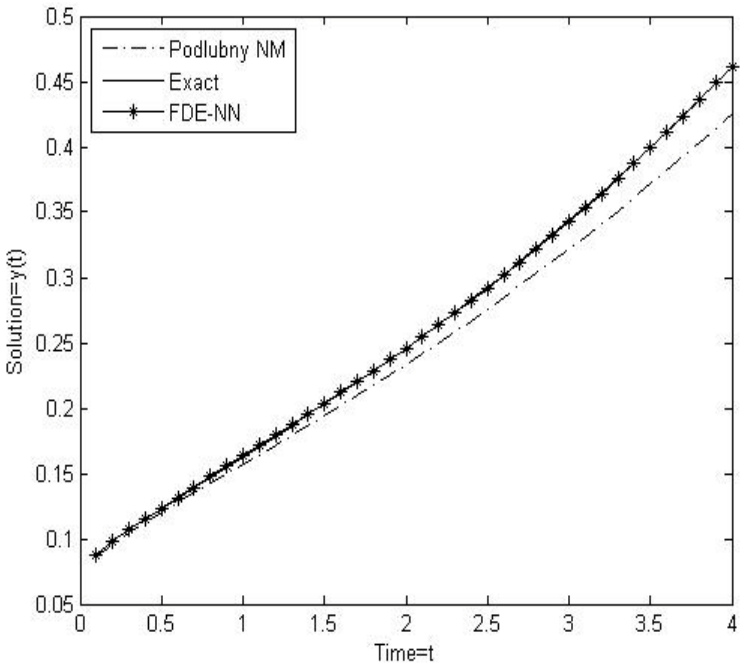



**Fig. 4.2** The Comparison of results obtained for interval t ∈ (0, 4).

It can be seen that our algorithm approximate the exact solutions more accurately than classical numerical method. For example, at time $t = 2$ the solution obtained by FDE-NN approach is $y(t) = 0.2459521$ and by Podlubny numerical techniques is $0.2334612$ whereas exact solution is $0.2429514$. The accuracy achieved by the FDE-NN method is of the order of $10^{-04}$ to $10^{-05}$ whereas it is of the order of $10^{-02}$ to $10^{-03}$ for Podlubny method.

### 4.1.3 PROBLEM III: NONLINEAR FRACTIONAL DIFFERENTIAL EQUATION

Consider a non-linear ordinary fractional differential equation given as [164]

$$D^{\nu} y = \frac{40320}{\Gamma(9-\nu)} t^{8-\nu} - 3\frac{\Gamma(5+\nu/2)}{\Gamma(5-\nu/2)} t^{4-\nu/2} + \frac{9}{4}\Gamma(\nu+1) + (\frac{3}{2}t^{\nu/2} - t^4)^3 - y^{3/2} \qquad (4.1.6)$$

with initial condition $y(0)=0$ for the case $0<\nu\le 1$. The exact solution for this equation is given as

$$y(t)=t^{8}-3t^{4+\nu/2}+\frac{9}{4}t^{\nu} \qquad \textbf{(4.1.7)}$$

The classical numerical technique used in problem II is unable to provide solution for such equations. However, modern deterministic methods with high computational cost can provide solution to (4.1.6), like fractional Adams method [223].

The simulation has been performed for this problem also and the results are summarized in Table 4.3 and plotted in Fig. 4.3 for inputs $t \in (0, 1)$ with step size *0.1*. The accuracy achieved is of the order of $10^{-02}$ to $10^{-03}$, whereas numerical result of the Adams scheme [30] at input $t=1$, for mesh size $h=0.1$ the error is $2.50\times10^{-01}$ and $-5.53\times10^{-03}$ for $\nu=0.25$ and $\nu=1.25$, respectively. It means that for such complex non-linear problems our designed method is still able to provide solution realiable and accurate.

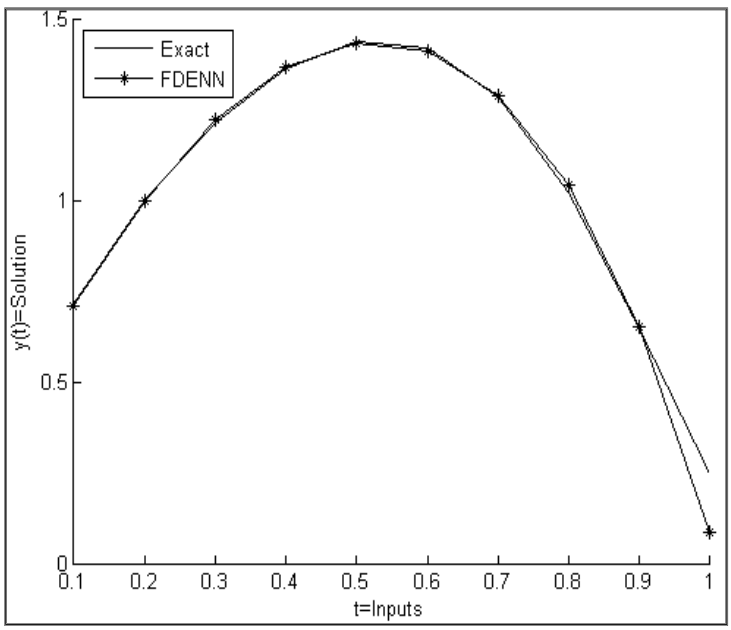



**Fig. 4.3** The result obtained by FDE-NN networks optimized with GA algorithm and exact solutions for t ∈ (0, 1).

**Table 4.3** Weights Obtained by FDE-NN Networks Trained with GA Algorithm and Results for FDE in Problem III.

| i | Unknown Weights | | | Time | Exact | FDE-NN | Absolute Error |
|---|---|---|---|---|---|---|---|
| | $w_i$ | $\delta_i$ | $b_i$ | | | | |
| 1 | 2.5205 | -0.4835 | -0.6029 | 0.1 | 0.7113 | 0.7066 | 0.0047 |
| 2 | 0.0156 | -0.2346 | -1.2299 | 0.2 | 1.0030 | 0.9994 | 0.0036 |
| 3 | -1.3029 | -1.1186 | 1.2410 | 0.3 | 1.2145 | 1.2198 | 0.0054 |
| 4 | 0.1110 | 1.3334 | 0.6163 | 0.4 | 1.3626 | 1.3656 | 0.0030 |
| 5 | 0.0268 | 0.7468 | -0.1811 | 0.5 | 1.4372 | 1.4317 | 0.0055 |
| 6 | 0.5211 | 0.2775 | 0.5408 | 0.6 | 1.4175 | 1.4095 | 0.0080 |
| 7 | -0.6929 | 0.9731 | 0.1722 | 0.7 | 1.2813 | 1.2857 | 0.0044 |
| 8 | 0.4450 | -0.1819 | -0.0018 | 0.8 | 1.0181 | 1.0420 | 0.0239 |

### 4.1.4 CONCLUSION

A stochastic computational approach has been developed for solving the FDEs using neural networks and genetic algorithm. This method has been successfully applied to different linear and non-linear ordinary FDE equations. It has been shown that proposed scheme approximates the exact solution with better accuracy than the standard numerical technique. An advantage of this method is that, unlike other numerical method, it gives approximate solution on the continuous finite time domain.

## 4.2 *NONLINEAR RICCATI DIFFERENTIAL EQUATIONS OF ARBITRARY ORDER*

In this section, the simulation results are presented for two different problems associated with Riccati differential equation of arbitrary order to prove the applicability of our algorithm. Comparative studies are also provided in this regard. The material provided is mostly based on published work [225].

### 4.2.1 PROBLEM I: FRACTIONAL RICCATI DIFFERENTIAL EQUATION

Consider the simple form of fractional Riccati differential equation

$$D^{\nu} y(t) = -y^2(t) + 1, \quad 0 \le t \le 1, \tag{4.2.1}$$

subject to the initial condition as $y(0) = 0$.

**First order case:** Let us take the value of order $\nu = 1$. Then the exact solution of the equation (4.2.1) is given as

$$y(t) = \frac{e^{2t} - 1}{e^{2t} + 1}. \tag{4.2.2}$$

The Adomian Decomposition method (ADM) can provide the approximate solution of equation (4.2.1) in the form of a rapidly convergent series with easily computable terms [226]. The solution in a series form is given as

$$\begin{aligned} y(t) = t &- 0.333333t^3 + 0.133333t^5 - 0.0539683t^7 + 0.0218695t^9 - \\ &0.00886324t^{11} + 0.00359213t^{13} - 0.00145583t^{15} + 0.000590027t^{17} \\ &- 0.000239129t^{19} + 0.0000969154t^{21}. \end{aligned} \tag{4.2.3}$$

In order to solve the problem with DE-NN networks using equations (3.1.5) and (3.1.6) and we have taken $m = 8$ number of neurons. Then the total number of 24 unknown parameter ($w_i$, $\alpha_i$, and $b_i$) are to be adapted. These adaptive weights are restricted to real numbers between $-10$ to $10$. The initial population consists of a set of 160 chromosomes or individuals divided into 8 subpopulations. Each chromosome or individual consist of 24 genes equivalent to number of weights. Input of the training set is taken form time $t \in (0, 1)$ with a step size of $0.1$. Its mean that function, $e_j$, is formulated as

$$e_j = \frac{1}{11}\sum_{i=1}^{11}[D\hat{y}(t_i) + \{\hat{y}(t_i)\}^2 - 1]^2 + \{\hat{y}(0)\}^2, \quad j = 1,2,3\ldots\ldots \tag{4.2.4}$$

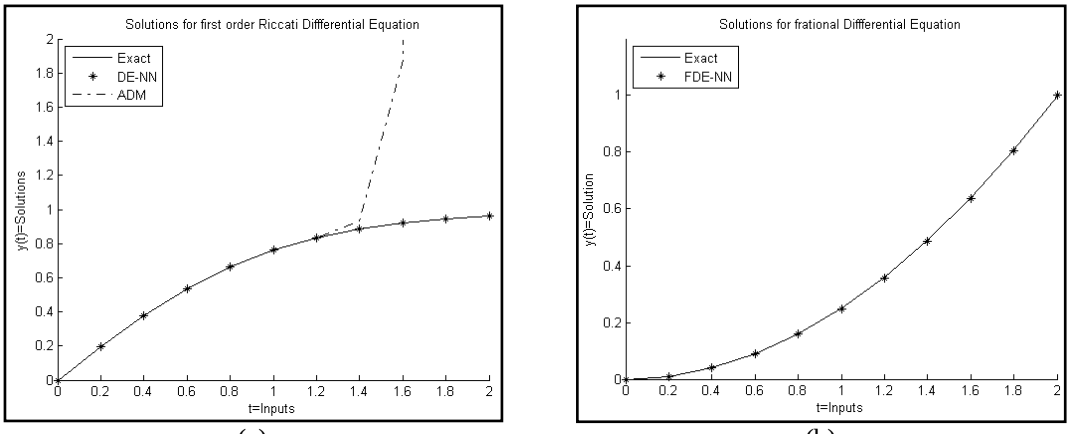
where $j$ is the number of generation, $\hat{y}(t)$ and $D\hat{y}(t)$ are networks provide as in equation (3.1.5), and (3.1.6), respectively with $m = 8$ number of neurons. Our scheme runs iteratively in order to find the minimum of the function, $e_j$, with stoppage criteria as 800 numbers of generations or the value for function $e_j \le 10^{-08}$ whichever

comes earlier. One of the unknown weights learned by DE-NN algorithm with the value of $e_j = 8.6429 \times 10^{-08}$ are provided in Table 4.4. Using these weights in expression (3.1.5) one can find the solution to this problem for any input time $t$ between $0$ and $1$.

**Table 4.4** Weights Obtained for ODE in Problem I.

| **Index** i | **Unknown Weights** | | | | | |
|---|---|---|---|---|---|---|
| | $w_i$ | | $\alpha_i$ | | $b_i$ | |
| | ν = 1 | ν = ½ | ν = 1 | ν = ½ | ν = 1 | ν = ½ |
| 1 | -1.1581 | 1.1585 | 0.0381 | 0.2443 | 2.3696 | 0.2119 |
| 2 | 1.6803 | 0.3749 | 1.8166 | 0.5420 | -0.0616 | 0.6061 |
| 3 | -2.5918 | 0.6269 | 0.0196 | -0.7132 | 5.3030 | 0.9942 |
| 4 | -2.3802 | 0.4679 | -1.2224 | 0.0372 | -2.3049 | -0.2808 |
| 5 | 4.0543 | 0.9160 | -0.3111 | 0.0247 | 1.1153 | -0.9092 |
| 6 | -0.8090 | 0.4100 | 3.0061 | 0.4812 | -1.6223 | 0.9103 |
| 7 | 2.2916 | 0.5152 | 0.9911 | 0.0292 | 0.4843 | -0.3812 |
| 8 | -0.3401 | -4.6597 | -1.8075 | -0.5627 | 2.7448 | 0.0154 |

We have obtained the analytic solution of represented by equations (4.2.2) and (4.2.3), as well as, by our proposed solution for the equation (4.2.1) for input time $t \in (0, 2)$ with a step size of $0.2$. Results are provided in Table 4.5 and it is shown graphically in Fig. 4.4. It can be seen that solutions obtained by GA algorithm hybridized simulating annealing are closest fit to the exact solutions.




(a) (b)

**Fig. 4.4** Graphical representation for the solution of ODE in problem I (a) for first order case ν = 1 and $0 \le t \le 2$ (b) for fractional order ν = 1/2 and $0 \le t \le 2$.

**Table 4.5** Comparison of Results for the Solution of ODE in problem I for First Order Case $\nu = 1$.

| **Time** t | **Exact** $y(t)$ | **ADM** $\varphi(t)$ | **DE-NN** $\phi(t)$ | **Error** | |
|---|---|---|---|---|---|
| | | | | $y(t)-\varphi(t)$ | $y(t)-\phi(t)$ |
| 0.0 | 0.0000 | 000.0000 | 0.0000 | 0.000000 | 2.208e-08 |
| 0.2 | 0.1974 | 000.1974 | 0.1971 | -2.559e-09 | 2.869e-04 |
| 0.4 | 0.3799 | 000.3799 | 0.3796 | -1.7847e-08 | 3.078e-04 |
| 0.6 | 0.5370 | 000.5370 | 0.5367 | -4.516e-08 | 2.997e-04 |
| 0.8 | 0.6640 | 000.6640 | 0.6637 | -2.373e-07 | 2.913e-04 |
| 1.0 | 0.7616 | 000.7616 | 0.7615 | -2.791e-05 | 1.171e-04 |
| 1.2 | 0.8337 | 000.8353 | 0.8338 | -0.001600 | -1.720e-04 |
| 1.4 | 0.8854 | 000.9356 | 0.8857 | -0.050200 | -3.663e-04 |
| 1.6 | 0.9217 | 001.8762 | 0.9220 | -0.954500 | -3.089e-04 |
| 1.8 | 0.9468 | 013.5716 | 0.9469 | -12.62400 | -4.592e-05 |
| 2.0 | 0.9640 | 126.6693 | 0.9639 | -125.7050 | 1.178e-04 |

**Fractional order case:** Let's take the value of order $\nu = 1/2$ then the expression (4.2.1) transforms into a fractional Riccati differential equation. ADM can be extended to provide the approximate analytic solution for such kind of the equation as well [226]. The analytic solution by this method can be given as

$$y(t) = 2x - 3.00901x^3 + 7.24332x^5 - 19.6157x^7 + 55.9634x^9 - 164.385x^{11} + 491.925x^{13} - 1491.22x^{15} + 4563.65x^{17} - 14068.5x^{19} + 43620.6x^{21}. \quad (4.2.5)$$

where for simplicity let $x = t^{1/2}$ in the above relation (4.2.5).

Similarly, the approximate analytic solution represented by expression (4.2.5) and stochastic proposed solution for the equation (4.2.1) are determined for input time $t \in (0,1)$ with a step size of $0.2$. The genes of best fitted chromosome in term of weights are given in Table 4.4 for order $\nu = 1/2$. The solution can be obtained using these weights and results are provided in Table 4.6 at some inputs along with the value of unsupervised error. It can be inferred from the results that our algorithm can also be applicable to provided approximate solutions of the Riccati differential equation of fractional order.

**Table 4.6** Solution of Equation in Problem 1 for Order $\nu = 1/2$.

| Time | FDE-NN | Value of Objective Function $e_j$ |
|---|---|---|
| 0.0 | 0.0505 | 8.475e-07 |
| 0.2 | 0.4047 | 7.623e-07 |
| 0.4 | 0.5507 | 3.031e-02 |
| 0.6 | 0.6178 | 1.652e-07 |
| 0.8 | 0.6594 | 1.086e-02 |
| 1.0 | 0.6992 | 2.464e-02 |

### 4.2.2 PROBLEM II: COMPLEX FRACTIONAL RICCATI DIFFERENTIAL EQUATION

A relatively complex example of Riccati differential equation of arbitrary order is taken to investigate the strengths and weaknesses of the proposed stochastic algorithm. Let us take another form of Riccati equations as

$$D^{\nu} y(t) = 2y(t) - y^2(t) + 1, \quad 0 \le t \le 1, \tag{4.2.6}$$

subjected to the initial condition as $y(0) = 0$.

The exact solution of the above equation for order $\nu = 1$ is given as

$$y(t) = 1 + \sqrt{2}\tanh(\sqrt{2}t) + \frac{1}{2}\log(\frac{\sqrt{2}-1}{\sqrt{2}+1}). \tag{4.2.7}$$

The analytic solution of (4.2.6) by ADM [226] for the case $\nu = 1/2$ is provided as

$$y(t) = 1.12838x + 2.0x^2 + 2.05121x^3 - 0.2732x^4 + \cdots\cdots + 80.4206x^{21} \tag{4.2.8}$$

where $x = t^{1/2}$.

This problem is solved for first order case as well as fractional order case on the same methodology adopted for problem I, but here we have taken $6$ number of neuron. DE-NN or FDE-NN method used to determine $18$ numbers of unknown weights. The stoppage criterion is that the value for function $e_j \le 10^{-06}$ or maximum number of generations is equal to 2000. The fitness function, $e_j$, used in said problem for the fractional order case $\nu = 1/2$ is given as

$$e_j = \frac{1}{11}\sum_{i=1}^{11}[D^{1/2}\hat{y}(t_i) + \{\hat{y}(t_i)\}^2 - 2\hat{y}(t_i) - 1]^2 + \{\hat{y}(0)\}^2, \quad j = 1,2,3\ldots, \tag{4.2.8}$$

where $\hat{y}(t)$, and $D^{1/2}\hat{y}$ are networks as provided in equation (3.1.8), and (3.1.10), respectively, using $m = 6$ number of neurons. Adaptive weights $\alpha_i, w_i$ and $b_i$ are restricted to real numbers between $-10$ and $10$ for the approximation of result. The unknown weights determined by our scheme are used in expression (3.1.5) and (3.1.8) for finding the solution of the equations for input timing between $0$ and $1$. Result are summarized in Table 4.7, where the order of the equation is taken as $\nu = 1$ and $\nu = 1/2$. It can be seen that the value of absolute error in first order case on average lies between $10^{-05}$ to $10^{-06}$, while for order $\nu = 1/2$ the value of unsupervised error, $e_j$, mostly lies around $10^{-03}$.

**Table 4.7** Results of Proposed Scheme for Solution of Equation in Problem II.

| T | First order Case ν = 1 | | | Fractional Order Case ν =1/2 | |
|---|---|---|---|---|---|
| | *Exact* $y(t)$ | *DE-NN* $\varphi(t)$ | *Error* $y(t)-\varphi(t)$ | *FDE-NN* | *Value of objective function* $e_j$ |
| 0.0 | 0.0000 | -0.0000 | 2.97e-09 | 0.0561 | 1.121e-07 |
| 0.1 | 0.1103 | 0.1103 | -3.14e-05 | 0.5610 | 1.226e-02 |
| 0.2 | 0.2420 | 0.2420 | -7.01e-06 | 0.9121 | 2.199e-03 |
| 0.3 | 0.3951 | 0.3950 | 5.58e-05 | 1.1594 | 1.876e-04 |
| 0.4 | 0.5678 | 0.5677 | 9.87e-05 | 1.3369 | 2.510e-03 |
| 0.5 | 0.7560 | 0.7559 | 6.96e-05 | 1.4671 | 2.247e-03 |
| 0.6 | 0.9536 | 0.9536 | -3.07e-05 | 1.5652 | 1.045e-06 |
| 0.7 | 1.1529 | 1.1531 | -1.40e-04 | 1.6414 | 2.398e-03 |
| 0.8 | 1.3464 | 1.3465 | -1.80e-04 | 1.7024 | 3.333e-03 |
| 0.9 | 1.5269 | 1.5270 | -1.20e-04 | 1.7528 | 1.863 e-03 |
| 1.0 | 1.6895 | 1.6896 | -6.20e-05 | 1.7957 | 2.337 e-03 |

### 4.2.3 CONCLUSION

On the basis of the simulation results obtained in the last section it can be concluded that nonlinear Riccati differential equation of arbitrary order can be solved with

evolutionary computation based on genetic algorithm. Combination of neural network aided with GA can provide stochastically the solution fairly close to the exact solution. Our proposed evolutionary computing approaches can also be applied on complex nonlinear differential equation of arbitrary order as well.

## 4.3 FRACTIONAL ORDER SYSTEM OF BAGLEY-TORVIK EQUATION

In this section, the simulation results are presented for three different fractional order systems represented by Bagley-Torvik equation which is famous Multi-term fractional differential equation. In order to prove the applicability of the designed scheme for such systems, we have considered the equation in problem I for which exact solution is available. Moreover, the statistical analysis of results is also carried out to verify and validate reliability of the algorithm. In problem II, we have tested the proposed methodology on Bagley-Torvik equation for which PMA technique fails to determine the results due to non-homogeneous initial conditions. In problem III, the strength and weakness of our scheme is analyzed by taking such equation for which the exact solution is not known, however, its numerical solution is available. The necessary discussion on strength and weakness of the proposed scheme is also provided for the results determined. The material provided in this section is mostly from the published work in [227].

### 4.3.1 PROBLEM I: BAGLEY-TORVIK EQUATION WITH KNOWN EXACT SOLUTION

Consider the Bagley-Torvik equation ([99], [228])

$$\frac{d^2y(t)}{dt^2}+\frac{d^{3/2}y(t)}{dt^{3/2}}+y(t)=2+4\sqrt{t/\pi}+t^2,\quad 0<t\le 1 \tag{4.3.1}$$

Subject to the initial condition and boundary condition as

$$y(0)=y'(0)=0,\quad y(1)=1,\quad y'(1)=2 \tag{4.3.2}$$

The exact solution of the equation is given as $y(t)=t^2$.

Mathematical modeling of the above equation is done with FDE-NN networks represented by equations (3.1.8) to (3.1.10) by taking $10$ numbers of neuron resulting in $30$ numbers of unknown parameters or weights. These weights are restricted to real numbers between $-10$ to $10$. The initial population consists of a set of $200$ individuals, which is divided into $10$ subpopulations each with $20$ numbers of individuals. Each individual consists of $30$ genes, which is equivalent to number of unknown parameters of FDE-NN. Input of the training set is taken from $t \in (0,1)$. Therefore the fitness function is formulated as

$$e_j = \left. \begin{array}{l} \frac{1}{2}\sum_{i=1}^{2}[\frac{d^2\hat{y}(t_i)}{dt_i^{\,2}} + \frac{d^{3/2}\hat{y}(t_i)}{dt_i^{\,3/2}} + \hat{y}(t_i) - 2 - 4\sqrt{t_i/\pi} - t_i^{\,2}]^2 \\ +[\hat{y}(0)]^2 + [\hat{y}(1)-1]^2 + [\hat{y}'(0)]^2 + [\hat{y}'(1)-2] \end{array} \right|_j \quad j = 1,2,3,\ldots, \qquad (4.2.3)$$

where, $j$ is the number of generations, $\hat{y}(t)$, $d^2\hat{y}/dt^2$ and $d^{3/2}\hat{y}/dt^{3/2}$ are networks provided as in equations (3.1.8), (3.1.9) and (3.1.10), respectively. Our scheme runs iteratively in order to find the minimum for function, $e_j$, with termination criteria as $3000$ number of generations or the value for $e_j \le 10^{-04}$ whichever comes earlier. Then, the best individual found so far is used in rapid local search algorithm as a starting point for further refinements. The parameter setting of GA and PS are given in Table 4.8.

One set of the weights of FDE-NN learned stochastically by PS and GA algorithms are provided in Table 4.9, while for GA-PS algorithm weights are given in Table 4.10. Using these weights in (3.1.8) one can find the solution to this problem for any input time $t$ between $0$ and $1$.

The results obtained by our scheme and other numerical solvers like Podlubny matrix approach (PMA) [102] and reported results of He's variational iteration method (VIM) [99] for inputs between $0$ and $1$ with a step of $0.1$ are given in Table 4.11. In order to determine the results by PMA technique, the library functions provided by Podlubny at MATLAB central file exchange are used [229]. Following parameter

setting is employed for PMA technique is as follows: The value of Fractional order derivative $\nu = 1.5$. Set the value for constant coefficients $A = B = C = 1$. The discretization step is taken as $h = 0.01$. Total number of time steps is taken as $100$.

It can be seen that the result obtained by our scheme is in good agreement with the state of art numerical solvers

**Table 4.8** Parameters Setting for Algorithms.

| **GA** | | **PS** | |
|---|---|---|---|
| Parameters | Setting | Parameters | Setting |
| Population Creation | Constrain dependant | Poll method | GPS Positive basis 2N |
| Scaling faction | Rank | Polling order | Consecutive |
| Selection function | Stochastic Uniform | Mesh Accelerator | Off |
| Crossover fraction | 0.75 | Mesh Rotae/Scale | On |
| Crossover fuction | Scattered | Mesh expansion factor | 2.0 |
| Mutation | Adaptive feasible | Mesh Contraction factor | 0.5l |
| Elite count | 2 | Cache Tolerance | EPS |
| Initial Penalty | 10 | Initial Penalty | 10 |
| Penalty factor | 100 | Penalty factor | 100 |
| Migration fraction | 0.2 | Max Iteration | 3000 |
| Migration interval | 20 | Max. function evolutions | 60000 |

**Table 4.9** Weights of FDE-NN Networks Trained by PS and GA Algorithms for Problem I.

| $I$ | $w_i$ | | $\alpha_i$ | | $\beta_i$ | |
|---|---|---|---|---|---|---|
| | *PS* | *GA* | *PS* | *GA* | *PS* | *GA* |
| 1 | 0.227249 | 0.014313 | 0.731960 | -1.16770 | 0.769316 | 0.046808 |
| 2 | 0.031563 | -0.15713 | 0.708549 | 0.554705 | 0.871064 | -0.73877 |
| 3 | 0.121262 | -0.29925 | -7.73928 | -0.16469 | 0.504503 | -0.12463 |
| 4 | 2.583099 | 0.000465 | -4.61892 | -0.74905 | -4.77791 | 0.050833 |
| 5 | 0.564499 | 1.169105 | 0.941493 | -0.14210 | 0.702424 | -0.00566 |
| 6 | 0.659389 | -1.30293 | 0.964177 | 0.334789 | 0.748012 | 0.248267 |
| 7 | 0.434814 | -0.40253 | 0.988058 | 0.284607 | 0.199868 | 0.648494 |
| 8 | -0.78165 | 0.906189 | 0.711952 | -0.07749 | 0.918857 | 0.372823 |
| 9 | -0.19301 | 0.348033 | 0.595146 | 0.076874 | 1.270213 | -1.37687 |
| 10 | 0.192005 | 0.975022 | 0.223502 | 0.271679 | 0.873072 | 1.502332 |

**Table 4.10** Weights of FDE-NN Networks Trained by GA-PS Method for Problem I.

| i | $w_i$ | $\alpha_i$ | $\beta_i$ |
|---|---|---|---|
| 1 | 0.076813 | -1.16770 | 0.042905 |
| 2 | -0.15713 | 0.304705 | -0.73877 |
| 3 | 3.669499 | -0.16469 | -8.12463 |
| 4 | 0.000465 | -0.74905 | 0.050833 |
| 5 | 1.169105 | -0.14210 | -0.00566 |
| 6 | -1.30293 | 0.209789 | 0.248252 |
| 7 | -0.90253 | 0.284607 | 0.648494 |
| 8 | -5.09381 | 3.922500 | -7.62717 |
| 9 | -4.64415 | 0.014374 | -9.37687 |
| 10 | 0.943742 | 0.271679 | 1.502332 |

**Table 4.11** Comparison of Results for the Solution of Problem I.

| | | ŷ(t) | | | | | \|y(t)- ŷ(t)\| | | |
|---|---|---|---|---|---|---|---|---|---|
| T | y(t) | *PS* | *GA* | *GA-PS* | *PMA* | *HVIM* | *GA-PS* | *PMA* | *HVIM* |
| 0.0 | 0.00 | 2.73e-02 | 0.06730 | 0.03346 | 0.00000 | 0.00000 | 3.34E-2 | 0.0000 | 0.00000 |
| 0.1 | 0.01 | 0.012392 | 0.08925 | 0.04437 | 9.29e-3 | 0.01005 | 3.43E-2 | 7.04E-4 | 5.48E-5 |
| 0.2 | 0.04 | 0.049646 | 0.12802 | 0.07338 | 0.03892 | 0.04063 | 3.33E-2 | 1.07E-3 | 6.31E-4 |
| 0.3 | 0.09 | 0.111143 | 0.18375 | 0.12044 | 0.08873 | 0.09266 | 3.04E-2 | 1.26E-3 | 2.66E-3 |
| 0.4 | 0.16 | 0.196074 | 0.25676 | 0.18573 | 0.15867 | 0.16748 | 2.57E-2 | 1.32E-3 | 7.48E-3 |
| 0.5 | 0.25 | 0.303374 | 0.34756 | 0.26961 | 0.24871 | 0.26679 | 1.96E-2 | 1.28E-3 | 1.67E-2 |
| 0.6 | 0.36 | 0.431620 | 0.45684 | 0.37262 | 0.35883 | 0.39277 | 1.26E-2 | 1.16E-3 | 3.22E-2 |
| 0.7 | 0.49 | 0.578880 | 0.58548 | 0.49549 | 0.48903 | 0.54806 | 5.49E-3 | 9.67E-4 | 5.80E-2 |
| 0.8 | 0.64 | 0.742531 | 0.73455 | 0.63911 | 0.63928 | 0.73588 | 8.80E-4 | 7.14E-4 | 9.58E-2 |
| 0.9 | 0.81 | 0.919015 | 0.90530 | 0.80457 | 0.80958 | 0.96007 | 5.42E-3 | 4.12E-4 | 1.50E-1 |
| 1.0 | 1.00 | 1.103535 | 1.09922 | 0.99308 | 0.99993 | 1.22519 | 6.91E-3 | 6.83E-5 | 2.25E-1 |

Moreover, the behavior of the derivative of the solution is also analyzed with the same individuals as given in Table 4.9 and Table 4.10. The results for derivative of the solutions are provided for inputs between $0$ and $1$ with a step of $0.1$ by PS, GA and GA-PS algorithms in Table 4.12. It is quite evident that the accuracy level lies in the range of $10^{-02}$ to $10^{-03}$.

Before moving toward the statistical analysis of our designed scheme, it is necessary to mention that entire surrogate model of the algorithm is based on MLF function. Its

generic form is given in expression (2.2.5). However, the difficulty faced in calculation of MLF function is due to its complex nature. This issue is tackled in our simulation by use of MATLAB code provided by Podlubny at MATLAB Central File Exchanges to determine the value of MLF function for desired inputs [230].

**Table 4.12** Comparison of Results for Derivative of the Solution for Problem I.

| T | $\hat{y}'(t)$ | $\hat{y}'(t)$ | | | $\lvert y'(t) - \hat{y}'(t) \rvert$ | | |
|---|---|---|---|---|---|---|---|
| | | *PS* | *GA* | *GA-PS* | *PS* | *GA* | *GA-PS* |
| 0.0 | 0.00 | 0.00267 | 0.13526 | 0.017687 | 2.67e-3 | 1.80e-2 | 1.76e-2 |
| 0.1 | 0.20 | 0.24906 | 0.30350 | 0.199858 | 4.90e-2 | 9.07e-2 | 1.41e-4 |
| 0.2 | 0.40 | 0.49494 | 0.47211 | 0.380284 | 9.49e-2 | 1.20e-2 | 1.97e-2 |
| 0.3 | 0.60 | 0.73366 | 0.64301 | 0.561324 | 0.13366 | 1.34e-2 | 3.86e-2 |
| 0.4 | 0.80 | 0.96318 | 0.81805 | 0.745050 | 0.16318 | 3.14e-2 | 5.49e-2 |
| 0.5 | 1.00 | 1.18047 | 0.99909 | 0.933359 | 0.18047 | 2.07e-2 | 6.66e-2 |
| 0.6 | 1.12 | 1.38125 | 1.18798 | 1.128047 | 0.18125 | 6.03e-2 | 7.19e-2 |
| 0.7 | 1.40 | 1.55960 | 1.38659 | 1.330850 | 0.15960 | 1.80e-2 | 6.91e-2 |
| 0.8 | 1.60 | 1.70751 | 1.59685 | 1.543479 | 0.10751 | 9.07e-4 | 5.65e-2 |
| 0.9 | 1.80 | 1.81424 | 1.82073 | 1.767629 | 0.01424 | 1.20e-2 | 3.23e-2 |
| 1.0 | 2.00 | 1.86549 | 2.06033 | 2.004987 | 0.13450 | 1.34e-2 | 4.98e-3 |

Training of the weights for neural networks of the equation is highly stochastic nature. It is necessary to have statistical analysis of the results obtained by FDE-NN algorithms. A total of 125 independents runs are executed for each stochastic optimizer. These optimizers are PS, GA and GA hybrid with PS (GA-PS) algorithms. The values of the best 100 runs of algorithm in term of absolute error $| y(t) - \hat{y}(t) |$ are provided in Table 4.13 for input at 0 and 1 while the absolute error for the derivative of the solution is given in Table 4.14 on the same input timing. The value of Mean and Standard deviation (STD) are also calculated for solution and its derivative and results are provided in Table 4.13 and Table 4.14, respectively.

The value of unsupervised error, $e_j$, of the equation in ascending order is plotted against each independent run of the algorithm in Fig. 4.5. The values of the absolute error for the solution at 0 and 1 are plotted in Fig. 4.6(a) and Fig 4.6(b), respectively,

against 100 independent runs of stochastic solvers. Moreover, results for derivative of the solutions are also plotted in Fig. 4.7(a) and Fig. 4.7(b). It can be seen from Fig. 4.5, Fig. 4.6, and Fig. 4.7, that the best results are obtained by the use of GA-PS algorithm. The same inference can be drawn for the results provided in Table 4.13 and Table 4.14.

**Table 4.13** Statistical Parameter Based on Value of Absolute Error ($|\, y(t) - \hat{y}(t)\,|$).

| T | FDE-NN Networks optimized with | Best | Worst | Mean | STD |
|---|---|---|---|---|---|
| 0 | PS | 2.36e-2 | 1.00232 | 0.32789 | 0.24679 |
| | GA | 4.87e-2 | 0.18695 | 0.14635 | 3.25e-2 |
| | GA-PS | 3.11e-3 | 0.17193 | 0.12111 | 4.14e-2 |
| 1 | PS | 2.35e-3 | 0.82603 | 0.24834 | 0.20391 |
| | GA | 4.61e-4 | 7.19e-2 | 3.36e-2 | 2.12e-2 |
| | GA-PS | 1.06e-4 | 6.87e-2 | 3.31e-2 | 2.00e-2 |

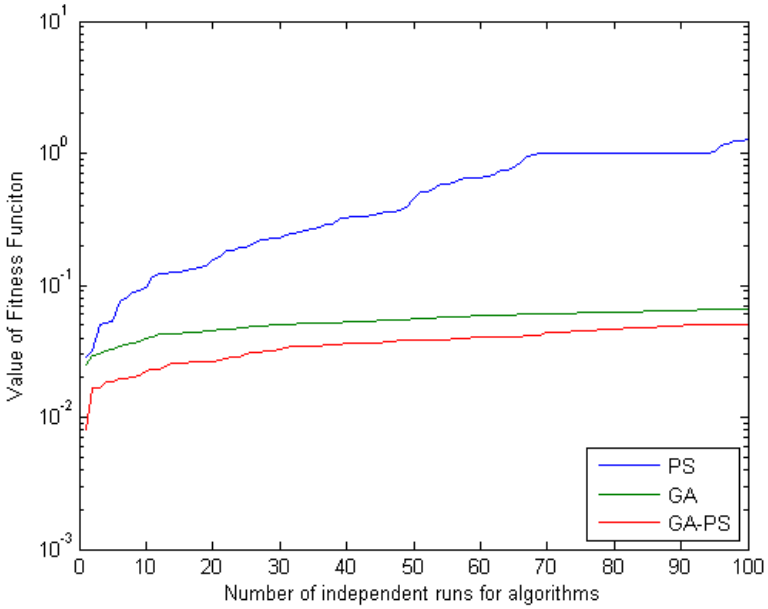



**Fig. 4.5** The value of the unsupervised error for the equation obtained by for different stochastic numerical solvers

**Table 4.14** Statistical Parameter Based on Value of Absolute Error ($|\, y'(t) - \hat{y}'(t)\,|$).

| T | FDE-NN Networks optimized with | Best | Worst | Mean | STD |
|---|---|---|---|---|---|
| 0 | PS | 2.91e-2 | 0.65725 | 0.27233 | 0.19051 |
| | GA | 6.13e-6 | 6.63e-2 | 2.99e-2 | 1.78e-2 |
| | GA-PS | 8.31e-4 | 5.82e-2 | 2.90e-2 | 1.70e-2 |
| 1 | PS | 9.59e-3 | 1.00357 | 0.40912 | 0.28778 |
| | GA | 1.48e-3 | 0.12491 | 7.54e-2 | 3.35e-2 |
| | GA-PS | 2.62e-2 | 0.11201 | 6.17e-2 | 3.37e-2 |

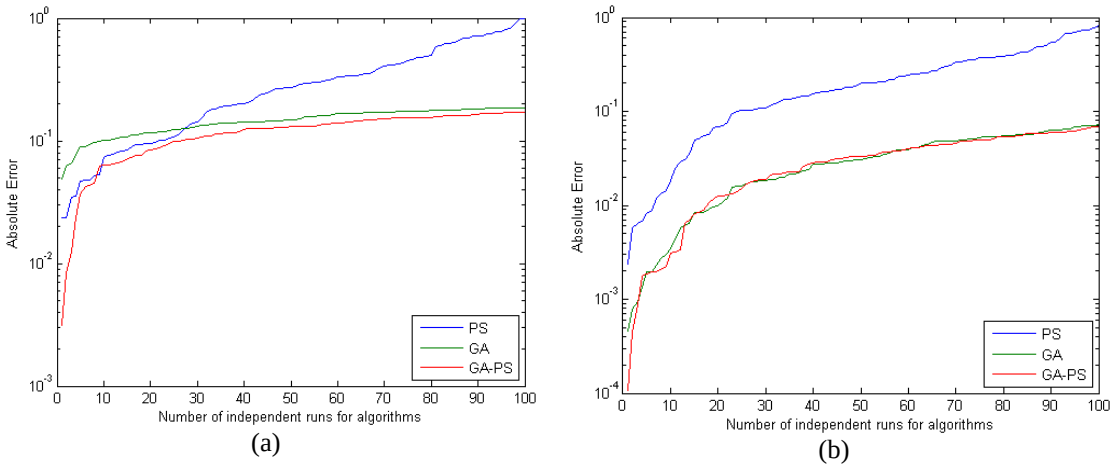



**Fig. 4.6** Comparison of FDE-NN for different stochastic numerical solvers (a) and (b) the value of |y(t)-ŷ(t| at t = 0 and t = 1, respectively.

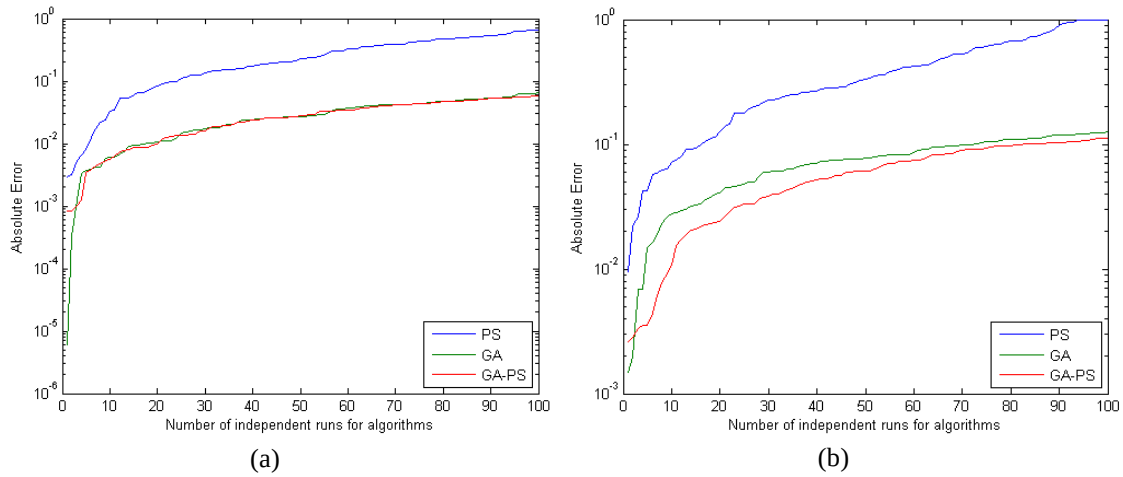



**Fig. 4.7** Comparison of FDE-NN for different stochastic numerical solvers (a) and (b) are for absolute error for derivative of solution at t = 0 and t = 1, respectively.

### 4.3.2 PROBLEM II: BAGLEY-TROVIK EQUATION WITH INHOMOGENEOUS INITIAL CONDITIONS

Another fractional order system of Bagley-Torvik equation is chosen to investigate the strengths and weaknesses of the proposed stochastic algorithm. Consider the Bagley-Torvik equation ([98], [100])

$$A\frac{d^2y(t)}{dt^2}+B\frac{d^{3/2}y(t)}{dt^{3/2}}+Cy(t)=C(t+1),\quad 0<t\le 1, \tag{4.3.4}$$

with initial condition as $y(0)=y'(0)=1$. It has the exact solution given as

$$y(t)=t+1. \tag{4.3.5}$$

This problem can be made simpler by use of following substitution

$$Y(t)=y(t)-1-t, \tag{4.3.6}$$

then the equation (4.3.4) will be

$$A\frac{d^2Y(t)}{dt^2}+B\frac{d^{3/2}Y(t)}{dt^{3/2}}+CY(t)=0,\quad Y(0)=Y'(0)=0. \tag{4.3.7}$$

This problem is solved on the same methodology adopted for previous example. However, the problem specific fitness function for equation (4.3.4) by taking the values of constant coefficient as unity can be formulated as

$$e_j=\frac{1}{5}\sum_{i=1}^{5}\left(\frac{d^2\hat{y}(t_i)}{dt_i^{\,2}}+\frac{d^{3/2}\hat{y}(t_i)}{dt_i^{\,3/2}}+\hat{y}(t_i)-(t_i+1)\right)^2\Bigg|_j+\frac{1}{2}\left([\hat{y}(0)]^2+[\frac{d}{dt}\hat{y}(0)-1]^2\right),\ j=1,2,3,..., \tag{4.3.8}$$

where, $j$ is the generations index, $\hat{y}(t)$, $d^2\hat{y}/dt^2$ and $d^{3/2}\hat{y}/dt^{3/2}$ are FDE-NN networks provided as in equations (3.1.8), (3.1.9) and (3.1.10), respectively. One of the unknown set of weights of FDE-NN networks trained stochastically by using PS and GA algorithm are given in Table 4.15 while for GA-SA weights are provided in Table 4.16. Using these weights the results are determined and is summarized in

Table 4.17. It can be inferred from the table that best results are obtained by FDE-NN networks trained by GA-PS algorithm. The average accuracy achieved in the given scheme lies in the range of $10^{-03}$ to $10^{-04}$.

**Table 4.15** Weights of FDE-NN Networks Trained by PS and GA Algorithms for Problem II.

| $i$ | $w_i$ | | $\alpha_i$ | | $\beta_i$ | |
|---|---|---|---|---|---|---|
| | *PS* | *GA* | *PS* | *GA* | *PS* | *GA* |
| 1 | -7.5259 | 0.45094 | 0.46874 | -0.6587 | 0.769316 | -1.1923 |
| 2 | -7.7316 | -0.0968 | 0.90230 | -0.5674 | 0.871064 | 0.67735 |
| 3 | -5.9519 | -0.3544 | -6.6940 | 0.51534 | 0.504503 | -0.7461 |
| 4 | -1.5897 | 1.75859 | 0.43284 | -0.1467 | -4.77791 | 0.33795 |
| 5 | 0.20093 | -0.8300 | 0.13829 | 0.14195 | 0.702424 | -1.7544 |
| 6 | 0.72030 | 0.05750 | 0.37976 | 0.49843 | 0.748012 | -0.8822 |
| 7 | -5.4554 | 1.10507 | 0.37135 | 1.51126 | 0.199868 | -0.1709 |
| 8 | -7.4683 | 1.55824 | 6.05083 | -0.7505 | 0.918857 | -1.0854 |
| 9 | -7.8736 | 0.24057 | 0.95588 | 0.22079 | 1.270213 | 1.82921 |
| 10 | -6.9500 | -0.1263 | 0.59958 | -0.2013 | 0.873072 | 0.47301 |

**Table 4.16** Weights of FDE-NN Networks Trained by GA-PS Method for Problem II.

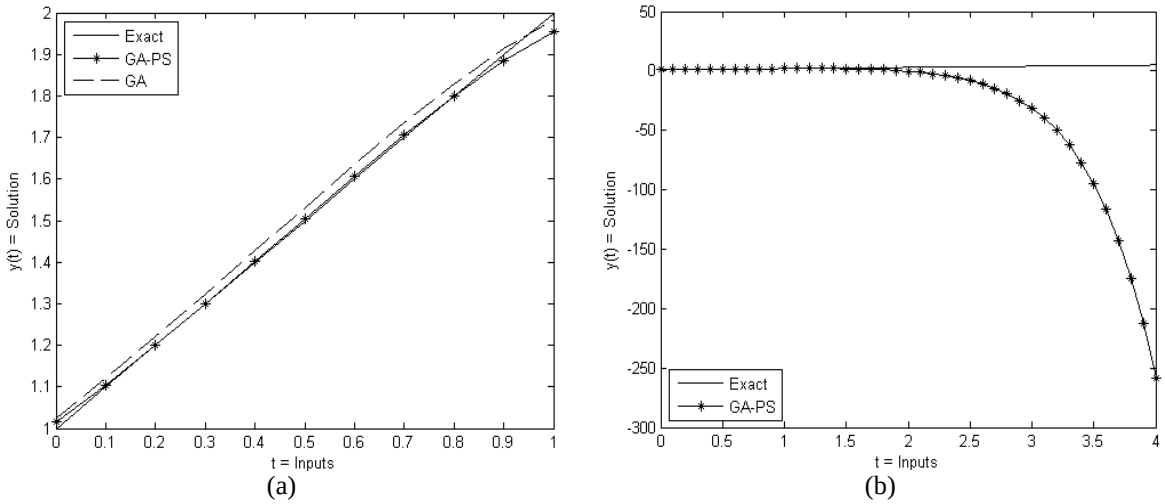
| i | $w_i$ | $\alpha_i$ | $\beta_i$ |
|---|---|---|---|
| 1 | 0.45094 | -0.6587 | 0.48045 |
| 2 | -0.0968 | -0.5674 | 0.84448 |
| 3 | -0.3544 | 0.51534 | 0.13803 |
| 4 | 1.75884 | -0.1467 | 0.57994 |
| 5 | -4.5800 | 0.14171 | 0.66264 |
| 6 | 0.05750 | 0.49062 | 0.68844 |
| 7 | 1.10702 | 1.50930 | 0.00385 |
| 8 | 1.55824 | -0.7505 | -5.9491 |
| 9 | 0.24057 | 0.22079 | 0.83593 |
| 10 | -0.1263 | -0.2013 | 0.63207 |

The results computed for the algorithm are also plotted graphically in Fig. 4.8. In the Fig. 4.8(a), the comparison of the results are given from exact solution on interval $(0, 1)$ with step of $0.1$. Moreover, the results are also plotted for larger interval $(0, 4)$

in Fig. 4.8(b) by using same weights and it can be seen that error starts to grow for inputs larger than 1 and it tends toward divergence for inputs greater than 2 . One can increase the accuracy slightly on large intervals by taking the large number of neurons in FDE-NN networks or by the training of networks for large input span or increasing the number of steps. However, it is worth mentioning that in these cases computational complexity will increase exponentially.

**Table 4.17** Comparison of Results for Problem II.

| | | $\hat{y}(t)$ | | | $\lvert y(t)-\hat{y}(t)\rvert$ | | |
|---|---|---|---|---|---|---|---|
| **t** | **y(t)** | PS | GA | GA-PS | PS | GA | GA-PS |
| 0.0 | 1.00 | 0.691604 | 1.024862 | 1.016007 | 0.30839 | 2.30e-2 | 1.60e-2 |
| 0.1 | 1.10 | 0.623749 | 1.121206 | 1.104733 | 0.47625 | 2.69e-2 | 4.73e-3 |
| 0.2 | 1.20 | 0.859697 | 1.220821 | 1.199804 | 0.34030 | 3.13e-2 | 1.95e-4 |
| 0.3 | 1.30 | 1.122266 | 1.323041 | 1.299333 | 0.17773 | 3.45e-2 | 6.66e-4 |
| 0.4 | 1.40 | 1.337736 | 1.426952 | 1.401629 | 6.22e-2 | 3.45e-2 | 1.62e-3 |
| 0.5 | 1.50 | 1.501839 | 1.531330 | 1.504972 | 1.83e-3 | 2.87e-2 | 4.97e-3 |
| 0.6 | 1.60 | 1.628907 | 1.634569 | 1.607429 | 2.89e-2 | 1.36e-2 | 7.42e-3 |
| 0.7 | 1.70 | 1.734458 | 1.734591 | 1.706705 | 3.44e-2 | 1.49e-2 | 6.70e-3 |
| 0.8 | 1.80 | 1.830520 | 1.828738 | 1.799987 | 3.05e-2 | 2.30e-2 | 1.27e-5 |
| 0.9 | 1.90 | 1.925308 | 1.913640 | 1.883785 | 2.53e-2 | 2.69e-2 | 1.62e-2 |
| 1.0 | 2.00 | 2.024099 | 1.985057 | 1.953762 | 2.40e-2 | 3.13e-2 | 4.62e-2 |




(a) (b)

**Fig. 4.8** Comparison of the results (a) is for interval (0, 1) and (b) is for interval (0, 4).

### 4.2.3 PROBLEM III: BAGLEY-TORVIK EQUATION WITH UNKNOWN EXACT SOLUTION

In this example, the fractional order system based on Bagley-Torvik equation is taken for which the exact solution is not available. However, its numerical solutions are obtained by different state of art solvers.

Consider the fractional order system ([30], [102])

$$\frac{d^2y(t)}{dt^2}+\frac{1}{2}\frac{d^{3/2}y(t)}{dt^{3/2}}+\frac{1}{2}y(t)=f(t),\quad 0<t \tag{4.3.9}$$

where

$$f(t)=\begin{cases}8, & (0\le t\le 1)\\ 0, & (t>1)\end{cases} \tag{4.3.10}$$

$$y(0)=y'(0)=0$$

The problem is also solved on the same manner as the previous one. However, the objective function used in this case can be written as

$$e_j=\frac{1}{10}\sum_{i=1}^{10}\left(\frac{d^2\hat{y}(t_i)}{dt_i^{\,2}}+\frac{1}{2}\frac{d^{3/2}\hat{y}(t_i)}{dt_i^{\,3/2}}+\frac{1}{2}\hat{y}(t_i)-8\right)^2\Bigg|_j,\ j=1,2,3,\ldots, \tag{4.3.11}$$
$$+\frac{1}{2}\left([\hat{y}(0)]^2+[\frac{d}{dt}\hat{y}(0)]^2\right)$$

The aim of our algorithm is to tune weights for which the value of fitness function as given in (4.3.11) is at its minimum. One such set of unknown weights of FDE-NN networks trained stochastically using PS and GA is provided in Fig 4.9(a) and Fig. 4.9(b) respectively, while for GA-SA that are given in Fig. 4.10. The results obtained based on these weights are summarized in Table 4.18. It also includes the results of PMA algorithm obtained with same set of parameters as taken in problem I. It can be inferred that the given scheme can perform comparable to the state-of-art solvers.

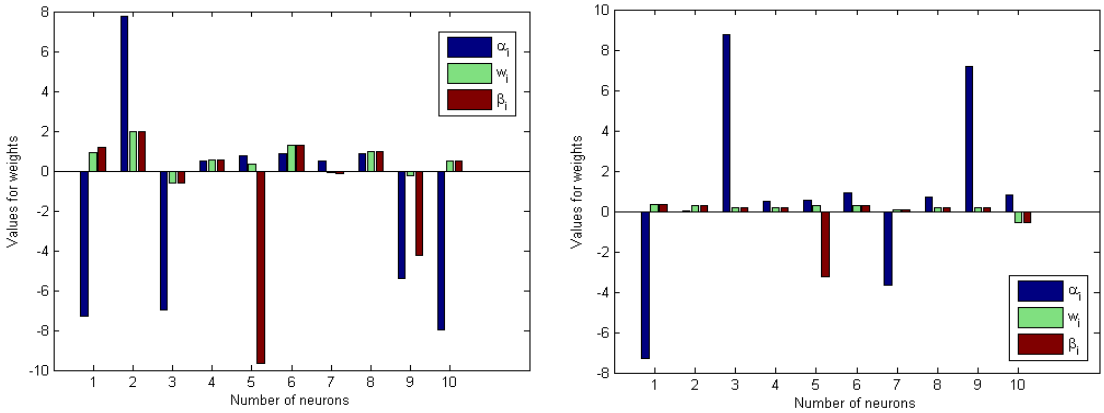



**Fig. 4.9** The weights trained for FDE-NN networks for example III, (a) is for PS technique (b) is for GA algorithm.

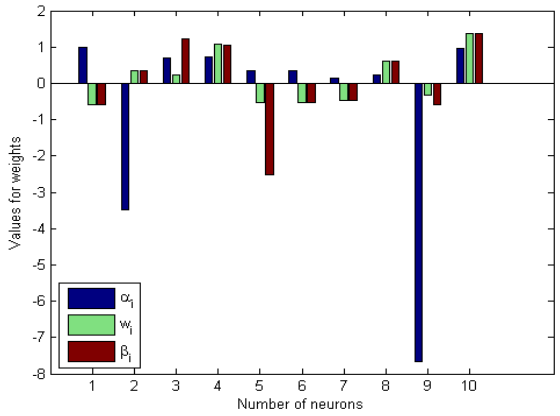



**Fig. 4.10** Weights obtained by GA-SA algorithm.

**Table 4.18** Comparison of Results for Problem III.

| T | PMA | ŷ(t) | | |
|---|---|---|---|---|
| | | *PS* | *GA* | *GA-PS* |
| 0.0 | 0.000000 | 0.016428 | 0.001534 | 0.001437 |
| 0.1 | 0.032734 | 0.101048 | 0.037205 | 0.036102 |
| 0.2 | 0.133300 | 0.276368 | 0.132477 | 0.132843 |
| 0.3 | 0.296654 | 0.327819 | 0.299232 | 0.292562 |
| 0.4 | 0.519024 | 0.442389 | 0.512509 | 0.520411 |
| 0.5 | 0.797107 | 0.682939 | 0.799598 | 0.801290 |
| 0.6 | 1.127822 | 0.123648 | 1.128962 | 1.126923 |
| 0.7 | 1.508199 | 0.160237 | 1.502085 | 1.504872 |
| 0.8 | 1.935317 | 2.009871 | 1.945076 | 1.928239 |
| 0.9 | 2.406276 | 2.536479 | 2.360927 | 2.382738 |
| 1.0 | 2.918175 | 2.978356 | 2.908120 | 2.910234 |

### 4.3.4 SOME FURTHER DISCUSSION

In this section, some necessary further discussion is added to elaborate the results. The advantages and limitation of the proposed method is also presented, so one should know about its effectiveness and reliability.

In the design scheme, the number of steps *s* is taken as 2, 5, and 10, respectively, for corresponding equations (4.3.3), (4.3.8) and (4.3.11) in the interval (0, 1) randomly. It means the mesh size used in our scheme is 0.5, 0.2 and 0.1 for problem I. II, and III respectively. On the other hand in PMA technique the results obtained with number of steps are 100, using mesh size $h = 0.01$ in the interval (0, 1). It is well-known that for the numerical method, decreasing the mesh size increases the accuracy of algorithm, but at the cost of extensive computational budget. It can be inferred from Table 4.11 and Table 4.18 that the results of our scheme are closed to PMA method with less number of steps. Another advantage of our algorithm is that once weights are obtained by the training of FDE-NN networks, the solution of the equation can be obtained for any continuous input time within the interval (0, 1), whereas, PMA approach results are available only on predefined discrete grid of inputs within the interval (0, 1). Therefore, if one desired the result on any new inputs timing, the whole cumbersome iterative procedure has to run.

It can be seen from the result given in Table 4.11 that on average the value of absolute error for GA-PS lies in the range $10^{-2}$ to $10^{-3}$, whereas reported results of VIM [99] have good approximation for the exact solution in closed vicinity of initial guess while the value of absolute error is going to raise with increase of time e.g. at $t = 1$, it is 0.225. However, the results by PMA method are better than that of GA-PS algorithm, but it has limitation as discussed earlier.

Moreover, it is bit tricky to decide the appropriate number for steps in optimization of weights by our scheme. The decision of the input span interval and step is made on compromise between the computation complexity and accuracy of the algorithm. The time taken for the computation is measured in order to solve Bagley-Torvik equation

with FDE-NN networks optimized with GA algorithm (200 individuals, 3000 generations) and PS technique (3000 generation). Optimization of weights with the help of fitness function provided in equation (4.3.11) the algorithm required executing MLF function 20000 times for single generation, whereas the single generation using fitness functions in equation (4.3.3) and (4.3.8) required 10000 and 4000 time execution of MLF function, respectively. The average total time taken by the algorithm for its single run and time taken exclusively by MLF function is shown in Table 4.19. It can be seen that about 80% of execution time is spent on the calculation of MLF function only. The time analysis provided in this article is carryout using Dell Latitude D630 laptop with Intel(R) Core(TM) 2 Duo CPU T9300, 2.50GHz and MATLAB version R2008b.

**Table 4.19** Computational Complexity of the Algorithms.

| Span (0,T) | Steps | FDE-NN | Counts for MLF (Million) | Total Time | Time (MLF) |
|---|---|---|---|---|---|
| (0,1) | 02 | GA | 12.0000 | 462s | 371s |
| | | PS | 0.06000 | 7.66s | 4.55s |
| (0,1) | 05 | GA | 30.0000 | 1149s | 920s |
| | | PS | 0.15000 | 15.4s | 9.34s |
| (0,1) | 10 | GA | 60.0000 | 2297s | 1852s |
| | | PS | 0.30000 | 31,7s | 18.6s |
| (0.1) | 20 | GA | 120.000 | 4593s | 3695s |
| | | PS | 0.60000 | 60.8s | 37.5s |

### 4.3.4 CONCLUSION

On the basis of the simulations and analysis in the last sections, it can be concluded that fractional order system of Bagley-Torvik equation can be solved by designed heuristic computational intelligence algorithm. The fractional differential equation neural networks of the equation trained by GA-PS algorithm is the best stochastic optimizer compared to PS, GA algorithms. On the basis of the statistical analysis, it can be inferred that our proposed computing approaches is reliable, effective and

easily applicable to such complex fractional order systems. In our future work, we intend to use other biologically inspired computational intelligence algorithms to solve these fractional order systems.

## 4.4 SUMMARY OF CHAPTER

This chapter presents the application of design methodology for solving fractional order system using feed-forward ANN optimized with evolutionary computational intelligence technique. The chapter is divided in three sections. In the first section generic form of fractional order system given by linear and non-linear differential equations of fractional order has been solved. In the second section design scheme is applied to well known Riccati fractional system. Two problems have been solved for such system for both integer and non-integer order cases. In the third section, solution of Bagley-Torvik fractional system is presented. Three problems have been solved. In the first problem, Bagley-Torvik equation with known exact solution is solved for validation of our design scheme to such systems. In the second problem, the given scheme is applied to solve Bagley-Torvik equation with inhomogeneous initial conditions for which Podlubny scheme fails to provide results. In the third problem, Bagley-Torvik equation with unknown exact solution is solved by the given approach.

# CHAPTER 5

# SWARM INTELLIGENCE OPTIMIZED NEURAL NETWORKS FOR SOLVING FRACTIONAL ORDER SYSTEMS

In this chapter, fractional order systems given by fractional differential equations involving single and multi term fractional derivatives, are solved using swarm intelligence optimized neural networks. A number of linear and nonlinear problems are solved with detailed analysis and investigation by the given scheme. The statistical analysis of the results is also given to prove the reliability and effectiveness of the proposed scheme. Beside the exact solution, the comparison of the results is made with various state of the art stochastic and deterministic numerical solvers. The time of computation was also analysed for the results by change of intervals and step size. Special kind of fractional order systems represented by fractional Riccati differential equations and Bagley-Torvik equations are also solved with the proposed scheme. Moreover, the comparative studies of results are made with FDE-NN networks optimized with evolutionary computation techniques.

## 5.1 FRACTIONAL ORDER SYSTEM REPRESENTED WITH FRACTIONAL DIFFERENTIAL EQUATIONS

In this section, the studies are presented to solve linear and nonlinear fractional order systems given by fractional differential equation by proposed scheme. Studies are carried out to analyze the applicability of the methodology to such equations and made comparison with available exact solutions, other numerical solvers. The material presented in this section is based on our accepted work [231].

### 5.1.1 PROBLEM I: LINEAR FRACTIONAL DIFFERENTIAL EQUATION WITH KNOWN EXACT SOLUTION

In this problem, a linear ordinary fractional differential equation with known exact solution is taken and analyzed the applicability of the proposed designed scheme. The following equation was taken which has also been solved by many authors of fractional calculus ([164], [216]).

$$D^{\nu} y(t) = t^2 + \frac{2}{\Gamma(3-\nu)} t^{2-\nu} - y(t), \quad 0 < \nu \le 1 \tag{5.1.1}$$

with condition $y(0) = 0$, and $y(1) = 1$ and the exact solution is given as

$$y(t) = t^2 \tag{5.1.2}$$

To solve this problem using FDE-NN methodology, the learning of weights was made with PSO-SA, a hybrid intelligent algorithm. Results were also determined by training of weights with PSO, GA, GA-SA and SA algorithms. A total of $10$ neurons were taken, which resulted in $30$ unknown parameters or weights ($\alpha_i, w_i$ and $b_i$). These weights were restricted to be real numbers between $-10$ and $10$. The values of the fractional order derivative $\nu$ were taken as $0.5$ and $0.75$. The input $t$ is taken between $0$ and $1$ with a step size of $0.10$. Therefore, the error function, $e_j$, is formulated as

$$e_j = \left. \begin{array}{l} \frac{1}{11} \sum_{i=1}^{11} [D^{\nu} \hat{y}(t_i) - t_i^2 - \frac{2}{\Gamma(3-\nu)} t_i^{2-\nu} + \hat{y}(t_i)]^2 \\ + [\hat{y}(0)]^2 + [y(1) - 1]^2 \end{array} \right|_j, \quad j = 1,2,3\ldots \tag{5.1.3}$$

where $j$ is the flight index, $\hat{y}(t)$ and $D\hat{y}(t)$ are networks given in (3.1.8) and (3.1.10), respectively. The parameter settings used for evaluation of results for PSO and SA techniques are provided in Table 5.1 while for GA the setting is given in Table 5.2.

Our scheme runs iteratively in order to find the minimum of error function, $e_j$, with stoppage criteria as $2000$ number of flights or value of $e_j < 10^{-08}$ whichever comes earlier. One of the set of unknown weights learned by the PSO-SA algorithm is provided in Table 5.3. These weights can be used in equation (3.1.8) to obtain the solution of the equation for any input time $t$ between $0$ and 1.

**Table 5.1** Parameters Setting for SA and PSO Algorithms.

| **PSO** | | **SA** | |
|---|---|---|---|
| *Parameters* | *Value / Setting* | *Parameters* | *Value / Setting* |
| Swarm Size | 80 | Start Point size | 30 |
| Particle size | 30 | Annealing function | Fast |
| Flights | 2000 | Iteration | 60000 |
| Local acceleration factor | Linear decreasing (5 to 0.5) | Max. function evaluations | 90000 |
| Global acceleration factor | Linear increasing (0.5 to 5) | Reannealing interval | 100 |
| Inertia weight | Linearly decreasing (0.9 to 0.4) | Temperature update | Exponential |
| $v_{max}$ | 02 | Initial temperature | 100 |

**Table 5.2** Parameters Setting for GA Algorithm.

| **No.** | **Parameters** | **Value / Setting** |
|---|---|---|
| 1 | Population Size | 200 |
| 2 | Chromosome size | 30 |
| 3 | Generations | 1000 |
| 4 | Selection | Stochastic Uniform |
| 5 | Crossover fraction & method | 0.8 & Scattered |
| 6 | Mutation | Adaptive feasible |
| 7 | Elite count | 4 |

The classical deterministic numerical technique in GL method [164] is also used to obtain the solution of the equation (5.1.1). The relation used to solve the equation can be written as

$$y_m = \frac{1}{1+h^{\nu}}((t_{m-1}^2 + \frac{2}{\Gamma(3-\nu)} t_{m-1}^{2-\nu}) + \sum_{j=1}^{m} W_j^{(\nu)} y_{m-j}), \quad m = 1,2,3\ldots100$$
$$y_m = y(mh), \quad h = 0.01 \quad t_m = mh, \quad W_j^{(\nu)} = (-1)^j \binom{\nu}{j}, \quad (j = 0,1,2,\ldots\ldots) \tag{5.1.4}$$

**Table 5.3** The Set of Weights of FDE-NN Networks trained by PSO-SA algorithm.

| I | **ν = 0.5** | | | **ν = 0.75** | | |
|---|---|---|---|---|---|---|
| | $w_i$ | $\alpha_i$ | $b_i$ | $w_i$ | $\alpha_i$ | $b_i$ |
| 1 | -0.868488 | 2.267107 | 0.158642 | 0.818694 | 1.037723 | -0.043346 |
| 2 | 0.000202 | 0.508606 | -1.322480 | 0.826450 | 0.690690 | -0.162463 |
| 3 | -0.114243 | -0.541645 | 1.406505 | 0.846779 | -0.513357 | -0.973743 |
| 4 | 0.351035 | -0.664693 | -0.157457 | 0.247052 | -0.093627 | -0.677092 |
| 5 | 0.194204 | -0.656206 | -3.438001 | 1.023052 | -0.416330 | -4.214132 |
| 6 | -0.365107 | 1.172230 | -0.431901 | 0.235052 | -1.818209 | -0.428702 |
| 7 | -0.573958 | -0.516307 | 0.319690 | 0.019579 | 0.597240 | 0.746637 |
| 8 | 0.727216 | 0.951739 | 0.458089 | -1.490539 | 0.292185 | 0.784214 |
| 9 | 0.792515 | 0.138632 | -0.129783 | 0.726425 | -0.321066 | -1.424499 |
| 10 | -0.557634 | -2.763771 | -0.504124 | -0.070790 | -1.453336 | 0.305023 |

The solution obtained for the equation by FDE-NN with PSO-SA algorithm and the GL method are given in Table 5.4 and Table 5.5 for $\nu = 0.5$ and $\nu = 0.75$, respectively. These tables also include the exact solution as well as the reported result of stochastic solver using neural network supported by genetic algorithms [217]. It can be seen from the tables that the proposed methodology gives better result than that of GL method. It also gives better results as compared to that of GA-SA algorithm, although the population size for GA-SA case was $200$ while that for PSO- SA was only $80$. Hence with less than half the computational cost, the results of PSO-SA technique are still comparable to that of GA-SA algorithm.

In order to make further analysis of the scheme results are also determined for various inputs $t \in (0, 10)$. The results are shown graphically in Fig. 5.1. It can be seen from Fig. 5.1(a), that the result obtained by our algorithm is overlapping the exact solution. In order to view the difference clearly the zoomed diagram is also shown in Fig.

5.1(b). The training of FDE-NN is made for bounded inputs between $0$ and $1$. Therefore, the error starts to accumulate for input greater then $1$, as can be seen from Fig. 5.1(c). It may start to grow rapidly for more deviated training inputs. Moreover, it can be seen from Fig. 5.1(d) that solution starts to diverge for inputs greater than $4$.

**Table 5.4** Results for Solution of FDE Equation Given in Problem I for $\nu = 0.5$.

| t | Exact | Solvers | | | Absolute error | | |
|---|---|---|---|---|---|---|---|
| | | *PSO-SA* | *GA[217]* | *GL* | *PSO-SA* | *GA[217]* | *GL* |
| 0.0 | 0.00 | 2.8e-05 | 0.0004 | 0.0000 | 2.8e-05 | 3.7e-04 | 0.0000 |
| 0.1 | 0.01 | 0.0099 | -- | 0.0104 | 1.1e-04 | -- | 4.0e-04 |
| 0.2 | 0.04 | 0.0404 | 0.0396 | 0.0407 | 3.6e-04 | 3.6e-04 | 7.4e-04 |
| 0.3 | 0.09 | 0.0910 | -- | 0.0910 | 9.6e-04 | -- | 1.1e-03 |
| 0.4 | 0.16 | 0.1614 | 0.1596 | 0.1613 | 1.4e-03 | 4.3e-04 | 1.3e-03 |
| 0.5 | 0.25 | 0.2514 | -- | 0.2516 | 1.4e-03 | -- | 1.6e-03 |
| 0.6 | 0.36 | 0.3611 | 0.3573 | 0.3619 | 1.1e-03 | 2.7e-03 | 1.9e-03 |
| 0.7 | 0.49 | 0.4905 | -- | 0.4921 | 4.9e-04 | -- | 2.1e-03 |
| 0.8 | 0.64 | 0.6399 | 0.6352 | 0.6423 | 6.5e-05 | 4.8e-03 | 2.3e-03 |
| 0.9 | 0.81 | 0.8098 | -- | 0.8126 | 1.7e-04 | -- | 2.6e-03 |
| 1.0 | 1.00 | 1.0007 | 1.0004 | 1.0028 | 7.5e-04 | 3.5e-04 | 2.8e-03 |

**Table 5.5** Results for Solution of FDE Equation Given in Problem I for $\nu = 0.75$.

| t | Exact | PSO-SA | GL | Absolute error | |
|---|---|---|---|---|---|
| | | | | *PSO-SA* | *GL* |
| 0.0 | 0.00 | 0.0001 | 0.0000 | 6.3e-05 | 0.0000 |
| 0.1 | 0.01 | 0.0103 | 0.0107 | 3.1e-04 | 6.7e-04 |
| 0.2 | 0.04 | 0.0414 | 0.0413 | 1.4e-03 | 1.2e-03 |
| 0.3 | 0.09 | 0.0928 | 0.0918 | 2.8e-03 | 1.8e-03 |
| 0.4 | 0.16 | 0.1636 | 0.1622 | 3.6e-03 | 2.2e-03 |
| 0.5 | 0.25 | 0.2538 | 0.2527 | 3.8e-03 | 2.7e-03 |
| 0.6 | 0.36 | 0.3631 | 0.3631 | 3.1e-03 | 3.1e-03 |
| 0.7 | 0.49 | 0.4918 | 0.4934 | 1.8e-03 | 3.4e-03 |
| 0.8 | 0.64 | 0.6402 | 0.6438 | 2.3e-04 | 3.8e-03 |
| 0.9 | 0.81 | 0.8091 | 0.8141 | 9.3e-04 | 4.1e-03 |
| 1.0 | 1.00 | 0.9991 | 1.0044 | 9.1e-04 | 4.4e-03 |

Enough simulations have been preformed to test the reliability of our designed scheme. In this regard, 100 independent runs were carried out for finding the weights of FDE-NN optimized with SA, GA, GA-SA, PSO, and PSO-SA algorithms. The MATLAB optimization toolbox is used for SA and GA with parameter setting as given in Table 5.1 and Table 5.2 respectively. The term best and worst corresponds to minimum and maximum errors. The statistical parameter like mean and standard deviation (STD) are useful to determine the behavior of the results.

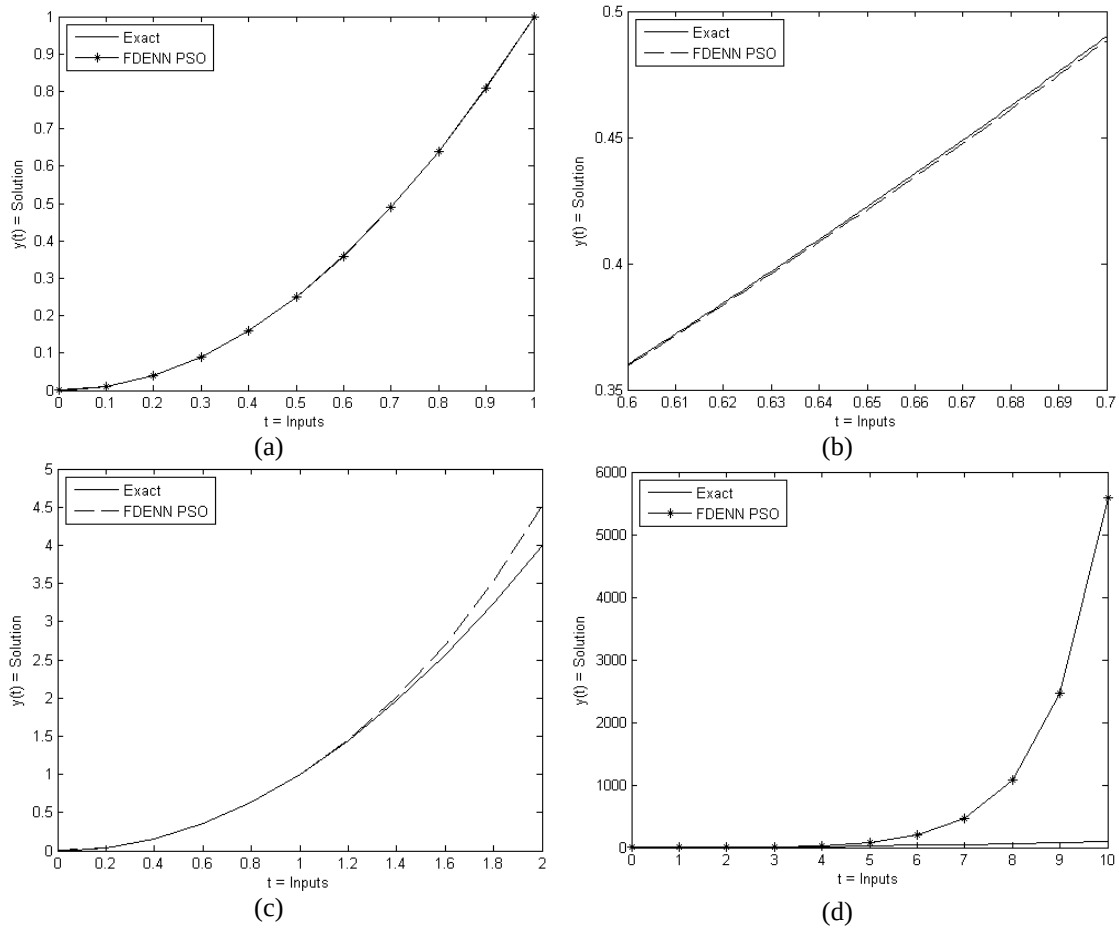



**Fig. 5.1** Comparison of proposed scheme with exact solution for $\nu = 0.75$, (a) is for inputs between (0, 1), (b) is for inputs between (0.6, 0.7), (c) is for inputs between (0, 2) (d) is for inputs (0, 10).

The best, worst, mean and STD for inputs $t \in (0, 1)$ with step $0.1$ is calculated for FDE-NN networks optimized by PSO-SA algorithm. The results are summarized in Table 5.6. These results validate the applicability of our scheme for such equations. Moreover, the other stochastic solvers are used for this optimization problem for

comparison. In this regard 100 independent runs for finding weights by SA, GA, GA-SA, and standard PSO are also executed. The comparison of the results is made for some inputs and it is given in Table 5.7 and Table 5.8 for fractional order $\nu = 0.5$ and $\nu = 0.75$, respectively. It can be seen that the best result are obtained using PSO-SA hybrid technique in both the cases.

**Table 5.6** The Statistical Parameters of the Solution by FDE-NN Networks Optimized with PSO-SA Scheme.

| T | ν = 0.5 | | | | ν = 0.75 | | | |
|---|---|---|---|---|---|---|---|---|
| | *Best* | *Worst* | *Mean* | *STD* | *Best* | *Worst* | *Mean* | *STD* |
| 0.0 | 3.03e-06 | 0.03467 | 4.45e-03 | 5.36e-03 | 2.55e-05 | 0.05611 | 6.03e-03 | 8.59e-03 |
| 0.1 | 1.31e-05 | 0.01877 | 5.22e-03 | 3.94e-03 | 1.31e-04 | 0.01429 | 5.33e-03 | 3.41e-03 |
| 0.2 | 1.69e-04 | 0.03917 | 8.42e-03 | 7.46e-03 | 9.79e-05 | 0.04466 | 9.82e-03 | 8.60e-03 |
| 0.3 | 2.34e-06 | 0.04928 | 8.94e-03 | 9.27e-03 | 1.88e-04 | 0.07427 | 1.22e-02 | 1.16e-02 |
| 0.4 | 8.41e-04 | 0.05069 | 1.00e-02 | 8.93e-03 | 8.71e-04 | 0.09121 | 1.41e-02 | 1.33e-02 |
| 0.5 | 1.91e-05 | 0.04241 | 9.37e-03 | 1.04e-02 | 2.77e-04 | 0.09974 | 1.39e-02 | 1.62e-02 |
| 0.6 | 1.04e-04 | 0.04956 | 9.44e-03 | 1.15e-02 | 1.32e-05 | 0.10673 | 1.30e-02 | 1.81e-02 |
| 0.7 | 2.46e-04 | 0.04828 | 1.09e-03 | 1.06e-02 | 3.69e-05 | 0.09606 | 1.41e-02 | 1.58e-02 |
| 0.8 | 3.55e-05 | 0.03809 | 1.06e-02 | 8.22e-03 | 2.67e-05 | 0.06624 | 1.27e-02 | 1.10e-02 |
| 0.9 | 4.89e-05 | 0.02684 | 5.59e-03 | 4.52e-03 | 1.20e-05 | 0.03013 | 6.10e-03 | 4.83e-03 |
| 1.0 | 3.77e-06 | 0.02643 | 6.55e-03 | 5.52e-03 | 3.15e-05 | 0.05315 | 8.49e-03 | 8.57e-03 |

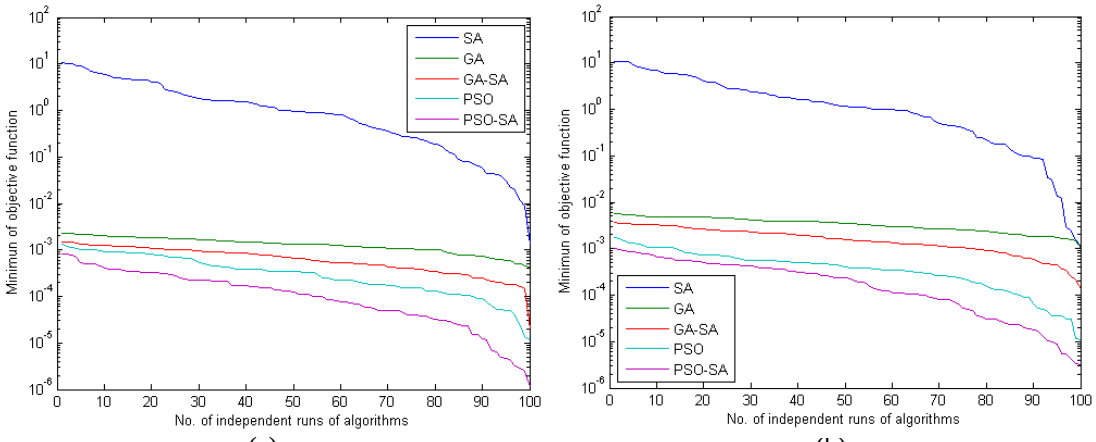
**Table 5.7** Comparison of Stochastic Solvers for Solution of Problem I for ν = 0.5.

| T | FDE-NN | ν = 0.5 | | | |
|---|---|---|---|---|---|
| | | *Best* | *Worst* | *Mean* | *STD* |
| 0.2 | SA | 2.47e-03 | 33.1611 | 1.25208 | 3.18773 |
| | GA | 4.52e-03 | 0.05746 | 1.89e-02 | 9.37e-03 |
| | GA-SA | 3.09e-04 | 0.04393 | 1.50e-02 | 8.57e-03 |
| | PSO | 7.25e-04 | 0.06474 | 1.12e-02 | 9.48e-03 |
| | PSO-SA | 1.69e-04 | 0.03917 | 8.42e-03 | 7.46e-03 |
| 0.8 | SA | 3.38e-03 | 59.8564 | 1.20177 | 5.42192 |
| | GA | 4.86e-04 | 0.07343 | 1.61e-02 | 1.26e-02 |
| | GA-SA | 1.67e-04 | 0.06357 | 1.38e-02 | 1.12e-02 |
| | PSO | 6.17e-05 | 0.06138 | 1.44e-02 | 1.16e-02 |
| | PSO-SA | 3.55e-05 | 0.03809 | 1.06e-02 | 8.22e-03 |

**Table 5.8** Comparison of stochastic solvers for solution of Problem I for ν = 0.75.

| T | FDE-NN | ν = 0.75 | | | |
|---|---|---|---|---|---|
| | | *Best* | *Worst* | *Mean* | *STD* |
| 0.2 | SA | 3.58e-03 | 10.4269 | 1.04062 | 1.38306 |
| | GA | 4.44e-04 | 0.08311 | 2.83e-02 | 1.09e-02 |
| | GA-SA | 4.38e-04 | 0.05050 | 2.45e-02 | 9.55e-03 |
| | PSO | 3.60e-04 | 0.06313 | 1.29e-02 | 1.08e-02 |
| | PSO-SA | 9.79e-05 | 0.04466 | 9.82e-03 | 8.60e-03 |
| 0.8 | SA | 6.48e-04 | 7.97759 | 0.68042 | 1.13319 |
| | GA | 2.28e-04 | 0.07917 | 3.14e-02 | 1.66e-02 |
| | GA-SA | 1.71e-05 | 0.06028 | 2.19e-02 | 1.39e-02 |
| | PSO | 4.95e-05 | 0.07404 | 1.41e-02 | 1.28e-02 |
| | PSO-SA | 2.67e-05 | 0.06624 | 1.27e-02 | 1.10e-02 |

Results are also provided in Fig. 5.2, in which the value of error function, $e_j$, is plotted in descending order against the number of independent runs of different solvers. It can be seen that the best results are given by PSO-SA algorithm. It is further added that our approach used optimization mainly based on particle swarm optimization technique, which is already proven to have reduced computational complexity than that of Genetic Algorithm based approaches ([218], [219]).




(a) (b)

**Fig. 5.2** Comparison of FDE-NN networks optimized with stochastic solvers, (a) is for the value of fractional order ν = 0.5 and (b) is for the value of fractional order ν = 0.75.

### 5.1.2 PROBLEM II: LINEAR FRACTIONAL DIFFERENTIAL EQUATION WITH UNKNOWN EXACT SOLUTION BUT AVAILABLE NUMERICAL SOLUTION

In this example our intent is to further analyze the proposed methodology by applying to the differential equation of fractional order for which no exact solution exists. Therefore, the comparative studies of the scheme are carried out against approximate analytic solver and other standard numerical methods. Let us take such ordinary fractional differential equation as [217]

$$Dy(t) - \tfrac{11}{15} D^{1/2} y(t) + \tfrac{2}{15} y(t) = 0, \quad y(0) = 0 \tag{5.1.5}$$

The approximate analytic solution can be determined by direct approach method [1]. It is written in the form of special function as

$$y(t) = a_1 \varepsilon_t(0, a_1^2) - a_2 \varepsilon_t(0, a_2^2) + a_1 \varepsilon_t(-\tfrac{1}{2}, a_1^2) - a_2 \varepsilon_t(-\tfrac{1}{2}, a_2^2). \tag{5.1.6}$$

where $a_1$ and $a_2$ are the zeros of indicial polynomial of (5.1.5) and $\varepsilon_t$ is a special function to two inputs and it can be represented in term of Mittag-Leffler function as

$$\varepsilon_t(\nu, a) = t^{\nu} \sum_{k=0}^{\infty} \frac{a t^k}{\Gamma(\nu + k + 1)} = t^{\nu} E_{1,\nu+1}(at) \tag{5.1.7}$$

where $E_{1,\nu+1}(at)$ is given in equation (2.2.5). Similarly, (5.1.6) can be written in terms of error function as

$$y(t) = a_1 e^{a_1^2 t} Erfc(-a_1 t^{1/2}) - a_2 e^{a_2^2 t} Erfc(-a_2 t^{1/2}). \tag{5.1.8}$$

The equation (5.1.5) is an important relation in fractional calculus. It can be interpreted as a simplified form of composite fractional relaxation equation, and also it is representing a special case of the unsteady motion of a particle accelerating in a viscous fluid under the action of gravity, which is referred to as Basset problem ([220], [221]).

This problem has been simulated by FDE-NN networks given in equations (3.1.8), (3.19) and (3.1.10) similar to previous problem. The input of the training set is chosen from time $t \in (0,1)$ with a step of $0.1$. The error function, $e_j$, can be formulated as

$$e_j = \frac{1}{11}\sum_{i=1}^{11}\left(D\hat{y}(t_i) - \frac{11}{15}D^{1/2}\hat{y}(t_i) + \frac{2}{15}\hat{y}(t_i)\right)^2 + \left(\hat{y}(0)\right)^2\Bigg|_j, \quad j = 1,2\ldots\ldots \tag{5.1.9}$$

The set of weights learnt stochastically using GA-SA, PSO, PSO-SA algorithms are given in Table 5.9. Using these weights in equation (3.1.8) the solution of the equation can be determined.

**Table 5.9** A Set of Weights Trained for FDE-NN Networks for Problem II.

| i | GA-SA | | | PSO | | | PSO-SA | | |
|---|---|---|---|---|---|---|---|---|---|
| | $w_i$ | $\alpha_i$ | $b_i$ | $w_i$ | $\alpha_i$ | $b_i$ | $w_i$ | $\alpha_i$ | $b_i$ |
| 1 | 0.4094 | 0.9826 | -0.2061 | -2.3181 | -0.1225 | 0.9630 | -0.0892 | 0.3091 | 0.1164 |
| 2 | -0.6304 | -0.7774 | -1.1842 | 1.0130 | 0.1499 | -0.6659 | 0.4324 | 0.1317 | -1.7017 |
| 3 | 0.2215 | 0.5764 | -0.5567 | -0.2743 | 0.1722 | -0.3252 | -0.0593 | 0.2479 | -1.5050 |
| 4 | -0.7881 | 0.2389 | -0.2642 | -0.9922 | 0.8438 | 0.0738 | -0.3343 | 0.3535 | -0.5573 |
| 5 | 0.5069 | 0.4321 | -0.6154 | 0.0771 | -2.5002 | -0.1074 | -6.0474 | -0.7880 | -1.2859 |
| 6 | 0.5308 | -0.7908 | 0.6351 | -0.4106 | 0.2219 | 0.3278 | 0.2264 | -0.0343 | 0.4295 |
| 7 | 0.3133 | 1.6549 | -0.5389 | -0.0258 | 0.9376 | 0.9195 | -2.0075 | 0.0881 | 0.9903 |
| 8 | 0.2510 | -1.3473 | -1.0939 | -1.0528 | 0.5050 | -0.5135 | -0.6530 | -0.2343 | 0.0673 |
| 9 | -0.1792 | 0.8190 | -0.3509 | -0.0671 | 0.8402 | -2.1567 | -0.0425 | -1.4225 | -0.8739 |
| 10 | -0.6003 | -1.6898 | -0.6316 | -0.8067 | -1.4298 | 0.1100 | 0.3306 | 0.1853 | 0.2807 |

The famous numerical technique developed by Podlubny [30] based on successive approximation method is also applied to solve the equation. The recursive relations are used for computations, given as

$$y_m = (1 + \frac{11}{15}h^{1/2} - \frac{2}{15}h)y_{m-1} + \frac{11}{15}h^{1/2}\sum_{j=1}^{m} W_j^{(1/2)} y_{m-j}, (m = 1,2,\ldots\ldots 100), h = 0.01, \tag{5.1.10}$$

where $y_m$ and $W_j^{(1/2)}$ are given in equation (5.1.4). In order to compare the results on time span $t \in (0,1)$ with step 0.1, Podlubny numerical method (PNM) using equation (5.1.10), approximate analytical solution using direct approach method (DAM) as

given in (5.1.8) and solution due to stochastic solvers GA-SA, PSO and PSO-SA have also been given in Table 5.10. It can be seen that PSO-SA algorithm provide best results.

**Table 5.10** Results for Solution of the Equation in Problem II.

| T | DAM | PNM | FDE-NN | | |
|---|---|---|---|---|---|
| | | | GA-SA | PS0 | PSO-SA |
| 0.0 | 0.066666 | 0.066700 | 0.008640 | 0.005948 | 0.003596 |
| 0.1 | 0.087156 | 0.080612 | 0.042230 | 0.048428 | 0.068514 |
| 0.2 | 0.097760 | 0.089371 | 0.071560 | 0.080050 | 0.103152 |
| 0.3 | 0.106931 | 0.096917 | 0.096658 | 0.103402 | 0.119309 |
| 0.4 | 0.115451 | 0.103878 | 0.117534 | 0.120541 | 0.126997 |
| 0.5 | 0.123620 | 0.110499 | 0.134181 | 0.133108 | 0.131393 |
| 0.6 | 0.131591 | 0.116910 | 0.146577 | 0.142423 | 0.135090 |
| 0.7 | 0.139453 | 0.123186 | 0.154685 | 0.149556 | 0.139316 |
| 0.8 | 0.147266 | 0.129376 | 0.158449 | 0.155386 | 0.144588 |
| 0.9 | 0.155071 | 0.135515 | 0.157798 | 0.160649 | 0.151071 |
| 1.0 | 0.162898 | 0.141627 | 0.152642 | 0.165974 | 0.158757 |

Once again 100 independent runs have been executed for this equation using solvers SA, GA, GA-SA, PSO and PSO-SA. The minimum value of function, $e_j$, is set as figure of merit for this equation. The summary of the statistical results are provided in Table 5.11 and plotted in Fig 5.3.

**Table 5.11** Statistical Analysis of Stochastic Solvers.

| FDE-NN | Value of error function '$e_j$' | | | |
|---|---|---|---|---|
| | Best | Worst | Mean | STD |
| SA | 1.47e-02 | 112.161 | 4.12208 | 7.1233 |
| GA | 1.37e-05 | 0.05746 | 2.81e-03 | 4.57e-03 |
| GA-SA | 9.01e-06 | 0.04393 | 1.75e-03 | 3.72e-03 |
| PSO | 1.25e-05 | 0.06474 | 1.82e-03 | 9.48e-03 |
| PSO-SA | 1.79e-06 | 0.03917 | 1.41e-03 | 2.45e-03 |

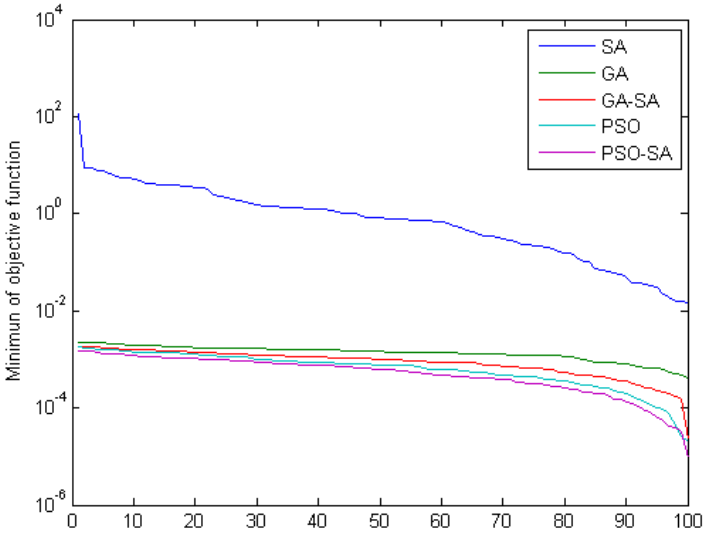



**Fig. 5.3** Comparison of results for FDE-NN networks optimized with stochastic solvers.

### 5.1.3 PROBLEM III: COMPLEX NONLINEAR FRACTIONAL DIFFERENTIAL WITH KNOWN EXACT SOLUTION

In this example a complex fractional differential equation is taken to determine the strength and weakness of our proposed scheme for such models. In this regard let us take a non-linear ordinary fractional differential equation given as ([164], [222])

$$D^{\nu} y(t) = \frac{40320}{\Gamma(9-\nu)} t^{8-\nu} - 3\frac{\Gamma(5+\nu/2)}{\Gamma(5-\nu/2)} t^{4-\nu/2} + \frac{9}{4}\Gamma(\nu+1) + (\frac{3}{2}t^{\nu/2} - t^{4})^{3} - (y(t))^{3/2} \quad (5.1.11)$$

with initial and boundary conditions given as $y(0) = 0$ and $y(1) = 0.25$, respectively. The exact solution for this equation is given as

$$y(t) = t^{8} - 3t^{4+\nu/2} + \tfrac{9}{4}t^{\nu} \quad (5.1.12)$$

The classical numerical techniques used for solution of FDEs in problem I and II are unable to provide solution to such problems. However, modern deterministic methods with higher computational cost can provide solution for (5.1.11), such as fractional Adams method [223] and Variation Iteration method [224].

The simulation has been performed for this problem on similar pattern to the previous examples. The order of fractional derivative $\nu$ is taken as $0.25, 0.5$ and $0.75$. The set

of weights learned stochastically using PSO-SA algorithms are given in Table 5.12. Using these weights in equation (3.1.8), the results can be obtained which are given in Table 5.13. Moreover, the graphic comparison of our obtained solution with the exact solution is given in Fig. 5.4. The numerical result of the Adams scheme [164] based on predictor corrector approach are given separately in Table 5.14 as the errors are given only for t = 1 for different mesh sizes.

**Table 5.12** Weights for FDE-NN Networks by PSO-SA Algorithm for Problem III.

| I | ν = 0.25 | | | ν = 0.5 | | | ν = 0.75 | | |
|---|---|---|---|---|---|---|---|---|---|
| | $w_i$ | $\alpha_i$ | $b_i$ | $w_i$ | $\alpha_i$ | $b_i$ | $w_i$ | $\alpha_i$ | $b_i$ |
| 1 | 0.1054 | 0.6575 | 0.0868 | -1.6488 | -0.0591 | 1.7376 | 0.5114 | 0.9316 | 0.1784 |
| 2 | 0.2637 | 3.1429 | -0.9555 | -0.1061 | 0.5117 | -1.1450 | -0.0521 | 0.9119 | -0.2006 |
| 3 | 0.0883 | 0.4428 | 0.7153 | -1.2606 | 0.4263 | -0.4043 | -4.6482 | 0.5299 | -2.0818 |
| 4 | -5.3349 | -1.9707 | 0.2794 | -0.7293 | -0.0394 | -7.9611 | -2.0321 | -0.657 | 1.1406 |
| 5 | 1.4004 | -0.6343 | 0.2388 | -0.5461 | 0.4472 | 1.6015 | -0.4419 | 0.0081 | -0.2150 |
| 6 | 0.4616 | -0.7894 | -0.3201 | 2.9929 | -0.4345 | -1.6370 | -1.7249 | -0.063 | 0.4909 |
| 7 | -0.1989 | 7.2787 | -6.1817 | 0.5516 | 1.1411 | -1.3916 | 1.2395 | 0.3243 | -0.5894 |
| 8 | 0.2006 | 1.2917 | 0.5472 | -2.5526 | -2.0805 | -0.5869 | -0.3688 | 1.2300 | -1.6236 |
| 9 | -0.8850 | -0.8777 | -1.5429 | -3.6911 | -0.8474 | 0.6270 | 2.0127 | 0.0272 | 1.3750 |
| 10 | 0.5557 | -0.7218 | 0.2288 | -0.5337 | 0.1364 | 0.4523 | 2.4863 | -2.503 | -2.1158 |

**Table 5.13** Summary of the Results for Solution of FDE in Problem III.

| T | Exact | ν = 0.5 | | | | ν = 0.25 | | ν = 0.75 | |
|---|---|---|---|---|---|---|---|---|---|
| | | FDE-NN | | Absolute Error | | Exact | PSO-SA | Exact | PSO-SA |
| | | PSO-SA | GA[23] | PSO-SA | GA[23] | | | | |
| 0.0 | 0.0000 | 3.8e-05 | 3.8e-05 | -- | -- | 0.0000 | 4.1e-04 | 0.0000 | 1.5e-06 |
| 0.1 | 0.7113 | 0.6235 | 0.0879 | 0.7066 | 0.0047 | 1.2650 | 0.9272 | 0.3999 | 0.3781 |
| 0.2 | 1.0030 | 1.0308 | 0.0278 | 0.9994 | 0.0036 | 1.5007 | 1.4600 | 0.6702 | 0.6811 |
| 0.3 | 1.2145 | 1.2787 | 0.0642 | 1.2198 | 0.0054 | 1.6443 | 1.7297 | 0.8966 | 0.9137 |
| 0.4 | 1.3626 | 1.4051 | 0.0425 | 1.3656 | 0.0030 | 1.7215 | 1.8183 | 1.0778 | 1.0779 |
| 0.5 | 1.4372 | 1.4339 | 0.0034 | 1.4317 | 0.0055 | 1.7240 | 1.7759 | 1.1971 | 1.1725 |
| 0.6 | 1.4175 | 1.3780 | 0.0395 | 1.4095 | 0.0080 | 1.6323 | 1.6323 | 1.2296 | 1.1926 |
| 0.7 | 1.2813 | 1.2410 | 0.0041 | 1.2857 | 0.0044 | 1.4268 | 1.4036 | 1.1494 | 1.1286 |
| 0.8 | 1.0181 | 1.0181 | 8.6e-05 | 1.0420 | 0.0239 | 1.1007 | 1.0970 | 0.9408 | 0.9658 |
| 0.9 | 0.6479 | 0.6955 | 0.0477 | -- | -- | 0.6794 | 0.7136 | 0.6174 | 0.6828 |
| 1.0 | 0.2500 | 0.2499 | 4.7e-05 | -- | -- | 0.2500 | 0.2501 | 0.2500 | 0.2499 |

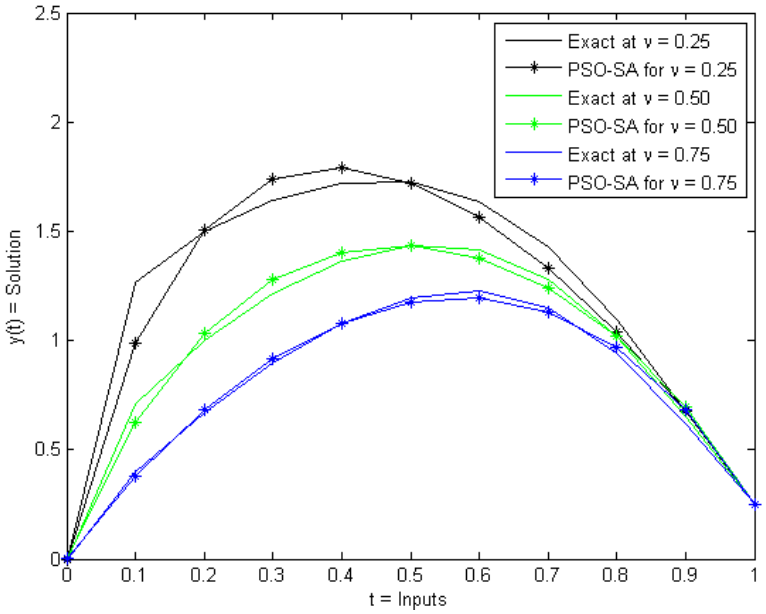



**Fig. 5.4** The exact solution and solution due to PSO-SA algorithm For problem III.

**Table 5.14** Numerical Results of Adam Scheme [164].

| No | Mesh size (h) | Error at *t = 1* |
|---|---|---|
| 1 | 1/10 | 0.25000 |
| 2 | 1/20 | 1.81e-02 |
| 3 | 1/40 | 3.61e-03 |
| 4 | 1/80 | 1.45e-03 |
| 5 | 1/160 | 6.58e-04 |
| 6 | 1/320 | 2.97e-04 |
| 7 | 1/640 | 1.31e-04 |

As can be seen from Table 5.13 and 5.14 that at time $t=1, \nu=0.25$ and mesh size $h=1/10$, the value of the error by Adam scheme is 0.25 while by our proposed scheme is $5.60\times10^{-04}$. This error reduces to $1.31\times10^{-04}$ in Adam scheme by reducing mesh size to $1/640$ but at cost of much greater computational complexity.

The statistical analysis, based on 100 independent runs of our scheme is provided in Table 5.15. It can be seen that differences exist between best and worst results. Moreover, the average accuracy obtained is in the range of $10^{-01}$ to $10^{-02}$. The stochastic solver like SA algorithm mostly provides the results with objective function value more than 1. Similarly the effects of SA in hybridization approach with PSO

and GA are also not significant. In most of the independent runs there is no sufficient improvement is seen with SA technique. Numbers of 125 independent runs of GA-SA and PSO-SA hybrid approaches are executed and the results of best 100 runs are plotted in Fig. 5.5, in which the value of the error function, $e_j$, has been drawn in descending order against the numbers of independent runs of the algorithms.

**Table 5.15** Statistical Parameters for Solution by FDE-NN Networks Optimized with PSO-SA Algorithm.

| T | ν = 0.5 | | | | ν = 0.75 | | | |
|---|---|---|---|---|---|---|---|---|
| | *Best* | *Worst* | *Mean* | *STD* | *Best* | *Worst* | *Mean* | *STD* |
| 0.0 | 2.06e-05 | 0.0392 | 0.0106 | 0.0184 | 1.73e-05 | 0.0961 | 0.0342 | 0.0283 |
| 0.1 | 2.34e-03 | 0.3549 | 0.1366 | 0.0866 | 8.30e-04 | 0.2191 | 0.0568 | 0.0487 |
| 0.2 | 2.10e-03 | 0.3164 | 0.0933 | 0.0762 | 1.31e-03 | 0.3038 | 0.0898 | 0.0722 |
| 0.3 | 2.58e-04 | 0.2946 | 0.0784 | 0.0599 | 4.35e-04 | 0.2476 | 0.0820 | 0.0646 |
| 0.4 | 1.55e-04 | 0.1627 | 0.0477 | 0.0334 | 4.29e-04 | 0.1288 | 0.0477 | 0.0322 |
| 0.5 | 8.82e-08 | 0.1835 | 0.0094 | 0.0259 | 4.19e-07 | 0.1837 | 0.0207 | 0.0349 |
| 0.6 | 4.50e-04 | 0.2091 | 0.0550 | 0.0398 | 3.03e-03 | 0.2762 | 0.0679 | 0.0541 |
| 0.7 | 3.82e-04 | 0.2690 | 0.0816 | 0.0630 | 1.48e-03 | 0.3071 | 0.0837 | 0.0720 |
| 0.8 | 4.10e-04 | 0.3646 | 0.0924 | 0.0782 | 3.52e-04 | 0.2698 | 0.0757 | 0.0643 |
| 0.9 | 1.80e-03 | 0.3331 | 0.0887 | 0.0767 | 1.77e-05 | 0.2619 | 0.0621 | 0.0419 |
| 1.0 | 1.08e-09 | 0.6208 | 0.0120 | 0.0641 | 1.06e-06 | 0.1944 | 0.0105 | 0.0296 |

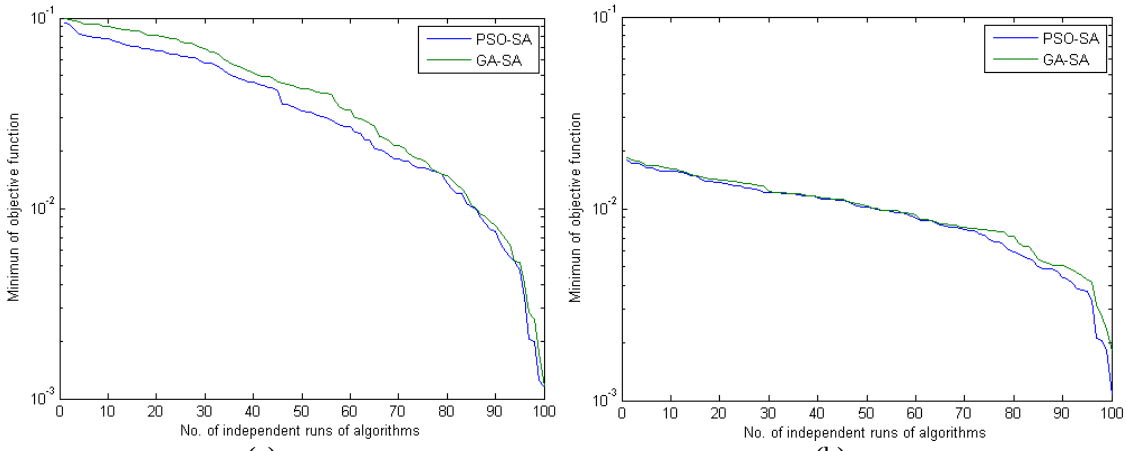



**Fig. 5.5** Comparison of FDE-NN networks optimized with stochastic solvers, in (a) and (b) the value of fractional order is taken as ν = 0.5 and ν = 0.75, respectively.

### 5.1.4 CONCLUSION

A new stochastic computational approach has been developed for solving the FDEs by swarm intelligence optimized neural networks. The method has been tested successfully by applying it to different linear and non-linear ordinary differential equations of fractional order. A large number of Monte Carlo simulations with stochastic solvers validated its reliability and effectiveness. The best results are achieved for FDE-NN networks optimized with PSO-SA algorithm instead of SA, GA, GA-SA, PSO algorithms. We have shown that PSO-SA for less than half the population as compared to GA-SA gives same accuracy. It has also been shown that proposed scheme can approximate the solution with the accuracy comparable to the standard state of art deterministic numerical solvers or even better in some cases. The strength of designed scheme over such solvers is that, it can provide the result on continuous finite time domain, instead of predefined discrete grid of points.

## 5.2 RICCATI DIFFERENTIAL EQUATION OF FRACTIONAL ORDER

In this section, the results of simulation are presented for two different problems associated with the nonlinear Riccati differential equation of arbitrary order. Apply the scheme to integer order case of the equation and then extend the proposed method for application to fractional order cases. The statistical analysis of results is carried out to test the verification, validation of the method. The effect of change of input span and step size to the accuracy and computational complexity has been analyzed. The comparison of the results of all problems is made with available exact solutions, stochastic solvers, and standard numerical techniques. The necessary discussion is also provided on the results obtained. Moreover, the material provide in this section is based on the published work [232].

### 5.2.1 PROBLEM I: NONLINEAR RICCATI DIFFERENTIAL EQUATION OF FRACTIONAL ORDER

Consider the nonlinear Riccati differential equation ([226], [233])

$$\frac{d^{\nu}}{dt^{\nu}} y(t) = -y^2(t) + 1, \quad 0 \le t \le 1 \tag{5.2.1}$$

Subject to the initial condition as $y(0) = 0$.

#### 5.2.1.1 FIRST ORDER CASE

The exact solution of the equation for $\nu = 1$ is given as

$$y(t) = \frac{e^{2t} - 1}{e^{2t} + 1}. \tag{5.2.2}$$

Mathematical modeling of the equation is done with differential equation neural network (DE-NN) represented by equations (3.1.5) and (3.1.6) by taking $10$ number of neurons resulting in $30$ number of unknown parameters or weights. These adaptive weights are restricted to real numbers between $-10$ to $10$. The initial population consists of a set of $160$ numbers of particles. Each particle consists of $30$ agents, which is equivalent to number of weights. Input of the training set is taken from $t \in (0, 1)$ with a step size of $0.1$. Therefore the fitness function, $e_j$, is formulated as

$$e_j = \frac{1}{11}\sum_{i=1}^{11}\left[\frac{d}{dt}\hat{y}(t_i) + \left[\hat{y}(t_i)\right]^2 - 1\right]^2 + \left[\hat{y}(0)\right]^2, \quad j = 1,2,3,\ldots, \tag{5.2.3}$$

where, $j$ is the number of flights, $\hat{y}(t)$ and $d\hat{y}/dt$ are networks provided in equation (3.1.5), and (3.1.6), respectively. Our scheme runs iteratively in order to find the minimum of fitness function, $e_j$, with stoppage criteria as 800 number of flights or value $e_j \le 10^{-08}$, whichever comes earlier. One of the unknown weights learned by DE-NN algorithm with value of, $e_j = 5.2035 \times 10^{-08}$, are provided in Table 5.16.

Using these weights in (3.1.5) one can find the solution to this problem for any input time $t$ between $0$ and 1. The solution obtained by our scheme is provided in Table 5.17. It also contains the reported result of other numerical solvers such as solution by DE-NN based on Genetic Algorithm (DE-NN GA) [225], the fourth-component approximate solution using $10$ terms by Modified Homotopy perturbation method

(HPM) [233], Solution obtained by Adomian decomposition method (ADM) approximated using 11 terms [226], and solution obtained by variational iteration method (VIM) approximated using 3 iterations [235]. It can be seen from Table 5.17 the result obtained by our approach are similar to other numerical techniques and more close to the exact solution.

**Table 5.16** Best Weights Trained for FDE-NN Networks Optimized with PSO-SA Hybrid Algorithm.

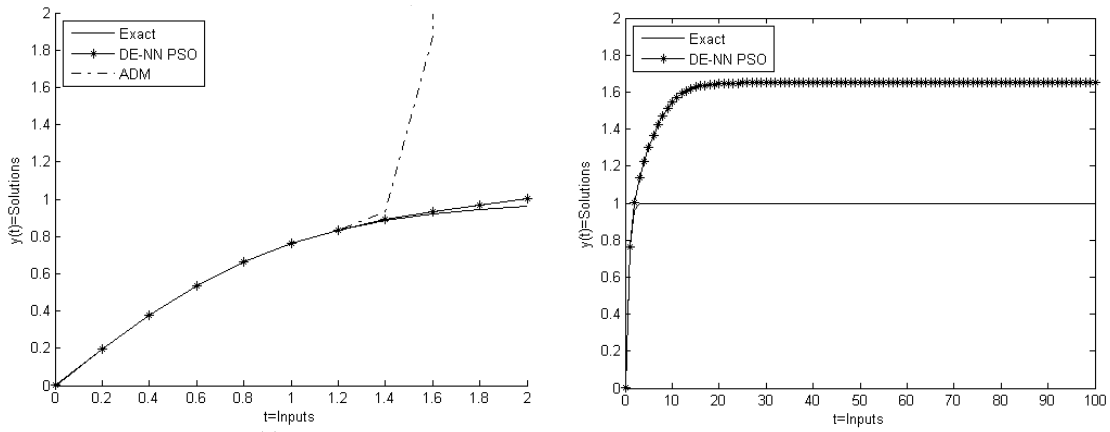
| | $w_i$ | | | $\alpha_i$ | | | $\beta_i$ | | |
|---|---|---|---|---|---|---|---|---|---|
| ***I*** | ν = 1 | ν = 0.5 | ν =0.75 | ν = 1 | ν = 0.5 | ν = 0.75 | ν = 1 | ν = 0.5 | ν = 0.75 |
| 1 | 2.9126 | -0.2862 | 0.9386 | 0.1035 | 0.2971 | 0.4767 | 0.6284 | -0.7362 | 0.8893 |
| 2 | -1.3172 | -0.1675 | 0.8134 | -0.7544 | 0.8051 | 0.1716 | 2.0250 | 0.6830 | -0.6803 |
| 3 | 2.7278 | 0.1429 | -1.8554 | 0.7892 | 0.8953 | -0.4685 | -1.3869 | 0.4173 | -0.8094 |
| 4 | 0.3031 | 0.0290 | 0.6009 | 1.2622 | -0.9547 | 1.1421 | -0.6302 | 0.5121 | -0.4901 |
| 5 | -0.7725 | -14.910 | 0.8612 | 1.1813 | 0.1186 | 0.0775 | -0.0486 | 0.3295 | 1.1524 |
| 6 | 1.1042 | -15.652 | 0.9230 | -0.2408 | -1.0017 | -0.7054 | -0.8391 | -0.8255 | 0.3864 |
| 7 | -1.5585 | -0.1842 | 0.9949 | 0.6595 | -0.1540 | -0.2469 | 0.0321 | -0.9051 | 0.9204 |
| 8 | -2.2683 | 0.2745 | 0.1913 | -1.9361 | -0.9104 | -0.7614 | -1.0197 | -0.7837 | -1.1823 |
| 9 | 2.3799 | -0.3484 | 0.2980 | -0.2635 | 0.1905 | 0.5860 | 1.0555 | 0.8232 | -1.3460 |
| 10 | -4.0783 | -0.9056 | -5.7971 | -0.2094 | -0.4053 | -0.2562 | 0.4381 | 1.0641 | -0.0492 |

**Table 5.17** Comparison of Results for Solution of the Equation for First Order Case.

| **t** | **Exact** | **DE-NN** | | **VIM** | **ADM** | **HPM** |
|---|---|---|---|---|---|---|
| | | *PSO-SA* | *GA* | | | |
| 0.0 | 0.000000 | $4.04\times10^{-9}$ | $2.20\times10^{-8}$ | 0.000000 | 0.000000 | 0.000000 |
| 0.1 | 0.099667 | 0.099665 | -- | 0.099667 | 0.099667 | 0.099668 |
| 0.2 | 0.197375 | 0.197400 | 0.197100 | 0.197375 | 0.197375 | 0.197375 |
| 0.3 | 0.291312 | 0.291351 | -- | 0.291312 | 0.291312 | 0.291312 |
| 0.4 | 0.379948 | 0.379954 | 0.379600 | 0.379946 | 0.379948 | 0.379944 |
| 0.5 | 0.462117 | 0.462081 | -- | 0.462103 | 0.462117 | 0.462078 |
| 0.6 | 0.537049 | 0.537021 | 0.536700 | 0.536983 | 0.537049 | 0.536857 |
| 0.7 | 0.604367 | 0.604390 | -- | 0.604124 | 0.604367 | 0.603631 |
| 0.8 | 0.664036 | 0.664095 | 0.663700 | 0.663300 | 0.664037 | 0.661706 |
| 1.0 | 0.761594 | 0.761596 | 0.761500 | 0.757166 | 0.761622 | 0.746032 |

In order to analyses further, the results are also obtained for different inputs by our algorithm. These results are provided graphically in Fig. 5.6. The traditional

approximate analytic numerical solvers give good approximation only in the neighborhood of the initial position [226], [233], [235]. It can be seen from Fig. 5.6(a) that our scheme gives good approximation for comparatively larger interval whereas the ADM method diverges. Moreover, in our proposed method, the error starts to build up for larger interval $(0, 100)$ but does not diverge earlier as can be seen in Fig. 5.6(b). So, it can be concluded that our approach gives better approximations even for larger intervals and not only in local vicinity of initial position.




(a) (b)

**Fig. 5.6** Comparison of results for problem I (a). is for interval (0, 2) and (b). is for interval (0, 100)

### 5.2.1.2 FRACTIONAL ORDER CASE

The exact solution for nonlinear equation (5.2.1) for fractional order case is not available, however, approximate numerical solutions are available like AMD [226], and HPM [234] etc. Stochastic numerical solution using neural networks optimized with GA [225] is also given for comparison.

In our scheme, mathematical modeling for equation (5.2.1) is done with FDE-NN networks represented by (3.1.8) and (3.1.10) by taking $10$ number of neurons resulting in $30$ number of unknown parameter or weights. These adaptive weights are restricted to real numbers between $-10$ to $10$. The value of the fraction order derivative $\nu$ is taken as $0.5$ and $0.75$. Randomly generated initial swarm consists of a set of $160$ particles. Each particle consists of $30$ elements equal to the number of

unknown weights in networks. Input of the training set is taken for time interval $t \in (0,1)$ with a step size of $0.1$. Therefore the fitness function is formulated as

$$e_j = \frac{1}{11}\sum_{i=1}^{11}\left[\frac{d^{\nu}}{dt^{\nu}}\hat{y}(t_i) + \left[\hat{y}(t_i)\right]^2 - 1\right]^2 + \left[\hat{y}(0)\right]^2, \quad \nu \in (0,1) \quad j = 1,2,3\ldots\ldots \tag{5.2.4}$$

where $j$ be the flight number, and $\hat{y}(t)$ and $d^{\nu}\hat{y}/dt^{\nu}$ are networks given in (3.1.8), and (3.1.10), respectively. Our scheme runs iteratively in order to find the minimum for function, $e_j$, with stoppage criteria as $2000$ number of flights or value of $e_j \le 10^{-06}$. The agents of best fitted particles in term of weights are given in Table 5.16 for order $\nu = 0.5$ and $\nu = 0.75$. These weights can be used in equation (3.1.8) to obtain the solution of the equation for any input time $t$ between $0$ and 1.

The proposed solution for the equation (5.2.1) is obtained for inputs $t \in (0,1)$ with a step size of $0.1$. The results are summarized in Table 5.18. It also contains reported results for the solution of the equation with HPM techniques based on fourth-term approximation [234] and evolutionary computation technique based on neural networks supported with GA algorithm [217]. It can be inferred that our algorithm provides approximate solution to the fractional Riccati differential equation effectively. Moreover, FDE-NN algorithms based on PSO is computationally efficient from GA based, because it does not involve the computationally extensive operation of crossover and mutations [219].

The statistical studies are necessary for the optimization of weights due to its highly stochastic nature. The reliability of results is determined with such analysis. In this regard, $100$ independent runs of the algorithm are executed, in order to make statistical analysis of the result obtained by our method. Results in the form of absolute unsupervised error at different inputs are given in Table 5.19. The parameters best and worst, defined as minimum and maximum, of error are given. The statistical mean and standard deviation are also determined for errors. It can be seen from the average and standard deviation values that reliable results are obtained by our scheme.

**Table 5.18** Comparison of Results of Problem I for Fractional Order Cases.

| t | ν = 0.5 | | | ν = 0.75 | |
|---|---|---|---|---|---|
| | *GA* | *PSO* | *HPM* | *PSO* | *HPM* |
| 0.0 | 0.0505 | 0.009493 | 0.000000 | 0.000798 | 0.000000 |
| 0.1 | | 0.289667 | 0.273875 | 0.165087 | 0.184795 |
| 0.2 | 0.4047 | 0.386489 | 0.454125 | 0.276350 | 0.313795 |
| 0.3 | | 0.441120 | 0.573932 | 0.356196 | 0.414562 |
| 0.4 | 0.5507 | 0.482348 | 0.644422 | 0.416916 | 0.492889 |
| 0.5 | | 0.516379 | 0.674137 | 0.465520 | 0.462117 |
| 0.6 | 0.6178 | 0.544872 | 0.671987 | 0.506004 | 0.597393 |
| 0.7 | | 0.568545 | 0.648003 | 0.540629 | 0.631772 |
| 0.8 | 0.6594 | 0.587895 | 0.613306 | 0.570632 | 0.660412 |
| 0.9 | | 0.603344 | 0.579641 | 0.596636 | 0.687960 |
| 1.0 | 0.6992 | 0.615268 | 0.558557 | 0.618873 | 0.718260 |

**Table 5.19** Statistical Analysis of Results by FDE-NN Networks Optimized with PSO-SA algorithm for Problem I.

| T | Best | | Worst | | Mean | | Standard deviation | |
|---|---|---|---|---|---|---|---|---|
| | ν = 0.5 | ν = 0.75 | ν = 0.5 | ν = 0.75 | ν = 0.5 | ν = 0.75 | ν = 0.5 | ν = 0.75 |
| 0.0 | 2.29e-10 | 5.22e-07 | 1.94e-04 | 12.206 | 9.30e-05 | 0.12720 | 2.11e-05 | 1.2205 |
| 0.2 | 8.69e-05 | 4.53e-07 | 0.4561 | 0.3411 | 2.93e-02 | 2.95e-02 | 5.78e-02 | 6.32e-02 |
| 0.4 | 2.16e-05 | 6.31e-07 | 0.2402 | 0.1688 | 3.78e-03 | 1.16e-02 | 2.40e-02 | 3.02e-02 |
| 0.6 | 9.02e-05 | 5.98e-07 | 7.40e-02 | 0.4403 | 2.81e-03 | 9.44e-03 | 7.88e-03 | 4.44e-02 |
| 0.8 | 3.28e-06 | 6.05e-07 | 9.16e-03 | 0.9329 | 9.80e-04 | 1.17e-02 | 1.79e-03 | 9.33e-02 |
| 1.0 | 9.08e-05 | 6.81e-07 | 0.1176 | 1.2612 | 7.93e-03 | 3.96e-02 | 1.57e-02 | 0.1362 |

Training of the weights for neural networks of the equation is also made by different stochastic optimizers. These optimizers are simulating annealing (SA), Genetic Algorithm (GA), Genetic algorithm hybridized with simulating annealing (GA-SA), PSO, and PSO hybridized with simulating annealing (PSO-SA). MATLAB optimization toolbox with built-in functions for SA and GA are used. In order to make the comparison, $100$ independent runs for each optimizer are made. The values of the fitness function, $e_j$, for inputs $0$ and $1$ are given in Table 5.20 and Table 5.21 for $\nu = 0.5$ and $\nu = 0.75$, respectively. Moreover, in Fig 5.7 the value of unsupervised error, $e_j$, is plotted in ascending order for each independent run of the algorithms. It can be seen that best results are obtained by PSO-SA algorithm.

**Table 5.20** Comparison for Stochastic Solvers Based on the value of Error Function, $e_j$, for Problem I by taking $\nu = 0.5$.

| t | FDE-NN | $\nu = 0.5$ | | | |
|---|---|---|---|---|---|
| | | *Best* | *Worst* | *Mean* | *STD* |
| 0 | SA | 8.62e-04 | 1.29e-02 | 1.47e-03 | 1.83e-03 |
| | GA | 9.40e-05 | 9.79e-04 | 1.22e-04 | 9.64e-05 |
| | GA-SA | 8.91e-05 | 2.72e-04 | 9.90e-05 | 1.93e-05 |
| | PSO | 3.03e-09 | 1.01e-03 | 1.36e-04 | 1.52e-04 |
| | PSO-SA | 2.29e-10 | 1.94e-04 | 9.30e-05 | 2.11e-05 |
| 1 | SA | 6.07e-03 | 5.71214 | 0.46847 | 0.81886 |
| | GA | 9.11e-05 | 8.19e-02 | 1.03e-02 | 1.67e-02 |
| | GA-SA | 9.10e-05 | 8.12e-02 | 9.76e-03 | 1.63e-02 |
| | PSO | 9.08e-05 | 0.11765 | 7.93e-03 | 1.57e-02 |
| | PSO-SA | 9.07e-05 | 0.11764 | 7.24e-03 | 1.57e-02 |

**Table 5.21** Comparison for Stochastic Solvers Based on the value of Error Function, $e_j$, for Problem I by taking $\nu = 0.75$.

| t | FDE-NN | $\nu = 0.75$ | | | |
|---|---|---|---|---|---|
| | | *Best* | *Worst* | *Mean* | *STD* |
| 0 | SA | 6.06e-06 | 0.24519 | 4.02e-03 | 2.54e-02 |
| | GA | 6.88e-07 | 1.64e-02 | 1.02e-03 | 2.43e-03 |
| | GA-SA | 5.88e-07 | 7.39e-05 | 4.87e-06 | 1.07e-05 |
| | PSO | 5.22e-07 | 0.28122 | 5.16e-03 | 2.94e-02 |
| | PSO-SA | 3.01e-8 | 3.16e-02 | 3.18e-04 | 3.16e-04 |
| 1 | SA | 2.96e-05 | 11.3487 | 0.53983 | 1.28091 |
| | GA | 2.08e-06 | 0.17387 | 2.13e-02 | 3.40e-02 |
| | GA-SA | 4.92e-06 | 0.17567 | 1.97e-02 | 3.10e-02 |
| | PSO | 6.81e-07 | 0.33742 | 2.71e-02 | 5.76e-02 |
| | PSO-SA | 6.82e-07 | 1.25906 | 3.83e-02 | 0.13574 |

It is necessary to mention that whole building block of our algorithm for fractional order case is based on MLF function. The calculation for MLF function is also very complex in nature. In our simulation, the MATLAB code for MLF function, patent by Podlubny at MATLAB central file exchange, is used [230]. The equation is solved with FDE-NN networks optimized with GA (160 individuals, 2000 generations) and

PSO (160 particle, 2000 flights) algorithms for time analysis. Average total time of execution for single run of algorithms, $T_{time}$, and time taken for calculation of MLF function only, $T_{MLF}$, is provided in Table 5.22. It can be inferred from the Table 5.22 that about 82% of execution time is spent on calculation of MLF function only. In this paper, the time analysis are carried using Dell Latitude D630, Intel(R) Core(TM) 2 Duo CPU T9300, 2.50GHz and Matlab version R2008b.

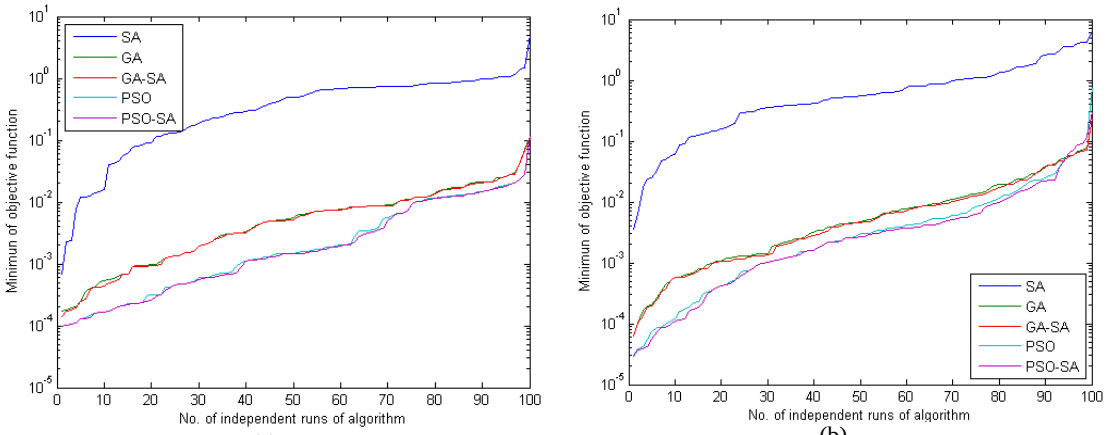



**Fig. 5.7** Comparison of FDE-NN optimized by stochastic numerical solvers, (a). is for ν = 0.5 and (b). is for ν = 0.75.

**Table 5.22** The Effects for Varying Intervals and Steps on Execution time and Accuracy of the Algorithms.

| (0,T) | Step size | Steps | Algo. | MLF calls Million | Average Values for ν = 0.5 | | | Average Values for ν = 0.75 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | $T_{time}$ | $T_{MLF}$ | $e_j$ | $T_{Time}$ | $T_{MLF}$ | $Ej$ |
| (0,1) | 0.10 | 11 | PSO | 3.5552 | 1305s | 1066s | 2.84e-03 | 1258s | 1026s | 3.48e-03 |
| | | | GA | 3.5552 | 1313s | 1051s | 4.98e-03 | 1313s | 1052s | 4.72e-03 |
| (0,2) | 0.10 | 21 | PSO | 6.7872 | 2348s | 1924s | 9.04e-03 | 2351s | 1931s | 9.82e-03 |
| | | | GA | 6.7872 | 2426s | 1972s | 6.97e-02 | 2462s | 2006s | 7.63e-02 |
| (0.1) | 0.05 | 21 | PSO | 6.7872 | 2386s | 1954s | 2.74e-03 | 2386s | 1953s | 2.91e-03 |
| | | | GA | 6.7872 | 2430s | 1992s | 4.92e-03 | 2469s | 2010s | 4.70e-03 |

The effect of change of intervals and step size on execution time, as well as, accuracy of algorithms is also evaluated. It can be seen from Table 5.22 that as the input span $(0, 1)$ is increased to $(0, 2)$ without changing the step size, the accuracy of the

algorithm decreases, but the computational time increases. As the step size is decreased from $0.1$ to $0.05$ without changing the interval size, the accuracy of the algorithms increase slightly, but it also increases computational time. Therefore, it can be concluded that there is always tradeoff between the computational complexity and accuracy while selecting the values of intervals and step sizes.

### 5.2.2 PROBLEM II: COMPLEX RICCATI DIFFERENTIAL EQUATION

Another form of Riccati differential equation of arbitrary order is taken to investigate the strength and weaknesses of the proposed stochastic algorithm. Consider the quadratic Riccati differential equation of fractional order ([236], [237])

$$\frac{d^{\nu}}{dt^{\nu}} y(t) = 2y(t) - y^2(t) + 1, \quad 0 \le t \le 1, \tag{5.2.5}$$

subject to the initial condition as $y(0) = 0$.

#### 5.2.2.1 FIRST ORDER CASE

The exact solution of the above equation for order $\nu = 1$ is given as

$$y(t) = 1 + \sqrt{2}\tanh(\sqrt{2}t) + \frac{1}{2}\log(\frac{\sqrt{2}-1}{\sqrt{2}+1}). \tag{5.2.6}$$

This problem is solved with the same methodology as adopted in example I. The termination criterion is that the value for function $e_j \le 10^{-08}$ or maximum number of flights is equal to $1000$. The fitness function, $e_j$, in case can be given as

$$e_j = \frac{1}{11}\sum_{i=1}^{11}\left[\frac{d}{dt}\hat{y}(t_i) + \left[\hat{y}(t_i)\right]^2 - 2\hat{y}(t_i) - 1\right]^2 + \left[\hat{y}(0)\right]^2, \quad j = 1,2,3\ldots\ldots \tag{5.2.7}$$

where $\hat{y}(t)$ and $d\hat{y}/dt$ are networks provides in equations (3.1.5), and (3.1.6), respectively, using $10$ number of neurons. The best unknown weights trained stochastically by our scheme are given in Table 5.23. These weights are used in expression (3.1.5) for finding the solution of the equations for some inputs between $0$

and 1. Results are summarized in Table 5.24. It also includes the reported results of HPM technique using fourth-term approximate solution [234], VIM based on 3 iterations [235], Chebyshev Wavelets (CW) using 192 set of block pulse functions [238], generalized differential transform method (GDTM) using 5 terms [216] and FDE-NN GA [225]. It can be inferred from the results that in this context our method performs comparable to or is better than specialized state-of-art numerical solvers.

**Table 5.23** Best Weights Trained for Neural Networks by PSO-SA in Problem II.

| $i$ | $w_i$ | | | $\alpha_i$ | | | $\beta_i$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $\nu = 1$ | $\nu = 0.5$ | $\nu = 0.75$ | $\nu = 1$ | $\nu = 0.5$ | $\nu = 0.75$ | $\nu = 1$ | $\nu = 0.5$ | $\nu = 0.75$ |
| 1 | -0.1781 | -0.2617 | 2.7969 | -0.6315 | 0.1442 | 0.3068 | 1.3428 | -0.7450 | 0 .1628 |
| 2 | 0.9627 | -1.8771 | -0.1816 | -0.0041 | -1.2903 | 0.9164 | -1.4170 | -0.0531 | 1.3084 |
| 3 | 0.8416 | 0.3915 | -1.6007 | 0.3301 | 0.6852 | -0.3002 | -0.5538 | -1.9151 | 1.0313 |
| 4 | 0.7852 | -9.7906 | 0.9376 | 0.3234 | -0.7260 | 0.5739 | -2.9840 | -0.4774 | 0.4451 |
| 5 | 0.6856 | -0.0861 | -1.9930 | -0.3826 | 0.2467 | 0.2111 | -1.0800 | -1.0307 | -0.9704 |
| 6 | -2.1440 | -0.0304 | 0.7810 | -0.0137 | 0.2348 | 0.2406 | -0.5275 | 1.6564 | -0.9591 |
| 7 | -0.3696 | -5.4479 | -0.2021 | -0.2856 | -0.2731 | -0.0024 | -0.5790 | -0.1701 | -0.3669 |
| 8 | -1.1879 | 0.6175 | -0.4029 | 0.0322 | 0.1959 | -1.6387 | -0.4203 | -2.7192 | 1.0064 |
| 9 | -0.7665 | 0.3376 | 2.4966 | 0.2818 | 0.4952 | -1.2774 | 0.1399 | -1.1731 | -0.6628 |
| 10 | 2.8433 | -1.0567 | 0.2644 | 2.7946 | 0.3140 | 1.4747 | -1.7701 | -0.1915 | -0.2350 |

**Table 5.24** Comparison of Results for Problem II for First Order Case.

| t | Exact | DE-NN | | HPM | CW | VIM | GDTM |
|---|---|---|---|---|---|---|---|
| | | *GA* | *PSO-SA* | | | | |
| 0.0 | 0.000000 | $2.97\times10^{-9}$ | $6.58\times10^{-9}$ | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 0.1 | 0.110295 | 0.1103 | 0.110328 | 0.110294 | 0.110311 | 0.110295 | 0.110296 |
| 0.2 | 0.241976 | 0.2420 | 0.241997 | 0.241965 | 0.241995 | 0.241977 | 0.241997 |
| 0.3 | 0.395104 | 0.3950 | 0.395101 | 0.395106 | 0.395123 | 0.395113 | 0.395231 |
| 0.4 | 0.567812 | 0.5677 | 0.567797 | 0.568115 | 0.567829 | 0.567845 | 0.568158 |
| 0.5 | 0.756014 | 0.7559 | 0.756008 | 0.757564 | 0.756029 | 0.756086 | 0.756250 |
| 0.6 | 0.953566 | 0.9536 | 0.953580 | 0.958259 | 0.953576 | 0.953666 | 0.951475 |
| 0.7 | 1.152948 | 1.1531 | 1.152973 | 1.163459 | 1.152955 | 1.153037 | 1.141385 |
| 0.8 | 1.346363 | 1.3465 | 1.346374 | 1.365240 | 1.346365 | 1.346379 | 1.343061 |
| 0.9 | 1.526911 | 1.5270 | 1.526890 | 1.554960 | 1.526909 | 1.526411 | 1.498164 |
| 1.0 | 1.689498 | 1.6896 | 1.689459 | 1.723810 | 1.689494 | 1.686027 | 1.600000 |

#### 5.2.2.2 FRACTIONAL ORDER CASES

The exact solution is not available for the Riccati differential equation of fractional order (5.1.17). However, for such equations solutions are determined by number of approximate analytic solvers such as HPM [234], Homotopy analysis method (HAM) [237] and CW techniques [238]. We have also applied our design scheme to solve the equation (5.1.17) for such cases.

The solutions are determined for input time $t \in (0,1)$ with a step size of $0.1$. The parameter setting for execution of algorithm is similar to the previous examples for fractional order cases. However, the function, $e_j$, developed for this case is given as

$$e_j = \frac{1}{11}\sum_{i=1}^{11}\left[\frac{d^{\nu}}{dt^{\nu}}\hat{y}(t_i) + \left[\hat{y}(t_i)\right]^2 - 2\hat{y}(t_i) - 1\right]^2 + \left[\hat{y}(0)\right]^2, \quad j = 1,2,3\ldots\ldots \qquad (5.2.8)$$

where $j$ be the number of flight index, $\nu$ is fractional order and $\hat{y}(t)$ and $d^{\nu}\hat{y}/dt^{\nu}$ are networks given in (3.1.8) and (3.1.10), respectively. The agents of best fitted particle in terms of weights are given in Table 23 for order $\nu = 0.5$ and $\nu = 0.75$. The results obtained in our approach (PSO) and other reported numerical techniques are provided in Table 5.25. The solution obtained by HPM technique based on fourth-component approximate solution with 16 terms [234]. The results of CW method are based on $192$ sets of block pulse functions [238]. It can be inferred from the results that our algorithm can also provide approximate solutions of the fractional Riccati differential equation.

The statistical analysis of the result obtained by our method on the basis of 100 independent runs of our algorithm is provided in Table 5.26. Results are given in the form of absolute unsupervised error at different inputs. It further validates the applicability and effectiveness of our methodology.

Training of the weights for neural networks of the equation is also made by SA, GA, GA-SA, PSO, PSO-SA optimizers. Statistical analysis is made on the basis of 100 independent runs for each optimizer. The values of unsupervised error function, $e_j$,

for some inputs are given in Table 5.27. It can be seen that PSO-SA algorithm provides better results. Moreover, the minimum value of function, $e_j$, is also plotted in ascending order in Fig 5.8. It can be seem from Fig. 5.8(a) that the least value of, $e_j$, is obtained in PSO-SA algorithm. The effectiveness of the simulating annealing can be seen in Fig. 5.8(b). This makes the average results slightly better than without it.

**Table 5.25** Comparison of Results for Fractional Order Cases of Problem II.

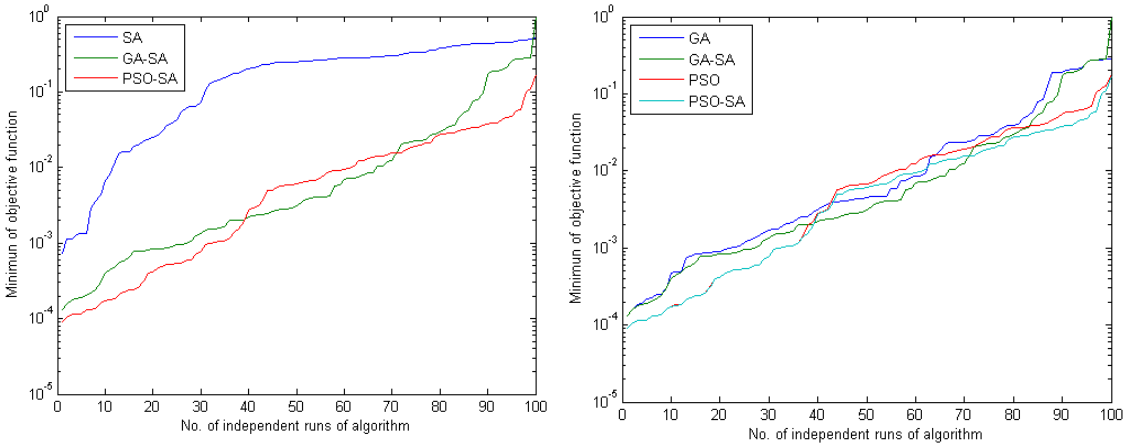
| T | **ν = 0.5** | | | | **ν = 0.75** | | |
|---|---|---|---|---|---|---|---|
| | *GA* | *PSO* | *HPM* | *CW* | *PSO* | *HPM* | *CW* |
| 0.0 | 0.0561 | 0.009420 | 0.000000 | 0.000000 | 0.000705 | 0.000000 | 0.000000 |
| 0.1 | 0.5610 | 0.574648 | 0.321730 | 0.592756 | 0.283503 | 0.216866 | 0.310732 |
| 0.2 | 0.9121 | 0.890890 | 0.629666 | 0.933179 | 0.539352 | 0.428892 | 0.584307 |
| 0.3 | 1.1594 | 1.090716 | 0.940941 | 1.173983 | 0.768804 | 0.654614 | 0.822173 |
| 0.4 | 1.3369 | 1.230069 | 1.250737 | 1.346654 | 0.971833 | 0.891404 | 1.024974 |
| 0.5 | 1.4671 | 1.334181 | 1.549439 | 1.473887 | 1.147939 | 1.132763 | 1.198621 |
| 0.6 | 1.5652 | 1.415512 | 1.825456 | 1.570571 | 1.296320 | 1.370240 | 1.349150 |
| 0.7 | 1.6414 | 1.480918 | 2.066523 | 1.646199 | 1.416139 | 1.594278 | 1.481449 |
| 0.8 | 1.7024 | 1.534604 | 2.260633 | 1.706880 | 1.506936 | 1.794879 | 1.599235 |
| 0.9 | 1.7528 | 1.579396 | 2.396839 | 1.756644 | 1.569252 | 1.962239 | 1.705303 |
| 1.0 | 1.7957 | 1.617332 | 2.466004 | 1.798220 | 1.605580 | 2.087384 | 1.801763 |

**Table 5.26** Statistical Analysis of Results for Problem II using FDE-NN Networks Optimized with PSO-SA Algorithm.

| T | **Best** | | **Worst** | | **Mean** | | **Standard deviation** | |
|---|---|---|---|---|---|---|---|---|
| | ν = 0.5 | ν = 0.75 | ν = 0.5 | ν = 0.75 | ν = 0.5 | ν = 0.75 | ν = 0.5 | ν = 0.75 |
| 0.00 | 7.57e-06 | 3.93e-07 | 1.67e-03 | 2.25457 | 4.38e-04 | 4.69e-02 | 3.99e=04 | 0.240993 |
| 0.20 | 9.08e-05 | 9.42e-07 | 0.974512 | 1.49943 | 2.91e-02 | 0.226850 | 9.82e-02 | 0.454657 |
| 0.40 | 9.02e-05 | 9.11e-04 | 4.51e-02 | 0.99943 | 5.42e-03 | 0.186886 | 9.02e-03 | 0.300739 |
| 0.60 | 8.87e-05 | 2.24e-06 | 2.84e-02 | 0.75562 | 3.09e-03 | 0.111913 | 5.28e-03 | 0.172089 |
| 0.80 | 9.02e-05 | 2.25e-06 | 2.47e-02 | 0.68828 | 3.04e-03 | 2.93e-02 | 5.24e-03 | 0.079974 |
| 1.00 | 9.01e-05 | 5.08e-07 | 4.38e-02 | 0.76585 | 4.96e-03 | 0.100795 | 7.84e-03 | 0.172544 |

Moreover, effects on the accuracy of results by changing the intervals and step size for this example were also evaluated. The summary of the results are provided in

Table 5.28. The value of the unsupervised error, $e_j$, as well as, total execution time of algorithms increases by changing the interval size from $(0,1)$ to $(0,2)$ without changing the step size. As step size decrease from $0.1$ to $0.05$ the total time of execution for algorithm, as well as, the accuracy increases. In addition to that, the time taken by MLF function, $T_{MLF}$, in the total time of execution of algorithm, $T_{MLF}$, is also determined. The average values of $T_{MLF}$, and $T_{time}$ for fractional order $\nu = 0.5$ and $\nu = 0.75$ for the solution of equation (5.2.5) due to PSO, GA are provided in Table 5.28. The parameter setting for PSO and GA are same as that of example 1. It can be seen that 81% of total time on average spent on calculation of MLF function.




(a) (b)

**Fig. 5.8** Graphical representation of results obtained by different stochastic numerical technique for example III by taking fractional order $\nu = 0.5$.

**Table 5.27** Comparison of Results for Solvers by Taking ν = 0.5.

| t | FDE-NN | Best | Worst | Mean | STD |
|---|---|---|---|---|---|
| 0 | SA | 3.83e-05 | 3.00e-02 | 2.81e-03 | 4.95e-03 |
| | GA | 8.74e-05 | 5.76e-03 | 3.83e-04 | 7.16e-04 |
| | GA-SA | 8.85e-05 | 3.83e-04 | 1.23e-04 | 5.42e-05 |
| | PSO | 8.94e-05 | 4.57e=-2 | 2.86e-03 | 7.45e-03 |
| | PSO-SA | 7.57e-06 | 1.67e-03 | 4.38e-04 | 3.99e-04 |
| 1 | SA | 1.05e-04 | 0.518259 | 5.19e-02 | 7.31e-02 |
| | GA | 9.37e-05 | 0.184723 | 2.14e-02 | 3.26e-02 |
| | GA-SA | 8.69e-05 | 0.103180 | 1.42e-02 | 2.22e-02 |
| | PSO | 9.01e-05 | 9.83e-02 | 6.98e-03 | 1.35e-02 |
| | PSO-SA | 9.01e-05 | 4.38e-02 | 4.96e-03 | 7.84e-03 |

**Table 5.28** Effects on Accuracy and Execution Time by Varying Intervals and Steps.

| (0,T) | Step size | Steps | Algo. | MLF calls Million | Average Values for ν = 0.5 | | | Average Values for ν = 0.75 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | $T_{time}$ | $T_{MLF}$ | $e_j$ | $T_{Time}$ | $T_{MLF}$ | $e_j$ |
| (0,1) | 0.10 | 11 | PSO | 3.5552 | 1314s | 1065s | 2.94e-03 | 1326s | 1032s | 3.04e-03 |
| | | | GA | 3.5552 | 1328s | 1051s | 9.36e-03 | 1334s | 1053s | 9.98e-03 |
| (0,2) | 0.10 | 21 | PSO | 6.7872 | 2420s | 1925s | 8.37e-03 | 2451s | 1936s | 8.24e-03 |
| | | | GA | 6.7872 | 2485s | 1971s | 1.18e-02 | 2462s | 1969s | 1.93e-02 |
| (0.1) | 0.05 | 21 | PSO | 6.7872 | 2434s | 1951s | 2.63e-03 | 2435s | 1928s | 3.01e-03 |
| | | | GA | 6.7872 | 2473s | 1996s | 9.62e-03 | 2450s | 1964s | 9.92e-03 |

### 5.2.3 CONCLUSION

On the basis of the simulations and results obtained in the last section, it can be concluded that nonlinear Riccati differential equation of arbitrary order can be solved by heuristic computational intelligence such as Particle swarm optimization algorithm supported with simulating annealing. The differential equation neural networks trained by PSO-SA algorithm is a better stochastic optimizer compared to SA, GA, GA-SA PSO algorithms. On the basis of the statistical analysis it can be stated that our proposed computing approach is reliable, effective and easily applicable to complex nonlinear differential equations of arbitrary order. In our future work, we intend to use other biologically inspired computational intelligence algorithms like ant-colony optimization and bee-colony optimization to solve these problems.

## 5.3 FRACTIONAL ORDER SYSTEM BASED ON BAGLEY-TORVIK EQUATION

In this section, fractional order systems represented by Bagley-Torvik FDE differential equations are solved by the designed heuristic computational intelligence techniques based on FDE-NN network and PSO algorithm. The results of simulation are narrated for three linear and non-linear problems of fractional order systems given by Bagley-Torvik equations. Comparison of the design scheme is made with available exact solutions, stochastic and deterministic numerical solvers. The necessary

discussion on results is also provided. Moreover, the material presented here is based on the publication [239].

### 5.3.1 PROBLEM I: BAGLEY-TORVIK EQUATION WITH KNOWN EXACT SOLUTION

In this problem, a fractional order system of Bagley-Torvik equation is taken, which has known exact solution, to check the validity and applicability of the proposed design scheme. Let us take the Bagley-Torvik equation in the form as in [228], [240].

$$D^2 y(t) + D^{\nu} y(t) + y(t) = 2 + 4\sqrt{t/\pi} + t^2, \quad 0 < t \le 1 \tag{5.3.1}$$

with the initial condition and boundary condition given as

$$y(0) = y'(0) = 0, \quad y(1) = 1, \quad y'(1) = 2 \tag{5.3.2}$$

The exact solution of the equation is given as

$$y(t) = t^2. \tag{5.3.3}$$

Mathematical model of the equation is made with FDE-NN networks given in equation (3.1.8) to (3.1.10) by taking 10 numbers of neurons resulting in $30$ number of unknown parameters or weights. These weights are restricted to be real numbers between $-10$ to $10$. The training of weights is made with PSO-PS algorithm, a hybrid computational intelligence algorithm. The parameter values setting for the execution of the algorithm are provided in Table 5.29.

Input of the training set is taken from $t \in (0, 1)$ with a step size $0.2$. Therefore, the fitness function, *e*, is formulated as

$$e = \frac{1}{1 + e_j} \quad j = 1,2,\ldots\ldots \tag{5.3.4}$$

where, $e_j$, is given as

$$e_j = \left. \begin{array}{l} \frac{1}{6}\sum_{i=1}^{6}[D^2\hat{y}(t_i) + D^{\nu}\hat{y}(t_i) + \hat{y}(t_i) - 2 - 4\sqrt{t_i/\pi} - t_i{}^2]^2 + [\hat{y}(0)]^2 \\ + [\hat{y}(1) - 1]^2 + [\hat{y}'(0)]^2 + [\hat{y}'(1) - 2]^2 \end{array} \right|_j \quad (5.3.5)$$

where, $j$ is the number of flights, $\nu = 3/2$, and $\hat{y}(t)$, $D^2\hat{y}$, and $D^{\nu}\hat{y}(t)$ are networks provided in equations (3.1.8), (3.1.9) and (3.1.10), respectively.

**Table 5.29** Parameters setting of the algorithms.

| **PSO** | | **PS** | |
|---|---|---|---|
| *Parameters* | *Value/Setting* | *Parameters* | *Value/Setting* |
| Initial population or Swarm Size | 200 | Poll Method | Generalized PS Positive basis 2N |
| Particle size | 30 | Poll order | Consecutive |
| No. of Flights | 2000 | Mesh Size | (1, 10) |
| Local acceleration factor | Linear decreasing between 4 to 0.5 | Expansion factor | 2 |
| Global acceleration factor | Linear increasing between 0.5 to 4 | Contraction factor | 0.5 |
| Inertia weight | Linearly decreasing between 0.9 to 0.2 | Maximum function Evaluation | 3000 |
| $v_{max}$ | 04 | Maximum Iteration | 50000 |

Our scheme runs iteratively in order to find the maximum of fitness function, $e$, or maximum number of flights as $2000$, whichever comes earlier. Then, the best particle found so far is used in rapid local search algorithm as a starting point for further refinements. One set of the weights of FDE-NN learned stochastically by PSO-PS algorithm are given in Table 5.30. It also includes the reported result in which weights are learned by GA-PS algorithm [227]. The solution of the equation for any input time $t$ between $0$ and $1$ can be determined by using Table 5.30 and equation (3.1.8).

The solution of the equation by deterministic numerical solver like Podlubny matrix approach (PMA) [102] and He's variational iteration method (HVIM) [99] are provided in Table 5.31 as a reported result. It also includes the results of our proposed designed scheme as well as solution by GA-PS algorithm [227]. The PMA results

based on the library functions, given at MATLAB central file exchange, patent by Podlubny [229]. The following setting is employed for parameters. Step of discretization is h = 0.01, total number of steps 100, setting the value of fractional order derivative as $\nu = 1.5$, and the value of constant coefficient of the equation is 1, whereas result of solution of equation by HVIM method are taken form article [99]. It can be seen form Table 5.31 that the value of mean error of our proposed scheme is better than that of GA-PS and HVIM algorithms and comparable with PMA algorithm. However, it should also be noted that the PMA algorithm result are determined at much higher computation cost due to very small step size 0.01 compared to 0.2 for our scheme.

**Table 5.30** A set of FDE-NN Weights Trained by Solvers for Problem I

| *I* | **Reported Results (GA-PS)** | | | **Our Results(PSO-PS)** | | |
|---|---|---|---|---|---|---|
| | $w_i$ | $\alpha_i$ | $\beta_i$ | $w_i$ | $\alpha_i$ | $\beta_i$ |
| 1 | 0.07681 | -1.16770 | 0.04291 | -1.03449 | 1.45013 | -0.85097 |
| 2 | -0.15713 | 0.30471 | -0.73877 | 0.87250 | 0.31841 | 1.12423 |
| 3 | 3.66950 | -0.16469 | -8.12463 | -0.33075 | -0.00428 | -0.76606 |
| 4 | 0.00046 | -0.74905 | 0.05083 | -0.00278 | -0.49270 | 1.37384 |
| 5 | 1.16911 | -0.14210 | -0.00566 | -1.45464 | 0.66446 | 0.03498 |
| 6 | -1.30293 | 0.20979 | 0.24825 | -2.09966 | -0.33922 | -0.78875 |
| 7 | -0.90253 | 0.28461 | 0.64849 | 0.57286 | 1.21462 | -0.45543 |
| 8 | -5.09381 | 3.92250 | -7.62717 | -0.50976 | -0.20076 | 1.03642 |
| 9 | -4.64415 | 0.01437 | -9.37687 | 0.74570 | -1.09118 | -1.29133 |
| 10 | 0.94374 | 0.27168 | 1.50233 | 0.59957 | -0.88360 | -2.31774 |

Moreover, derivative of the solution can be determined by using Table 5.30 and equation (3.1.9). The results of PSO-PS hybrid technique along with reported results are provided in Table 5.32. It can be seen that the results obtained by PSO-PS method are better than that of PS, GA, GA-PS algorithms.

**Table 5.31** Comparison of Results for the Solution of Problem I.

| T | y(t) | Reported Result | | | | | | Our Result | |
|---|---|---|---|---|---|---|---|---|---|
| | | $\hat{y}(t)$ | | | $\lvert y(t)-\hat{y}(t)\rvert$ | | | $\hat{y}(t)$ | $\lvert y(t)-\hat{y}(t)\rvert$ |
| | | GA-PS | PMA | HVIM | GA-PS | PMA | HVIM | PSO-PS | PSO-PS |
| 0.0 | 0.00 | 0.03346 | 0.00000 | 0.00000 | 3.34e-2 | 0.00000 | 0.00000 | 0.00212 | 2.12e-3 |
| 0.1 | 0.01 | 0.04437 | 9.29e-3 | 0.01005 | 3.43e-2 | 7.04e-4 | 5.48e-5 | 0.01220 | 2.20e-3 |
| 0.2 | 0.04 | 0.07338 | 0.03892 | 0.04063 | 3.33e-2 | 1.07e-3 | 6.31e-4 | 0.04263 | 2.63e-3 |
| 0.3 | 0.09 | 0.12044 | 0.08873 | 0.09266 | 3.04e-2 | 1.26e-3 | 2.66e-3 | 0.09298 | 2.98e-3 |
| 0.4 | 0.16 | 0.18573 | 0.15867 | 0.16748 | 2.57e-2 | 1.32e-3 | 7.48e-3 | 0.16297 | 2.97e-3 |
| 0.5 | 0.25 | 0.26961 | 0.24871 | 0.26679 | 1.96e-2 | 1.28e-3 | 1.67e-2 | 0.25246 | 2.46e-3 |
| 0.6 | 0.36 | 0.37262 | 0.35883 | 0.39277 | 1.26e-2 | 1.16e-3 | 3.22e-2 | 0.36149 | 1.49e-3 |
| 0.7 | 0.49 | 0.49549 | 0.48903 | 0.54806 | 5.49e-3 | 9.67e-4 | 5.80e-2 | 0.49027 | 2.67e-4 |
| 0.8 | 0.64 | 0.63911 | 0.63928 | 0.73588 | 8.80e-4 | 7.14e-4 | 9.58e-2 | 0.63917 | 8.34e-4 |
| 0.9 | 0.81 | 0.80457 | 0.80958 | 0.96007 | 5.42e-3 | 4.12e-4 | 1.50e-1 | 0.80873 | 1.27e-3 |
| 1.0 | 1.00 | 0.99308 | 0.99993 | 1.22519 | 6.91e-3 | 6.83e-5 | 2.25e-1 | 0.99970 | 3.05e-4 |
| | | | | **Mean** | 1.89e-2 | 8.14e-4 | 5.35e-2 | | 1.78e-3 |

**Table 5.32** Comparison of Results for Derivative of the Solution of Problem I.

| t | $\hat{y}'(t)$ | Reported Result | | | | | | Our Result | |
|---|---|---|---|---|---|---|---|---|---|
| | | $\hat{y}(t)$ | | | $\lvert y(t)-\hat{y}(t)\rvert$ | | | $\hat{y}(t)$ | $\lvert y(t)-\hat{y}(t)\rvert$ |
| | | PS | GA | GA-PS | PS | GA | GA-PS | PSO-PS | PSO-PS |
| 0.0 | 0.00 | 0.00267 | 0.13526 | 0.017687 | 2.67e-3 | 1.80e-2 | 1.76e-2 | 0.002837 | 2.84e-3 |
| 0.1 | 0.20 | 0.24906 | 0.30350 | 0.199858 | 4.90e-2 | 9.07e-2 | 1.41e-4 | 0.203398 | 3.40e-3 |
| 0.2 | 0.40 | 0.49494 | 0.47211 | 0.380284 | 9.49e-2 | 1.20e-2 | 1.97e-2 | 0.404523 | 4.52e-3 |
| 0.3 | 0.60 | 0.73366 | 0.64301 | 0.561324 | 0.13366 | 1.34e-2 | 3.86e-2 | 0.602020 | 2.02e-3 |
| 0.4 | 0.80 | 0.96318 | 0.81805 | 0.745050 | 0.16318 | 3.14e-2 | 5.49e-2 | 0.797445 | 2.55e-3 |
| 0.5 | 1.00 | 1.18047 | 0.99909 | 0.933359 | 0.18047 | 2.07e-2 | 6.66e-2 | 0.992412 | 7.59e-3 |
| 0.6 | 1.12 | 1.38125 | 1.18798 | 1.128047 | 0.18125 | 6.03e-2 | 7.19e-2 | 1.188575 | 1.14e-2 |
| 0.7 | 1.40 | 1.55960 | 1.38659 | 1.330850 | 0.15960 | 1.80e-2 | 6.91e-2 | 1.387622 | 1.24e-2 |
| 0.8 | 1.60 | 1.70751 | 1.59685 | 1.543479 | 0.10751 | 9.07e-4 | 5.65e-2 | 1.591275 | 8.72e-3 |
| 0.9 | 1.80 | 1.81424 | 1.82073 | 1.767629 | 0.01424 | 1.20e-2 | 3.23e-2 | 1.801289 | 1.29e-3 |
| 1.0 | 2.00 | 1.86549 | 2.06033 | 2.004987 | 0.13450 | 1.34e-2 | 4.98e-3 | 2.019460 | 1.95e-2 |
| | | | | **Mean** | 1.11e-1 | 2.64e-2 | 3.93e-2 | | 6.93e-3 |

Training of the weights for such networks representing the equation is of extremely stochastic nature. It is, therefore, necessary to calculate the statistical parameter for in-depth analysis of the results. In this regard, the four statistical modes, minimum, maximum, mean, and standard deviation, are chosen. The minimum and maximum of error are named as best and worst results, respectively. A total number 125

independents runs are executed for training of weights for FDE-NN networks with PSO-PS algorithm. The statistical parameters for best 100 are provided in Table 5.33 for input at 0 and 1. Table 5.33 also contained the reported result of PS, GA, and GA-PS algorithms for comparison [227].

**Table 5.33** Statistical Parameter Based on the Value of Absolute Error for Problem I.

| t | **Modes** | **Reported Result** | | | | | | **Our Result** | |
|---|---|---|---|---|---|---|---|---|---|
| | | *PS* | | *GA* | | *GA-PS* | | *PSO-PS* | |
| | | $\|y(t)-\hat{y}(t)\|$ | $\|\acute{y}(t)-\hat{y}'(t)\|$ | $\|y(t)-\hat{y}(t)\|$ | $\|\acute{y}(t)-\hat{y}'(t)\|$ | $\|y(t)-\hat{y}(t)\|$ | $\|\acute{y}(t)-\hat{y}'(t)\|$ | $\|y(t)-\hat{y}(t)\|$ | $\|\acute{y}(t)-\hat{y}'(t)\|$ |
| 0 | Best | 2.36e-2 | 2.91e-2 | 4.87e-2 | 6.13e-6 | 3.11e-3 | 8.31e-4 | 7.38e-7 | 9.02e-4 |
| | Worst | 1.00232 | 0.65725 | 0.18695 | 6.63e-2 | 0.17193 | 5.82e-2 | 1.49e-2 | 7.03e-2 |
| | Mean | 0.32789 | 0.27233 | 0.14635 | 2.99e-2 | 0.12111 | 2.90e-2 | 4.46e-3 | 2.55e-2 |
| | Std | 0.24679 | 0.19051 | 3.25e-2 | 1.78e-2 | 4.14e-2 | 1.70e-2 | 4.45e-3 | 1.72e-2 |
| 1 | Best | 2.35e-3 | 9.59e-3 | 4.61e-4 | 1.48e-3 | 1.06e-4 | 2.62e-2 | 1.08e-5 | 4.81e-4 |
| | Worst | 0.82603 | 1.00357 | 7.19e-2 | 0.12491 | 6.87e-2 | 0.11201 | 1.35e-2 | 0.10629 |
| | Mean | 0.24834 | 0.40912 | 3.36e-2 | 7.54e-2 | 3.31e-2 | 6.17e-2 | 5.32e-3 | 5.07e-2 |
| | Std | 0.20391 | 0.28778 | 2.12e-2 | 3.35e-2 | 2.00e-2 | 3.37e-2 | 3.81e-3 | 3.14e-2 |

The value of unsupervised error given by the function $e_j$ of the equation is plotted in descending order against number of independent runs for the stochastic solvers in Fig. 5.9. It can be seen that on average PSO-SA algorithm provides better convergence for error function $e_j$ than that of PS and GA-PS techniques.

The values of the absolute error for the solution at 0 and 1 are plotted in Fig. 5.10(a) and Fig. 5.10(b), respectively, against the number of independent run of the algorithms. Moreover, the behavior of the derivative of the solution is also analyzed by plotted it values in Fig. 5.10(c) and Fig 5.10(d) for input timing 0 and 1, respectively, against the number of independent runs of the stochastic solvers.

It can be inferred from Fig. 5.9 and 5.10, as well as Table 33 that the best results for FDE-NN networks are obtained by PSO-PS algorithm as compared with PS and GA-PS techniques.

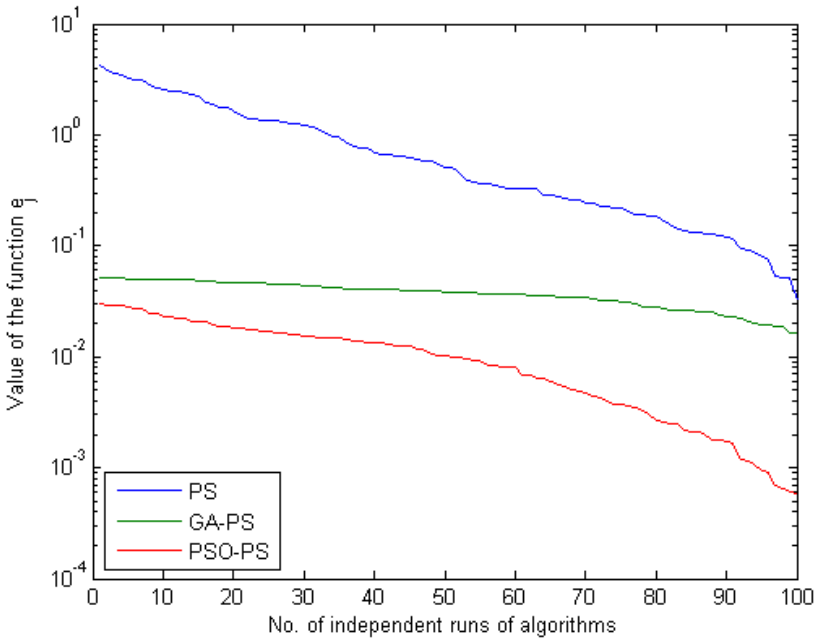
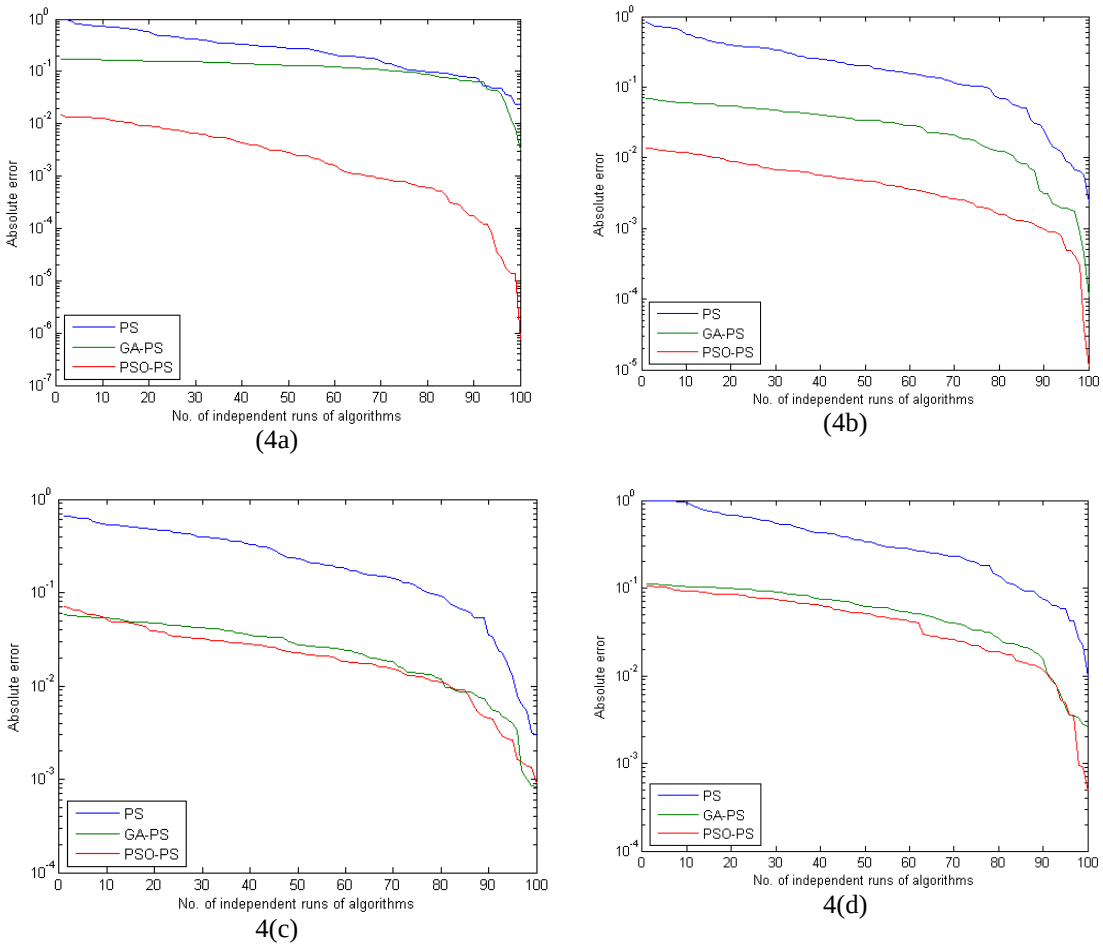



**Fig. 5.9** The value of function $e_j$ obtained by different stochastic numerical solvers for example 1




(4a) (4b) 4(c) 4(d)

**Fig. 5.10** Comparison of stochastic numerical solvers, (4a) and 4(b) the value of |y(t)-ŷ(t| at t = 0 and t = 1 respectively, 4(c) and 4(d) the value of |ý(t)-ŷ́(t| at t = 0 and t = 1 respectively.

### 5.3.2 PROBLEM II: BAGLEY-TORVIK EQUATION WITH INHOMOGENEOUS INITIAL CONDITIONS

In this example, a fractional order system of Bagley-Torvik equation is taken for which PMA method is unable to determine the solution due to non-homogeneous initial condition. However, our intention is to check the applicability of our scheme for such problems. In this regard, consider the Bagley-Torvik equation with non-homogeneous initial conditions ([98], [100])

$$AD^2 y(t) + BD^{3/2} y(t) + Cy(t) = C(t+1), \quad 0 < t \le 1 \tag{5.3.6}$$

with initial condition as $y(0) = y'(0) = 1$ It has the exact solution given as

$$y(t) = t + 1 \tag{5.3.7}$$

This problem is work out with the same methodology as used in the previous example. A set of weights for FDE-NN networks trained stochastically by using PSO-PS and reported result of GA-PS algorithms [227] are given in Table 5.34.

**Table 5.34** FDE-NN Networks Weights Trained by Different solvers for Problem II.

| *i* | **Reported Results (GA-PS)** | | | **Our Results(PSO-PS)** | | |
|---|---|---|---|---|---|---|
| | $w_i$ | $\alpha_i$ | $\beta_i$ | $w_i$ | $\alpha_i$ | $\beta_i$ |
| 1 | 0.45094 | -0.65870 | 0.48045 | -0.90928 | 0.23072 | 0.85724 |
| 2 | -0.09680 | -0.56740 | 0.84448 | -0.68412 | 0.46887 | 0.20841 |
| 3 | -0.35440 | 0.51534 | 0.13803 | -0.03747 | -0.54460 | -1.46511 |
| 4 | 1.75884 | -0.14670 | 0.57994 | 1.65704 | 0.16841 | 0.73563 |
| 5 | -4.58000 | 0.14171 | 0.66264 | 0.49229 | 0.69282 | -0.92286 |
| 6 | 0.05750 | 0.49062 | 0.68844 | 0.98725 | 0.00986 | -0.36973 |
| 7 | 1.10702 | 1.50930 | 0.00385 | 0.47866 | 0.71605 | -0.12232 |
| 8 | 1.55824 | -0.75050 | -5.94910 | -1.34877 | -0.40643 | 0.74156 |
| 9 | 0.24057 | 0.22079 | 0.83593 | 2.83351 | -0.12491 | -0.79326 |
| 10 | -0.12630 | -0.20130 | 0.63207 | 0.44834 | -0.55608 | -0.45546 |

The solution of the problem can be obtained using Table 5.34 and equation (3.1.8) for inputs between *0* and *1*. The reported results, PS, GA, GA-PS [227], and the PSO-PS

algorithms, are given in Table 5.35. It can be inferred from the results that the best result are obtained by FDE-NN trained by PSO-PS algorithm.

**Table 5.35** Comparison of Results for the Solution of Problem II.

| T | y(t) | Reported Result | | | | | | Our Result | |
|---|---|---|---|---|---|---|---|---|---|
| | | $\hat{y}(t)$ | | | $\|y(t)-\hat{y}(t)\|$ | | | $\hat{y}(t)$ | $\|y(t)-\hat{y}(t)\|$ |
| | | PS | GA | GA-PS | PS | GA | GA-PS | PSO-PS | PSO-PS |
| 0.0 | 1.0 | 0.691604 | 1.024862 | 1.016007 | 0.30839 | 2.30e-2 | 1.60e-2 | 1.000257 | 2.57e-4 |
| 0.1 | 1.1 | 0.623749 | 1.121206 | 1.104733 | 0.47625 | 2.69e-2 | 4.73e-3 | 1.097252 | 2.75e-3 |
| 0.2 | 1.2 | 0.859697 | 1.220821 | 1.199804 | 0.34030 | 3.13e-2 | 1.95e-4 | 1.194203 | 5.80e-3 |
| 0.3 | 1.3 | 1.122266 | 1.323041 | 1.299333 | 0.17773 | 3.45e-2 | 6.66e-4 | 1.292454 | 7.55e-3 |
| 0.4 | 1.4 | 1.337736 | 1.426952 | 1.401629 | 6.22e-2 | 3.45e-2 | 1.62e-3 | 1.392900 | 7.10e-3 |
| 0.5 | 1.5 | 1.501839 | 1.531330 | 1.504972 | 1.83e-3 | 2.87e-2 | 4.97e-3 | 1.495865 | 4.13e-3 |
| 0.6 | 1.6 | 1.628907 | 1.634569 | 1.607429 | 2.89e-2 | 1.36e-2 | 7.42e-3 | 1.600920 | 9.20e-4 |
| 0.7 | 1.7 | 1.734458 | 1.734591 | 1.706705 | 3.44e-2 | 1.49e-2 | 6.70e-3 | 1.706612 | 6.61e-3 |
| 0.8 | 1.8 | 1.830520 | 1.828738 | 1.799987 | 3.05e-2 | 2.30e-2 | 1.27e-5 | 1.810101 | 1.01e-2 |
| 0.9 | 1.9 | 1.925308 | 1.913640 | 1.883785 | 2.53e-2 | 2.69e-2 | 1.62e-2 | 1.906636 | 6.64e-3 |
| 1.0 | 2.0 | 2.024099 | 1.985057 | 1.953762 | 2.40e-2 | 3.13e-2 | 4.62e-2 | 1.988841 | 1.11e-2 |
| | | | | **Mean** | 1.37e-1 | 2.62e-2 | 9.52e-3 | | 5.72e-3 |

**Table 5.36** Statistical Parameter Based on the Value of Absolute Unsupervised Error.

| Absolute error | T | Statistical Parameters | | | |
|---|---|---|---|---|---|
| | | Best | Worst | Mean | STD |
| $\|y(t)-\hat{y}(t\|$ | 0.0 | 2.37e-06 | 0.173705 | 0.084275 | 0.047378 |
| | 0.2 | 8.55e-04 | 0.140164 | 0.068181 | 0.035941 |
| | 0.4 | 1.65e-03 | 0.117775 | 0.048829 | 0.030167 |
| | 0.6 | 5.58e-05 | 0.115513 | 0.039045 | 0.034439 |
| | 0.8 | 4.56e-04 | 0.150716 | 0.050269 | 0.043599 |
| | 1.0 | 2.26e-04 | 0.224342 | 0.093478 | 0.065664 |
| $\|\acute{y}(t)-\hat{y}'(t\|$ | 0.0 | 1.65e-03 | 0.191944 | 0.082781 | 0.052253 |
| | 0.2 | 1.21e-03 | 0.167551 | 0.082567 | 0.047238 |
| | 0.4 | 3.10e-04 | 0.125426 | 0.072890 | 0.035678 |
| | 0.6 | 6.34e-04 | 0.104915 | 0.051419 | 0.028603 |
| | 0.8 | 1.74e-03 | 0.259299 | 0.128442 | 0.081655 |
| | 1.0 | 0.134624 | 0.487895 | 0.365691 | 0.088140 |

Moreover, the statistical parameters for best 100 out of 125 independents runs are executed for training of weights for FDE-NN networks with PSO-PS algorithm. The values of the absolute error for solution of equation and its derivative for input between 0 and 1 with step size of 0.2 are given in Table 5.36. It can be inferred from that result that our scheme approximates the solution and its derivative closer to the exact solutions.

### 5.3.3 PROBLEM III: BAGLEY-TORVIK EQUATION WITH UNKNOWN EXACT SOLUTION

In this example, the fractional order system based on Bagley-Torvik equation is taken for which the exact solution is not available. However, its numerical solutions are obtained by different state of the art solvers.

Consider the fractional order system [30], [102]

$$D^2 y(t) + \frac{1}{2} D^{3/2} y(t) + \frac{1}{2} y(t) = g(t), \quad t > 0 \tag{5.3.8}$$

where the function g(t) and initial conditions are given as

$$g(t) = \begin{cases} 8, & (0 \le t \le 1) \\ 0, & (t > 1) \end{cases}$$
$$y(0) = y'(0) = 0 \tag{5.3.9}$$

Set of unknown weights of FDE-NN networks trained stochastically using PSO-PS algorithm and reported results for GA-PS algorithms are given in Table 5.37. The results obtained based on these weights are summarized in Table 5.38. It also includes the results of PMA algorithm. The results shows that our scheme, PSO-PS, performs better than other due to its computational superiority over GA in term of fast convergence as reported in [190], [218]. Moreover, PMA takes a small step size of 0.01while our scheme use step size of 0.2 and still gives comparable results.

**Table 5.37** FDE-NN Networks Weights Trained by Different Solvers for Problem III.

| *i* | Reported Results (GA-PS) | | | Our Results(PSO-PS) | | |
|---|---|---|---|---|---|---|
| | $w_i$ | $\alpha_i$ | $\beta_i$ | $w_i$ | $\alpha_i$ | $\beta_i$ |
| 1 | 1.17510 | 0.35237 | -0.58370 | -0.90928 | 0.23072 | 0.85724 |
| 2 | 1.96535 | 0.29495 | 0.33525 | -0.68412 | 0.46887 | 0.20841 |
| 3 | -0.59990 | 0.16328 | 1.22993 | -0.03747 | -0.54460 | -1.46511 |
| 4 | 0.55992 | 0.18356 | 1.06363 | 1.65704 | 0.16841 | 0.73563 |
| 5 | -9.66370 | -3.21670 | 2.52132 | 0.49229 | 0.69282 | -0.92286 |
| 6 | 1.28638 | 0.26612 | 0.53455 | 0.98725 | 0.00986 | -0.36973 |
| 7 | -0.13130 | 0.08985 | -0.48310 | 0.47866 | 0.71605 | -0.12232 |
| 8 | 0.98771 | 0.20273 | 0.60954 | -1.34877 | -0.40643 | 0.74156 |
| 9 | -4.21050 | 0.20278 | -0.58700 | 2.83351 | -0.12491 | -0.79326 |
| 10 | 0.49425 | -0.55250 | 1.35857 | 0.44834 | -0.55608 | -0.45546 |

**Table 5.38** Comparison of Results for the Solution of Problem III.

| T | Reported results | | | | Our Result |
|---|---|---|---|---|---|
| | *PMA* | *PS* | *GA* | *GA-PS* | *PSO-PS* |
| 0.0 | 0.000000 | 0.026428 | 0.095404 | 0.011779 | 0.004812 |
| 0.1 | 0.032734 | 0.191048 | 0.176205 | 0.172272 | 0.103582 |
| 0.2 | 0.133300 | 0.260819 | 0.282477 | 0.298043 | 0.278269 |
| 0.3 | 0.296654 | 0.317447 | 0.419234 | 0.427562 | 0.321835 |
| 0.4 | 0.519024 | 0.405184 | 0.592514 | 0.587411 | 0.593562 |
| 0.5 | 0.797107 | 0.537824 | 0.809598 | 0.791994 | 0.843210 |
| 0.6 | 1.127822 | 0.724469 | 1.079262 | 1.052421 | 1.253451 |
| 0.7 | 1.508199 | 0.983967 | 1.412097 | 1.380009 | 1.592648 |
| 0.8 | 1.935317 | 1.360854 | 1.820877 | 1.787807 | 1.993762 |
| 0.9 | 2.406276 | 1.954990 | 2.321026 | 2.291464 | 2.464897 |
| 1.0 | 2.918175 | 2.984840 | 2.931170 | 2.909908 | 2.945782 |

### 5.3.4 CONCLUSION

On the basis of the simulations and analysis it can be concluded that both linear and non-linear fractional order system based on Bagley-Torvik equations can be solved by proposed heuristic computational intelligence algorithm. The fractional differential equation neural networks of the equation trained by PSO-PS algorithm is the better stochastic optimizer than that of PS, GA, GA-PS algorithms. On the basis of the

statistical analysis, it can be inferred that our proposed computing approach is reliable, effective and easily applicable to such complex fractional order systems. In our future work, our intention is to use other biologically inspired computational intelligence algorithms to solve these fractional order systems.

## 5.4 SUMMARY OF CHAPTER

This chapter presents the application of the design approach to solve fractional differential equations with the help of swarm intelligence optimized neural networks. The comparative studies with standard numerical techniques, as well as, with different stochatics solvers are also presented. The chapter starts with application of the scheme to generic form of fractional order system associated with linear and non-linear fractional differential equations. The designed methodology also tested for Riccati and Bagley-Torvik fractional systems on the same problems as solved in the last chapters. The accuracy, reliability, convergence, computational complexity and effectiveness of different stochastic numerical solvers have been described based on large numbers of independents runs and their statistical analysis.

# CHAPTER 6

# CONCLUSION

## 6.1 SUMMARY OF RESULTS

In this section, the conclusions are presented on the basis of the simulation and results given in the last three chapters.

A successful design of mathematical model for differential equation of fractional order has been given using the strength of artificial neural network defines by taking exponential function as an activation function. The linear combination of neural networks is defining an unsupervised error for the equation. Optimization problem of highly stochastic in nature is formulated for finding the weights for networks. The accuracy of proposed scheme is dependent on availability of weights for which the value of unsupervised error is minimized.

The evolutionary computational techniques are applied successfully to this difficult optimization problem. In the scheme, the effective global search is carried out using Genetic Algorithm hybridized with efficient local search techniques like simulating annealing, active-set algorithm, and pattern search etc.

The designed methodology has been tested successfully for a number of linear and nonlinear differential equations of fractional order. Comparison of the results is made with available exact solutions as well as with numerical solvers like ADM, DAM, and PMA etc. It is found that the error obtained in the range of $10^{-02}$ to $10^{-04}$ in case of linear FDE equations. However, it is decreased slightly in case of solving complex nonlinear fractional differential equations.

Moreover, the solution for nonlinear Riccati differential equation of arbitrary order has also been obtained successfully by the scheme. It is further added that the solution to fractional order system, known as Bagley-Torvik equation, is a special example for

successful applications of the designed scheme. The reliability of the algorithm is analyzed with comprehensive statistical analysis of the results. It is found that FDE-NN networks optimized with GA-PS is best stochastic solver as compared with GA and PS techniques.

Swarm intelligence is also used successfully as an alternate to evolutionary algorithm learning of weights of FDE-NN networks and it has produced comparatively superior results than that of other solvers.

The method has been tested successfully by applying it to different linear and non-linear ordinary differential equations of fractional order. A large number of Monte Carlo simulations with stochastic solvers validated its reliability and effectiveness through statistical analysis. It is found that the best results are achieved for FDE-NN networks optimized with PSO-SA algorithm instead of SA, GA, GA-SA, PSO algorithms. In addition to that, it is observed that PSO-SA, for less than half the population as compared to GA-SA, gives same accuracy.

The accuracy of the proposed scheme is comparable with standard state of the art deterministic numerical solvers. It is found that our solution is in agreement with these techniques or even better in some cases. The strength of the proposed scheme over such solvers is that it can provide the result on continuous finite time domain, instead of predefined discrete grid of points.

The nonlinear system based on Riccati differential equation of arbitrary order is also solved successfully by this scheme. It is found again that the FDE-NN networks optimized with PSO-SA Hybrid approach is the best solver compared to SA, GA, GA-SA PSO algorithms.

A linear and non-linear fractional order system based on Bagley-Torvik equations are solved successfully by proposed heuristic computational intelligence algorithm based on PSO hybridized with PS algorithm. It is found that PSO-PS technique is better stochastic optimizer than PS, GA, and GA-PS algorithm.

## 6.2 DIRECTIONS OF FUTURE WORK

- One can look for other basis function in fractional neural network modeling with a better capability of universal function approximation than that of exponential function and availability of its fractional derivative in term of computable mathematical terms.
- Optimization with smaller step size, more number of neuron and training for larger interval will improve the strength of FDE-NN networks for approximate modeling of the equation, but at the cost of computational complexity. However, in future with the availability of efficient computing equipment it can be tested easily.
- Time of computation of the designed approaches can also be decreased with the use of more sophisticated hardware platforms like grid, parallel, distributed processing techniques.
- Training of weights can also be investigated with other hybrid stochastic optimization techniques like genetic programming, ant and bee colony optimization etc and compare the results with the given algorithm.
- The designed scheme can also be applied on systems of ordinary linear and nonlinear differential equation of fractional order and a practical example of fractional-order Chua-hartley System is taken to investigate the strength and weakness of the approach.
- The fractional heat, wave and diffusion equation are used extensively in fractional orders system in engineering and their models can be made by expanding our scheme to be applicable to partial differential equations of fractional order.

## REFERENCES


[1]. K. S. Miller and B. Ross, "An introduction to the fractional calculus and fractional differential equations," A Wiley-Interscience Publication, John Wiley & Sons Inc., New York, 1993.

[2]. S. Samko, A. Kilbas, and O. Marichev, "Fractional Integrals and derivatives: Theory and Applications," Gordon and Breach Science Publishers, Yverdon, 1993.

[3]. A. K. Anatoly, H. M. Srivastava, and J. J. Trujillo.: Theory and application of fractional differential equations. North-Holland Mathematics Studies 204, Elsevier (2006).

[4]. B. Ross, "Origins of fractional calculus and some applications," Internat. J. Math.Statist. Sci., vol. 1, no. 1, pp. 21-34, 1992.

[5]. B. Ross, "The development of fractional calculus 1695.1900," Historia Math., vol. 4, pp. 75-89, 1977.

[6]. B. Ross, "Fractional calculus and its applications," proceedings of the international conference held at the University of New Haven, vol. 457, Lecture notes in mathematics, Springer Verlag, Berlin, 1975.

[7]. J. T. Machado, V. Kiryakova, and F. Mainardi, "Recent history of fractional calculus," Communication in Nonlinear Science and Numerical Simulation, vol. 16, no. 3, pp. 1140-1153, 2011.

[8]. A. R. Hall, "*Philosophers at war*. The quarrel between Newton and Leibniz," Cambridge University Press, Cambridge, 1980.

[9]. G. H. Pertz and C. J. Gerhardt, "Briefwechsel zwischen Leibniz, Hugens van Zulichem und dem Marquis de l'Hospital," editors, Leibnizens gesammelte Werke, Lebinizens mathematische Schriften, Erste Abtheilung, Band II, pp. 301-302, Dritte Folge Mathematik (Erster Band), A. Asher & Comp., 1849.

[10]. G. H. Pertz and C. J. Gerhardt, "Briefwechsel zwischen Leibniz, Jacob Bernoulli, Joh. Bernoulli und Nic. Bernoulli," editors *Leibnizens gesammelte Werke, Lebinizens mathematische Schriften, Erste Abtheilung, Band III*, page 225, Dritte Folge Mathematik (Erster Band), A. Asher & Comp., 1850.

[11]. G. H. Pertz and C. J. Gerhardt, "Briefwechsel zwischen Leibniz,Wallis, Varignon, Guido Grandi, Zendrini, Hermann und Freiherrn von Tschirnhaus," editors. Leibnizens gesammelte Werke, Lebinizens mathematische Schriften, Erste Abtheilung, Band IV, page 25, Dritte Folge Mathematik (Erster Band), A. Asher& Comp., 1859.

[12]. L. Euler, "De progressionibus transcendendibus seu quarum termini generals algebraicae dari nequent," Comm. Acad. Sci., Petropolitanae, vol. 5 pp. 36.57,

1738. Translated to english by S. G. Langton, University of San Diego, www.sandiego.edu/_langton.

[13]. J. L. Lagrange, “Sur une: Nouvelle espéce de calcul relatif. A la diff´erentiation et a l’intégration des quantités variables,” Gauthier-Villars, Paris, 1849.

[14]. P. S. Laplace. Théorie analytique des probabilités, pages 85.86+156. Imprimeur-Libraire pour les Mathématiques, Paris, 1820.

[15]. J. B. J. Fourier, “Théorie analytique de la chaleur,” pp. 561-562, Didot, Paris, 1822.

[16]. J. Liouville, “Mémoire sur le changement de la variable indépendante dans le calcul des différentielles indices quelconques,” J. l’Ecole Roy. Polytéchn., vol. 15, no. 24, pp. 17-54, 1835.

[17]. J. Liouville, “Mémoire sur l’usage que l’on peut faire de la formule de fourier, dan le calcul des diff´erentielles _a indices quelconques” J. für reine und angew. Math., vol 13, no. (1-3), pp. 219-232, 1835.

[18]. J. Liouville, “Mémoire sur l’intégration des équations différentielles à indices fractionnaires,” J. l’Ecole Roy. Polytéchn., vol. 15, no. 55, pp. 58-84, 1837.

[19]. A. Cayley. Note on Riemann's paper .Versuch einer allgemeinen Auflosung der Integration und Differentiation.. Math. Ann., vol.16 pp. 81-82, 1880.

[20]. H. Weyl, “Bemerkungen zum Begriff des Differentialquotienten gebrochener Ordnung,” *Vierteljshr. Naturforsch. Gesellsch. Zürich*, vol. 62, pp. 296-302, 1917.

[21]. A. Marchaud, “Sur les dérivées et sur les différences des fonctions des variablesréelles,” J. math. pures et appl., vol. 6, no. 4, pp. 371-382, 1927.

[22]. Y. Watanabe. “Notes on the generalized derivative of Riemann-Liouville and its application to Leibnitz's formula, I and II,” Tòhoku Math. J., vol. 34, pp. 8-41, 1931.

[23]. M. Riesz, “Potentiels de divers ordres et leurs fonctions de Green” *C. R. Congérs* Intern. Math, vol. 2, pp. 62-63, 1936.

[24]. M. Riesz, “L’intégrales de Riemann-Liouville et potentiels,” Acta Litt. Acad. Sci. *Szeged*, vol. 9, pp. 1-42, 1938.

[25]. B. S. Nagy, “Űber gewisse Extremalfragen bei transformierten trigonometrischen Entwicklungen, I: Periodischer Fall. *Ber. Verhandl. Sächsich. Akad. Wiss. Leipzig,*” Math. Phys. Kl., vol. 90, pp. 103-134, 1938.

[26]. A. Erdélyi, “On fractional integration and its application on the theory of Hankel transforms,” Quart. J. Math., Oxford ser., vol.11, no. 44, pp. 293-303, 1940.

[27]. A. Erdélyi and H. Kober, “Some remarks on Hankel transforms,” Quart. J. Math., *Oxford ser.*, vol.11 no. 43, pp. 212-221, 1940.

[28]. B. Ross, “Fractional calculus and its applications,” Proceedings of the international conference held at the University of New Haven, vol. 457 Lecture notes in mathematics. Springer Verlag, Berlin, 1975.

[29]. K. B. Oldham and J. Spanier, “*The fractional calculus,”* Academic Press [A subsidiary of Harcourt Brace Jovanovich, Publishers], New York-London, 1974.

[30]. I. Podlubny, “Fractional differential equations,” vol. 198, Mathematics in Science and Engineering, Academic Press Inc., San Diego, CA, 1999.

[31]. M. T. Barlow, “Diffusions on fractals, in: Lectures on Probability Theory and Statistics,” vol. 1690, Lect. Notes in Math., Springer-Verlag, Berlin, pp. 1-121, 1995.

[32]. D. Ben-Avraham, and S. Havlin, “Diffusion and Reactions in Fractals and Disordered Systems,” Cambridge University Press, Cambridge, 2000.

[33]. A. Carpinteri, and F. Mainardi, (Eds.), “Fractals and Fractional Calculus in Continuum Mechanics,” vol. 378, CISM Courses and Lectures, Springer-Verlag, Berlin, 1997.

[34]. R. Metzler, and J. KLAFTER, The random walk's guide to anomalous diffusion: A fractional dynamics approach, Phys. Rep., 339(1), (2000) .

[35]. H. M. Ozaktas, Z. Zalevsky, and M. A. Kutay, “The Fractional Fourier Transform with Applications in Optics and Signal Processing,” John Wiley and Sons, New York, 2000.

[36]. C. A. Manje, Y. Q. Chen, B. M. Vinagre, D. Xue, and V. Feliu, “Fractional-order System and Control: Fundamentals and Applications,” Spinger-Verlag London, Limited, 2010

[37]. B. J. West, M. Bologna, and P. Grigolini, “Physics of Fractal Operators,” Springer-Verlag, New York and Berlin, 2003.

[38]. Edited by M. D. Ortigueira and J. A. T. Machado, “Special issue on fractional signal processing and applications” j. Signal Processing, vol. 83, no. 11, pp. 2285-2480, 2003.

[39]. Edited by M. D. Ortigueira and J. A. T. Machado, “Special section: Fractional calculus applications in signal and systems,” j. Signal Processing, vol. 86, no. 10, pp. 2503-3094, 2006.

[40]. Edited by L. Fawang, M. M. Meerschaert, S. Momani, N. N. Leonenko, W. Chen and O. P. Agrawal, “Special Issue on Fractional Differential Equations,” Inter. J. Diff. Eqs. vol. 2010, 2010.

[41]. A. Erdélyi (ed.)**, "**Higher Transcendental Functions," vol. 1, McGraw-Hill, New York, 1955.

[42]. G. M. Mittag-Leffler, "Sur la nouvelle function $E_{\alpha}(x)$," C. R. Acad. Sci. Paris, vol. 137, pp. 554-558, 1903.

[43]. A. Wiman, "Űber die nulstellen der funktionen $E_{\alpha}(x)$," Acta Math., vol. 29 pp. 217-234, 1905.

[44]. R. P. Agarwal, "A propos d'une note de M. Pierre Humbert," C. R. Séances Acad. Sci., vol. 236, no. 21, pp. 2031-2032, 1953.

[45]. M. Caputo, "Linear model of dissipation whose Q is almost frequency independent," II, Geophys, J. R. Astr. Soc., vol. 13, pp. 529-539, 1967.

[46]. M. Caputo, "Elasticità e Dissipazione," Zabuchelli, Bologna, 1969.

[47]. S. G. Samko, A. A. Kilbas and O. I. Marithchev, "Integrals and derivatives of the fractional order and some of their application," Nauka I Tekhnika, Minsk, 1987 (in Russian).

[48]. P. L. Butzer, A. A. Kilbas, J. J. Trujillo, "Compositions of Hadamard-type fractional integration operators and the semigroup property," J. Math. Anal. Appl., vol. 269, no. 2, pp. 387-400, 2002.

[49]. J. Hadamard, "Essai sur létude des fonctions données par leur développement de Taylor," J. Math. Pures Appl., vol. 8, ser. 4, pp. 101-186, 1892.

[50]. A. A. Kilbas, J. J. Trujillo, "Differential equation of fractional order: methods, results and problems," Appl. Anal., vol. 78, no. (1-2), pp. 153-192, 2001.

[51]. A. A. Kilbas, J. J. Trujillo, "Differential equation of fractional order: methods, results and problems II," Appl. Anal., vol. 81, no. 2, pp. 435-493, 2002.

[52]. A. A. Kilbas, J. J. Trujillo, "On solutions of a homogeneous fractional differential equations with nonconstant coefficients (Preprint)," Depto. Anàlisis Matemàtico (Universidad de La Laguna), 2005.

[53]. S. T, Tu, D. K. Chyan and H. M. Srivastava, "Some families of ordinary and partial fractional differintegral equations," Integral Transform Spec. Funct., vol. 11, pp. 291-302, 2001.

[54]. H. M. Ozaktas, M. A. Kutay, G, Bozdagi, "Digital computation of the fractional Fourier transform," IEEE Trans. Sig. Proc., vol. 44, pp. 2141-2150, 1996.

[55]. S. C. Pei, M. H. Yeh, and C. C. Tseng, "Discrete fractional Fourier-transform base on orthogonal projections," IEEE Trans. Sig. Proc., vol. 47, no. 5, pp. 1335-1348, 1999.

[56]. J. E. Solteiro, J. A. Tenreiro-Machado and P. B. Moura, “Fractional order dynamics in GA planner,” J. Signal Proces., vol. 83, no. 11, pp. 2377-2386, 2003.

[57]. A. Bultheel, H. Martinez-Sulbaran, “Computation of the fractional Fourier transform, Appl. Comp. Harm. Anal., vol. 16, no. 3, pp. 182-202, 2004.

[58]. L. B. Eldred, W. P. Baker and A. N. Palazotto, “Numerical application of fractional derivative model constitutive relations for viscoelastic meterials,” Computers Struct., vol. 60, pp. 875-882, 1996.

[59]. F. Bella, P. F. Biagi, M. Caputo, G. Della Monica, A. Ermini, P. Manjgaladze, V. Sgrigna and D. Zilpiniani, “Very slow moving crustal strain disturbances,” Tectonophysics, vol. 179, pp. 131-139, 1990.

[60]. M. Caputo, “Linear model of dissipation whose Q is almost frequency independent II,” Geophys, J. R. Astr. Soc., vol 13, pp. 529-539, 1967.

[61]. M. Caputo, “The rheology of an anelastic medium studied by mean of the observation of the splitting of its eigenfrequencies,” J. Acoust. Soc. Am., vol. 86, no. 5 pp. 1984-1987.

[62]. M. Caputo, “The splitting of the free oscillations of the Earth caused by rheology,” Rend. Fis. Acc. Lineri, vol. 1, no. 9, pp. 119-125, 1990.

[63]. A. Fenander, “Modal synthesis when modeling damping by use of fractional derivatives, AIAA journal, vol. 34, no. 5, pp. 1051-1058, 1998.

[64]. Ch. Friedrich, “Linear viscoelastic behavior of branched plybutadiene: a fractional calculus approach,” Acta Polymer, vol. 46, pp. 385-390.

[65]. Ch. Friedrich and H. Braun, “Linear viscoelastic behavior of complex polymeric materials” a fractional mode representation,” Colloid and Polymer Science, vol. 272, pp. 1536-1546, 1994.

[66]. B. Gross, “On creep and relaxation,” J. Appl. Phys., vol. 18, pp. 212-221, 1947.

[67]. F. Mainardi, “Fractional relaxation in anelastic solids,” J. Alloys and Compounds, vol. 211/212, pp. 534-538, 1994.

[68]. F. Mainardi, The fundamental solutions for the fractional diffusion-wave equation, Appl. Math. Lett., vol. 9, no. 6, pp. 23-28, 1996.

[69]. R. L. Bagley, “Power law and fractional calculus model of viscoelasticity, AIAA Journal, vol. 27, no. 10, pp. 1412-1417, 1989.

[70]. R. L. Bagley and P. J. Torvik, “Fractional calculus – a different approach to analysis of viscoelastically damped structures,” AIAA Journal, vol. 21, no. 5, Rpp. 741-748, 1983.

[71]. R. L. Bagley and P. J. Torvik, "Fractional calculus in the transient analysis of viscoelastically damped structures," AIAA Journal, vol. 23, no. 6, pp. 918-925, 1985.

[72]. H. W. Bode, "Network analysis and feedback amplifier design," Tung Hwa Book Company, Shanghai, China, 1949.

[73]. Haykin S. "Adaptive filter theory" 4th Ed. Prentice Hall, New Jersey, 2002.

[74]. R. M. A. Zahoor, J. A. Khan, and I. M. Qureshi, "A Modified Least Mean Square Algorithm Using Fractional Derivative and its Application to System Identification" *EJSR*, vol. 35, no. 1, pp. 14-21, 2009.

[75]. W. T. Reid, "Riccati differential equations (Mathematics in science and engineering)," vol. 86, Academic Press, New York, 1972.

[76]. B. D. Anderson, J. B. Moore, "Optimal control-linear quadratic methods," Prentice-Hall, New Jersey, 1999.

[77]. S. Bittanti, "History and prehistory of the Riccati equation" Proc. 35th IEEE Conf. on Decision and Control, Kobe, Japan, pp. 1599-1604, 1996.

[78]. H. H. Goldstine, "A History of the Calculus of Variations from the 17th through the 19th Century" Springer-Verlag, 1980.

[79]. S. K. Mitter, "Filtering and Stochastic Control: a Historical Perspective" Control Systems IEEE Magazine, vol. 16, no. 3 pp. 67-76, 1996.

[80]. Kravchenko, and V. Vladislav, "Applied Pseudoanalytic Function Theory, ch. 6 Complex Riccati equation, pp. 65-72 frontiers in mathematics," Brikhauser, Basel 2009.

[81]. R. Conte, and M. Musette, "Link between solitary waves and projective Riccati equations," J. Phys. A. Math. Gen., vol. 25, pp. 5609-5623, 1992.

[82]. J. L. Wu, G. H. Chen, "A new operational approach for solving Fractional Calculus and Fractional differential equations numerically," Proc., vol. 397, Software Engineering and Applications, 2003.

[83]. K. Diethelm, J. M. Ford, N. J. Ford, and W. Weilbeer, "Pitfalls in fast numerical solvers for fractional differential equations," J. Comput. Appl. Math., vol. 186, pp. 482-503, 2006.

[84]. F. Mainardi, G. Pagnini, and R. Gorenflo, "Some aspects of fractional diffusion equations of single and distributed orders," Appl. Math. Comput., vol. 187, no. 1, pp. 295-305, 2007.

[85]. F. Geng, Y. Lin, and M. Cui, "A piecewise variational iteration method for Riccati differential equations," Comp. math. appl., vol. 58, pp. 2518-2522, 2009.

[86]. C. Li, and Y. Wang, " Numerical algorithm based on Adomian decomposition for fractional differential equations," Comp. Math. appl., vol. 57, pp. 1672-1681, 2009.

[87]. S. Abbasbandy, "A new application of He's variational iteration method for quadratic Riccati differential equation by using Adomian's polynomials," J. Comput. Appl. Math., vol. 207, no. 1, pp. 59-63, 2007.

[88]. L. Galeone, and R. Garrappa, "Fractional Adams-Moulton methods," J. Math. Comp. Simul., vol.79, no. 4, pp. 1358-1367, 2008.

[89]. R. L. Bagley, and P. J. Torvik, "On the appearance of the fractional derivative in the behavior of real materials," ASME J. Appl. Mech., vol. 5, no. 2, pp. 294-298, 1984.

[90]. S. S. Ray, and R. K. Bera, "Analytical solution of the Bagley–Torvik equation by Adomian decomposition method," Appl. Math. Comput., vol. 168, no. 1, pp. 398-410, 2005.

[91]. S. Manabe, "A suggestion of fractional-order controller for flexible spacecraft attitude control," Nonlinear Dyn., vol. 29, no. 1-4, pp. 251-268, 2004.

[92]. S. Das, "Solution of Extraordinary Differential Equations with Physical Reasoning by Obtaining Modal Reaction Series," Modelling and Simulation in Engineering, vol. 2010, Article ID 739675, 19 pages, 2010.

[93]. R. P. Agarwal, M. Belmekki, and M. Benchohra, "A survey of semilinear differential equations and inclusions involving Riemann-Liouville fractional derivative," Advances in difference equations, vol. 2009, Article ID 981728, 47 pages 2009.

[94]. Katica (Stevanović) Hedrih, "The transversal creeping vibrations of a fractional derivative order constitutive relation of nonhomogeneous beam," Mathematical Problems in Engineering, vol. 2006, Article ID 46236, 18 pages, 2006.

[95]. Y. Tian and A. Chen, "The Existence of Positive Solution to Three-Point Singular Boundary Value Problem of Fractional Differential Equation," Abstract and Applied Analysis, vol. 2009, Article ID 314656, 18 pages, 2009.

[96]. M. Naber, "Linear fractionally damped oscillator," International Journal of Differential Equations, vol. 2010, Article ID 197020, 12 pages, 2010.

[97]. Y. Hu, Y. Luo, Z. Lu, "Analytical solution of the linear fractional differential equation by Adomian decomposition method," Journal of Computational and Applied Mathematics, vol. 215, pp. 220–229, 2008.

[98]. A. M. A. El-Sayed et al., "Analytical and numerical solutions of multi-term nonlinear fractional orders differential equations," Applied Numerical Mathematics, vol. 60 pp. 788-797, 2010.

[99]. A. Ghorbani, and A. Alavi**, "**Application of He's variational iteration method to solve semidifferential equations of nth Order," Mathematical Problems in Engineering, vol. 2008, Article ID 627983, 9 pages, 2008. DOI:10.1155/2008/627983

[100]. Y. Cenesiz, Y. Keskin, and A. Kurnaz, "The solution of the Bagley-Torvik equation with generalized Taylor collocation method," Journal of the Franklin Institute, vol. 347, pp. 452-466, 2010.

[101]. K. Diethelm and J. Ford. "Numerical solution of the Bagley-Torvik Equation," BIT Numer. Math. vol. 42, no. 3 pp. 490-507, 2002.

[102]. I. Podlubny, T. Skovranek, and V. B. M. Jara, "Matrix approach to discretization of fractional derivatives and to solution of fractional differential equations and their systems," IEEE Conf. ETFA 2009, pp. 1-6, 2009.

[103]. T. T. Hartley and F. Mossayebi, "Control of Chua's circuit," J. Circuits Syst. Comput., vol. 3, no. 1, pp. 173-194, 1993.

[104]. T. T. Hartley, C. F. Lorenzo and H. K. Qammer, "Choas in fractional order Chua's system," IEEE Trans. On Circuits and Systems- I: Fundamental Theory and Applications, vol. 42, no. 8, pp. 485-490, 1995.

[105]. B. Mandelbrot, "The fractal geometry of nature," Freeman, San Francisco, 1982.

[106]. R. Metzler, W. G. Glöckle, and T. F. Nonnenmacher, "Fractional model equation for anomalous diffusion," Physica A, vol. 211, pp. 13-24, 1994.

[107]. R. R. Nigmatullin, "To the theoretical explanation of the 'universal response'," Phys. Sta. Sol. (b), vol. 123, pp. 739-745, 1984.

[108]. R. R. Nigmatullin, "The realization of the generalized transfer equation in a medium with fractal geometry," Phys. Sta. Sol. (b), vol. 133, pp. 425-430, 1986.

[109]. R. Gorenflo and R. Rutman, "On ultraslow and on intermediate processes," in P. Rusev, I. Dimavski and V. Kiryakova (eds.), Transform method and special functions, SCT Publishers, Singapore, 1995.

[110]. F. Mainardi, "Fractional relaxation-oscillation and fractional diffusion-wave phenomena," Chaos Solitons and Fractals, vol. 7, pp. 1461-1477, 1996.

[111]. F. Mainardi, "The time fractional diffusion-wave equation," Radiofizika, vol. 38, pp. 20-36, 1995.

[112]. A. N. Kochubei, "Fractional order diffusion," J. Diff. Equations, vol. 26, pp. 485-492, 1990, (English translation form Russian).

[113]. A. M. A. El-Sayed, "Fractional order diffusion-wave equation," Int. J. of Theor. Phys., vol. 35, pp. 311-322, 1996.

[114]. J. A. T. Machado, “A probabilistic interpretation on the fractional-order differentiation,” J. of Fractional Calculus & Applied Analysis, vol. 6, no. 1, pp. 73-80, 2003.

[115]. J. A. T. Machado, and A. M. S. Galhano, “Statistical fractional dynamics,” ASME J. Comput. Nonlinear Dyn., vol. 3, no. 2 pp. 021201-1 – 021201-5, 2008.

[116]. N. M. F. Ferreira, and J. A. T. Machado, “Fractional control of coordinated manipulators,” J. of Advanced Comput. Intelligence and Intelligent Informatics, Fuji Technology Press, pp. 1072-1078, vol. 11, No. 9, 2007.

[117]. M. D. G. Marcos, F. B. M. Duarte, and J. A. T. Machado, “Fractional dynamics in the trajectory control of redundant manipulators,” Comm. in Nonlinear Sci. Numerical Simulations, Elsevier, vol. 13, pp. 1836–1844, 2008.

[118]. M. F. Silva, J. A. T. Machado, and R. S. Barbosa, “Using fractional derivatives for the joint control of hexapod robots,” Journal of Vibration and Control, Sage Publishing, vol. 14: pp. 1473-1485, 2008.

[119]. G. E. Carlson and C. A. Halijak, “Approximation of fixed impedance,” IRE Trans. on Circuit Theory, vol. CT-9, no. 3, pp. 302-302, 1962.

[120]. G. E. Carlson and C. A. Halijak, “Approximation of fractional capacitors $(1/s)^{1/n}$ by a regular Newton process,” IRE Trans. on Circuit Theory, vol. CT-11, no. 2, pp. 210-213, 1962.

[121]. R. S. Barbosa, and J. A. T. Machado, “Describing function analysis of systems with impacts and backlash,” Nonlinear Dynamics, Kluwer, USA, vol. 29, n. 1-4, pp. 235-250, 2002.

[122]. N. M. F. Ferreira, and F. B. Duarte, M. F. M. Lima, M. G. Marcos, and J. A. T. Machado, “Application of fractional calculus in the dynamical analysis and control of mechanical manipulators,” Journal of Fractional Calculus & Applied Analysis, vol. 11, no 1, pp. 91-113, 2008.

[123]. F. Duarte, J. A. T. Machado, “Describing function of two masses with backlash,” accepted for publication, Nonlinear Dynamics, Springer.

[124]. K. B. Oldham, “A signal-independent electroanalytical method,” Anal. Chem., vol. 44, no. 1, pp. 196-198, 1972.

[125]. K. B. Oldham, “Interrelation of current and concentration at electrodes,” J. Appl. Electrochem., vol. 21, pp. 1068-1072, 1991.

[126]. F. Riewe, “Mechanics with fractional derivatives,” Phys. Rev. E, vol. 55, no. 3, pp. 3581-3592, 1997.

[127]. F. Riewe, “Nonconservative Lagrangian and Hamiltonian mechanics,” Phys. Rev. E, vol. 53, no. 2, pp. 1890-1899, 1996.

[128]. J. A. T. Machado, and Isabel S. Jesus, “A suggestion from the past?,” Journal of Fractional Calculus & Applied Analysis, vol. 7, n. 4, pp. 403-407, 2004.

[129]. J. A. T. Machado, I. S. Jesus, and A. Galhano, “Fractional order electromagnetics,” Signal Processing, Elsevier, vol. 86, Issue 10, pp. 2637-2644, 2006.

[130]. R. S. Barbosa, J. A. T. Machado, and I. M. Ferreira, “Controller tuning based on Bode's ideal transfer function,” Nonlinear Dynamics, Kluwer, vol. 38, n. 1-4, pp. 305-321, 2004.

[131]. R. S. Barbosa, J.A. T. Machado, and A. M. Galhano, “Performance of fractional PID algorithms controlling nonlinear systems with saturation and backlash phenomena,” Journal of Vibration and Control, , vol. 13, no. 9-10, pp. 1407-1418, 2007.

[132]. R. S. Barbosa, J. A. T. Machado, M. F. Silva, “Discretization of complex-order algorithms for control applications,” Journal of Vibration and Control, Sage Publishing, vol. 14: pp. 1349 – 1361, 2008.

[133]. N. Engheta, “On fractional calculus and fractional multipoles in electromagnetism,” IEEE Trans. on Antennas and Propagations, vol. 44, no. 4, pp. 554-566, 1996.

[134]. L. Figueiredo, J. A. T. Machado, and J. R. Ferreira, “Dynamical analysis of freeway traffic,” IEEE Transactions on Intelligent Transportation Systems, vol. 5, no. 4, pp. 259-266, 2004.

[135]. L. Figueiredo, and J. A. T. Machado, “Simulation and Dynamics of Freeway Traffic,” Nonlinear Dynamics, pp. 567-577, vol. 49, no. 4, 2007.

[136]. R. S. Barbosa, J.A. T. Machado, B.M. Vinagre, and A.J. Calderón, “Analysis of the Van der Pol oscillator containing derivatives of fractional order,” Journal of Vibration and Control, vol. 13, no. 9-10, pp. 1291-1301, 2007.

[137]. I. S. Jesus, and J. A. T. Machado, “Application of fractional calculus in the control of heat systems,” Journal of Advanced Computational Intelligence and Intelligent Informatics, Fuji Technology Press, vol. 11, no. 9, pp. 1086-1091, 2007.

[138]. I. S. Jesus, J. A. T. Machado, “Fractional control of heat diffusion systems,” Nonlinear Dynamics, vol. 54, No. 3, pp. 563-282, 2008.

[139]. I. S. Jesus, J. A. T. Machado and J. B. Cunha, “Fractional electrical impedances in botanical elements,” Journal of Vibration and Control, Sage Publishing, Sept. 2008; vol. 14, pp. 1389-1402, 2008.

[140]. I. S. Jesus, and J. A. T. Machado, “Development of fractional order capacitors based on electrolyte processes,” accepted for publication, Nonlinear Dynamics, Springer.

[141]. R. T. Baillie and M. L. King, “Fractional differencing and long memory processes,” J. Econometrics, vol. 73, no. 1, pp. 1-3, 1996.

[142]. F. Mainardi, M. Raberto, R. Gorenlo, R., and E. Scalas, “Fractional calculus and continuous time finance II: the waiting-time distribution,” Physica A, vol. 278, no. 3-4, pp.468-481, 2000.

[143]. H. Osada, “Self-similar diffusion on a class of infinitely ramified fractals,” J. Math. Soc., Japan, vol. 47, no. 4, pp. 591-616, 1995.

[144]. G. M. Zaslavsky, Hamiltonian Chaos and fractional dynamics, Oxford University Press, Oxford, 2005.

[145]. R. Gorenflo, and S. Vessela, “Abel Integral Equations,” Lect. Notes in Math., vol. 1461, Springer-Verlag, Berlin, 1991.

[146]. R. Gorenflo, “Fractional calculus: Some numerical methods, in: A. Carpinteri and F. Mainari (Eds.), Fractal and Fractional Calculus in Continuum Mechanics,” CISM Courses and Lectures, Springer-Verlag, vol. 378, pp. 277-290, 1997.

[147]. M. K. Shkhanukov, “On the convergence of difference scheme for differential equations with a fractional derivative,” Dokl. Akad. Nauk, vol. 348, no. 6 pp. 746-748, 1996.

[148]. L. Blank, “Numerical treatment of differential equations of fractional order,” Nonlinear World, vol. 4, no. 4, pp. 437-440, 2001.

[149]. K. Diethelm, “An algorithm for the numerical solution of differential equations of fractional order,” Electron. Trans. Numer. Anal., vol. 5, pp. 1-6, 1997.

[150]. K. Diethelm, “ Generalized compound quadrature formulae for finite-part integrals,” IMA J. Numer. Anal., vol. 17, no. 3, pp. 479-493, 1997.

[151]. K. Diethelm, J. Neville, A. D. Ford and A. Freed, “Predictor-corrector approach for numerical solution of fractional differential equations in: J. A. Tenreiro-Machado (Ed.), Special issue of fractional order calculus and its applications,” Nonlinear Dynamics, vol. 29, no. 1-4, pp. 03-22, 2002.

[152]. I. Podlubny, “Numerical solution of ordinary fractional differential equations by the fractional difference method,” in Advances in Differential Equations, Gordon and Breach, Amsterdam, pp. 507-515, 1995.

[153]. P. J. Torvik, and R. L. Bagley, “On the appearance of the fractional derivative in the behavior of real materials,” J. Appl. Mech., vol. 51, pp. 294-298, 1984.

[154]. K. Diethelm, and N. J. Ford, “Numerical Solution of the Bagley-Torvik equation,” BIT, vol. 42, no. 3, pp. 490-507, 2002.

[155]. K. Diethelm, N. J. Ford, and A. D. Freed, “Detail error analysis for a fractional Adams method,” Numer. Algorit., vol. 1, no. 36, pp. 31-52, 2004.

[156]. J. T. Eidelman, N. J. Ford, A. C. Simpson, “The numerical solution of linear multi-term fractional differential equation; system of equations,” J. Comp. Math. Appl., vol. 148, no. 2, pp. 401-418, 2002.

[157]. N. J. Ford and A. C. Simpson, “The numerical solution of fractional differential equations: speed versus accuracy,” Numer. Algorit, vol. 26, no. 4, pp. 333-346, 2001.

[158]. C. Lubich, “Fractional linear multistep method for Abel-Volterra integral equations for the second kind,” Math. Comput., vol. 45, no. 172, pp. 463-469, 1985.

[159]. C. Lubich, “Fractional linear multistep methods for Abel-Volterra integral equations for first kind,” IMA J. Numer. Anal., vol. 1, no. 7, pp. 97-106, 1987.

[160]. A. A. Kilbas and S. A. Marzan, “Cauchy problem for differential equations with Caputo derivative (Russian), Dokl. Akad. Nauk, vol. 399, no. 1, pp. 07-11, 2004.

[161]. K. Diethelm, “Fractional Differential Equations, Theory and Numerical Treatment,” TU Braunschweig, Braunschweig, 2003.

[162]. C. Lubich, “On the stability of linear multistep methods for Volterra convolution equations,” IMA J. Numer. Anal., vol. 3, no. 4, pp. 439-465, 1983.

[163]. E. Hairer, C. Lubich, and M. Schlichte, “Fast numerical solution of nonlinear Volterra convolution equations,” SIAM J. Sci. Statist. Comput., vol. 6, no. 3, pp. 532-541, 1985.

[164]. M. Weibeer, “Efficient Numerical Methods for fractional differential equations and their analytical Background,” Ch. 6, PhD Thesis, ISBN: 978-3-89720-846-9, 2005.

[165]. Q. Yu, J. Song, F. Liu, V. Anh, and I. Turner, “A approximate solution for the Rayleigh-Stokes problem for a heated generalized second grade fluid with fractional derivative model using the Adomian decomposition method,” Journal of Algorithms & Computational Technology, vol. 3, no. 4, pp. 553-572, 2009.

[166]. A. M. A. El-Sayed, S. H. Behiry and W. E. Raslan, “Adomian’s decomposition method for solving an intermediate fractional advection–dispersion equation” Computer and Mathematics with applications, vol. 59, no. 5, pp. 1759-1765, 2010.

[167]. Z. Wang, and H. Zhang, "Construct solitary solutions of discrete hybrid equation by adomian decomposition method," Chaos, Solitons and Fractals, vol. 40, no. 2, pp. 676-683, 2009.

[168]. C. Li and C. Tao, "On the fractional Adams method," Computers & Mathematics with Applications, vol. 58, no. 8, pp. 1573-1588, 2009.

[169]. R. Garrappa, "On some explicit Adams multistep methods for fractional differential equations," Computational and Applied Mathematics, vol. 229, no. 2, pp. 392-399, 2009.

[170]. L. P. Aarts and P. V. D. Veer, "Neural Network Method for solving the partial Differential Equations," Neural Processing Letters, vol. 14 pp. 261-271 2001.

[171]. D. R. Rarisi et al, "Solving differential equations with unsupervised neural networks," J. Chemical Engineering and Processing, vol. 42, pp. 715-721, 2003.

[172]. L. G. Tsoulos, D. Gavrilis, and E. Glavas, "Solving differential equations with constructed neural networks," Neurocomputing, vol. 72, no. 10-12, pp. 2385-2391, 2009.

[173]. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," Science, New Series, vol. 220**,** no. 4598, pp. 671–680, 1983.

[174]. J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark functions," IEEE Transactions on Evolutionary Computation, vol. 10, no**.** 6, pp. 646-657, 2006.

[175]. R. Storn and K. Price, "Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces," Journal of Global Optimization, vol. 11, pp**.** 341–359, 1997.

[176]. J. T. Tsai, T. K. Liu and J. H. Chou, "Hybrid Taguchi-genetic algorithm for global numerical optimization," IEEE Trans. on Evoulutionart Computations, vol. 8, pp. 365-377, 2004.

[177]. A. E. Eiben, and J E. Smith, "Introduction of Evolutionary Computing," ISBN 3-540-40184-9, Springer-Verlag Berlin Heidelberg, 2003.

[178]. D. Ashlock, "Evolutionary Computation for Modeling and Optimization," ISBN-10 0-387-22196-4, Springer+Business Media, Inc, New York, 2006.

[179]. D. B. Fogel, "Evolutionary Computation: Towar a new philosophy of machine intelligence," 3rd Edition, John Wiley & Sons, Inc., Hoboken, New Jersey, 2006.

[180]. M. Jalali-Heravi, and A. Kyani, "Application of genetic algorithm-kernel partial least square as a novel nonlinear feature selection method: Activity of

carbonic anhydrase II inhibitors," Europen Journal of Medicinal Chemistry, vol. 42, no. 5, pp. 649-659, 2007.

[181]. K. S. Shin and Y. J. Lee, "A genetic algorithm application in bankruptcy prediction modeling," Expert System with Applications, vol. 23, no. 3 pp. 321-328, 2002.

[182]. T. Yokota, M. Gen, Y. X. Li, Genetic algorithm for non-linear mixed integer programming problems and it's application," Computer & Industrial Engineering, vol. 30, no. 4, pp. 905-917, 1996.

[183]. J. Kennedy and R. Eberhart, "Particle Swarm Optimization," Proc. IEEE International Conference on Neural Networks, Perth, Australia, IEEE Service Center, vol. 4. Piscataway, NJ, pp. 1942-1948, 1995.

[184]. C. A. G. Santos, L. E. M. Pinto, P. K. D. M. M. Freire and S. K. Mishra, "Application of a particle swarm optimization to a physically-based erosion model," Annals of Warsaw Univesity of Life Sciences-SGGW. Land Reclamation, vol. 42, no. 1, pp. 39-49, 2010.

[185]. R. M. Chen, C. L. Wu, C. M. Wang, and S. T. Lo, "Using novel particle swarm optimization scheme to solve resource-constrained scheduling problem in PSPLIB," Expert Systems with Applications, vol. 37, no. 15, pp. 1899-1910, 2010.

[186]. T. L. Lin, S. J. Horng, T. W. Kao, Y. H. Chen, R. S. Run, R. J. Chen, J. L. Lai and I. H. Kuo, "An efficient job-shop scheduling algorithm based on particle swarm optimization," Expert Systems with Applications, vol. 37, no. 3, pp. 2629-2636, 2010.

[187]. C. Blum, and D. Merkle (Eds.), "Swarm Intelligence: Introduction and Applications," ISBN: 978-3-540-74088-9, Springer-Verlag, Berlin Heideberg, 2008.

[188]. E. Bonabeau, M. Dorigo, and G. Theraulaz, "Swarm Intelligence: From natural to artificial systems," Oxford University Press, 1999.

[189]. F. V. D. Bergh, "An Analysis of Particle Swarm Optimizers (PhD thesis)," University of Pretoria, Faculty of Natural and Agricultural Science, 2001.

[190]. R. C. Eberhart, Y. Shi, "Comparison between Genetic Algorithms and Particle Swarm Optimization," V. W. Porto, editor, Evolutionary Programming, vol.1447, Lecture Notes in Computer Science, pp. 611-616 Springer Heidelberg, 1998.

[191]. Z. S. Lu and S. Yan, "Multiuser detector based on particle swarm algorithm," Proc. IEEE Symp. Emerging Technologies: Mobile and Wireless Communications, Shanghai, China, 2004.

[192]. C. Liu and Y. Xiao, “Multiuser detection using the particle swarm optimization algorithm,” Proc. IEEE ISCIT, 2005.

[193]. C. H. Hsu, C. S. Tsou and F. J. Yu, Multicriteria tradeoffs in inventory control using memetic particle swarm optimization, International Journal of Innovative Computing, Information and Control, vol. 5, no.11(A), pp. 3755-3768, 2009.

[194]. S. N. Sivanandam and P. Visalakshi, “Multiprocessor Scheduling Using Hybrid Particle Swarm optimization with dynamically varying inertia,” International Journal of Computer Science & Applications, vol. 4, no. 3, pp. 95-106, 2007.

[195]. C. Lin, Y. Liu and C. Lee, “An efficient neural fuzzy network based on immune particle swarm optimization for prediction and control applications,” International Journal of Innovative Computing, Information and Control, vol. 4, no. 7, 2008.

[196]. G. D. Li, S. Masuda, D. Yamaguchi and M. Nagai, “The optimal GNN-PID control system using particle swarm optimization algorithm,” International Journal of Innovative Computing, Information and Control, vol. 5, no. 10(B), pp. 3457-3470, 2009.

[197]. C. Wang, P. Sui and W. Liu, “Improved Particle swarm optimization algorithm based on double mutation,” ICIC Express letters, vol. 3, no. 4(B), pp. 1417-1422, 2009.

[198]. J. H. Seo, C. H. Im, C. G. Heo, J. K. Kim, H. K. Jung and C. G. Lee, Multimodal function optimization based on particle swarm optimization, IEEE transaction on magnetic, vol.42, no.4, pp. 1095-1098, 2006.

[199]. S. Kirkpatrick, C. D. Gelatt, and M. Vecchi, “Optimization by simulated annealing,” Science, vol. 220, no. 4598, pp. 498-516, 1983.

[200]. V. Cenny, Thermodynamical approach to the traveling saleman problem: An efficient simulation algorithm, J. Optim Theory Appl., vol. 45, no. 1, pp. 41-51, 1985.

[201]. K. Y. Lee, and M. A. El-Sharkawi, “Modern Huristic Optimization Techniques: Theory and application to power systems,” John Wiley & Sons, Inc., Hoboken, New Jersey, 2008.

[202]. R. Romero, R. A. Gallego, A. Monticelli, “Transmission system expension planning by simulated annealing,” IEEE Trans Power Systems, vol. 11, no. 1, pp. 364-369, 1996.

[203]. C. Koulmas, S. R. Antony, and R. A. Jaen, “A survey of simulated annealing application to operations research problems,” Omega Int. J. Management Sci. vol. 22, no. 1 pp. 44-56, 1994.

[204]. A. H. Mantawy, Y. L. Abdel-Magid, S. Z. Selim, “Integrating genetic algorithms, tabu search and simulated annealing for the unit commitment problem,” IEEE Trans Power Systems, vol. 14, no. 3, pp. 829-836, 1999.

[205]. R. Hooke, T. A. Jeeves, “Direct search solution of numerical and statistical problems,” Journal of the Association for Computing Machinery (ACM), vol. 8, no. 2, pp. 212-229, 1961.

[206]. W. C. Yu, “Positive basis and a class of direct search techniques,” Scientia Sinica [Zhongguo Kexue], pp. 53-68, 1979.

[207]. V. J. Torczon, “On the convergence of pattern search algorithms,” SIAM Journal on Optimization, vol. 7, no. 1, pp. 1-25, 1997.

[208]. E. D. Dolan, R. M. Lewis, V. J. Torczon, “On the local convergence of pattern search,” SIAM Journal on Optimization, vol. 14, no. 2, pp. 567-583, 2003.

[209]. Lewis, R. Michael and V. Torczon, “Pattern Search Algorithms for Bound Constrained Minimization,” SIAM Journal on Optimization, vol. 9, no. 4, pp. 1082-1099, 1999.

[210]. Lewis, R. Michael and V. Torczon, “Pattern Search Methods for Linearly Constrained Minimization,” SIAM Journal on Optimization, vol. 10, no. 3, pp. 917-941, 2000.

[211]. A. R. Conn, N. I. M. Gould, and Ph. L. Toint, “A Globally Convergent Augmented Lagrangian Algorithm for Optimization with General Constraints and Simple Bounds,” SIAM Journal on Numerical Analysis, vol. 28, no. 2, pp. 545-572, 1991.

[212]. L. Wang, “Support Vector Machines: Theory and Application,” Springer-Verlag Berlin Heidelberg, 2005.

[213]. J. J. Judice, H. D. Sherali, I. M. Ribeiro and A. M. Faustino, “Complementarity active-set algorithm for mathematical programming problems with equilibrium constraints,” Journal of Optimization Theory and Applications, vol. 134, no. 3, pp. 467-481, 2007.

[214]. W. W. Hager and H. Zhang, “A New Active Set Algorithm for Box Constrained Optimization,” SIAM J. on Optimization, vol. 17, no. 2 pp. 526-557, 2006.

[215]. S. W. Sloan, “A steepest edge active set algorithm for solving sparse linear programming problems,” Int. J. Numerical Method in Engineering, vol. 26, no. 12, pp. 2671-2685, 2005.

[216]. Z. Odibat, S. Momani, and V. S. Erturk, “Generalized differential transform method: Application to differential equations of fractional order,” Appl. Math. Comput., vol. 197, pp. 467-477, 2008.

[217]. M. A. Z. Raja, A. K. Junaid and I. M. Qureshi, “Evolutionary Computational Intelligence in solving the Fractional Differential Equations,” LNCS vol. 5990, Part I, Springer-Verlag, Hue City, Vietnam, pp. 231-240, 2010.

[218]. P. Angeline, “Evolutionary Optimization versus Particle Swarm Optimization: Philosophy and Performance Differences,” Evolutionary Programming, vol. 1447, LNCS, Springer, pp. 601-610, 1998.

[219]. R.C. Eberhart, and Y. Shi, “Comparison between Genetic Algorithms and Particle Swarm Optimization,” In et al. V. W. Porto, editor, Evolutionary Programming, vol. 1447, LNCS, Springer, pp. 611-616, 1998.

[220]. R. Gorenflo, and F. Mainardi, “Fractional calculus: Integral and differential equations of fractional order,” CISM Lecture notes, International centre for mechanical Sciences, Udine, Italy 1996.

[221]. F. Mainardi, “Fractional relaxation and fractional diffusion equations,” Mathematical Aspects, Proceedings 12-th IMACS world Congress (Ed. W.F. Ames), Georgia Tech, Atlanta 1, pp. 329-332, 1994.

[222]. A. Saadatmandi, and M. Dehghan, “A new operational matrix for solving fractional-order differential equations,” 2009., DOI: 10.1016/j.camwa.2009.07.006.

[223]. C. Li and C. Tao, “On the fractional Adams method,” J. Comp. Math. Appl. vol. 58, no. 8, pp. 1573-1588, 2009.

[224]. Z. M. Odibat, and S. Momani, “Application of variational iteration method to nonlinear differential equations of fractional order,” International Journal of Nonlinear Sciences and Numerical Simulation, vol. 7, pp. 27-34, 2006.

[225]. R. M. A. Zahoor, J. A. Khan., and I. M. Qureshi, “Evolutionary Computation Technique for Solving Riccati Differential Equation of Arbitrary Order” WASET, vol. 58, pp. 303-309, 2009.

[226]. S. Momani, N. Shawagfeh, “Decomposition method for solving fractional Riccati differential equations”, J. Appl. Math. Comput., vol. 182, pp. 1083-1092, 2006.

[227]. M. A. Z. Raja, J. A. Khan., and I. M. Qureshi, “Solution of fractional order system of Bagley-Torvik equation using Evolutionary computational intelligence” Journal of Mathematical Problem in Engineering, vol. 2011, Article ID 675075, pp. 1-18, 2011.

[228]. E. A. Rawashdeh, “Numerical solution of semidifferential equations by collocation method,” Appl. Math. Comput., vol. 174, no. 2, pp. 869–876, 2006.

[229]. I. Podlubny, “Matrix approach to discretization of ODEs and PDEs of arbitrary real order,” File ID: #22071, Matlab Central file exchange, 12 Nov 2008, Updated 18 May 2009.

[230]. I. Podlubny, “Mittag-Leffler Function,” File ID 8738, Matlab Central, 2009. http://www.mathworks.com/matlabcentral/fileexchange/8738-mittag-leffler-function

[231]. M. A. Z. Raja, J. A. Khan and I. M. Qureshi, “Swarm intelligence optimized neural networks for solving fractional differential equations,” International Journal of Innovative Computing, Information and Control, vol. 7 no. 11, pp. 6301-6318, 2011.

[232]. M. A. Z. Raja, J. A. Khan and I. M. Qureshi, “A new Stochastic Approach for Solution of Riccati Differential Equation of Fractional Order,” Ann Math Artif Intell,vol. 60, no. (3-4), pp. 229-250, 2010.

[233]. M. A. El-Tawil, A. A. Bahnasawi, A. Abdel-Naby, “Solving Riccati differential equation using Adomian’s decomposition method,” Appl. Math. comput., vol. 157, pp. 503-514, 2004.

[234]. Z. Odibat, and S. Momani, “Modified homotopy perturbation method: application to quadratic Riccati differential equation of fractional order,” J. Chaos, Solitons & Fractal, vol. 36, no. 1, pp. 167-174, 2008.

[235]. B. Batiha, M. S. M. Noorani, and Hashim, “Application of Variational iteration method to general Riccati equation,” International mathematical forum, vol. 2, no. 56, pp. 2759-2770, 2007.

[236]. S. Abbasbandy, “Homotopy perturbation method for quadratic Riccati differential equation and comparison with Adomians decomposition method,” Appl. Math. Comput. vol. 172, pp. 485-490, 2006.

[237]. Y. Tan, S. Abbasbandy, “Homotopy analysis method for quadratic Riccati differential equation,” Commun. Nonlinear Sci. Numer. Simulat., vol.13, pp. 539-546, 2008.

[238]. Y. Li, Solving a nonlinear fractional differential equation using Chebyshev wavelets. Commun. Nonlinear Sci. Numer. Simulat. (2009), doi:10.1016/j.cnsns.2009.09.020.

[239]. M. A. Z. Raja, J. A. Khan, and I. M. Qureshi, “Swarm Intelligence Optimized neural network for solving Fractional Order Systems of Bagley-Tervik equation” accepted for publication in intelligent engineering systems for Electrical Engineering and Communication.

[240]. V. S. Erturk et al, “Application of generalized differential transfrom method to multi-order fractional differential equations,” Communications in Nonlinear Science and Numerical Simulation, vol. 13, pp. 1642-1654, 2008.