World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:3, No:10, 2009

Evolutionary Computation Technique for Solving Riccati Differential Equation of Arbitrary Order

Raja Muhammad Asif Zahoor, Junaid Ali Khan, and I. M. Qureshi

Open Science Index, Computer and Information Engineering Vol:3, No:10, 2009 publications.waset.org/2829.pdf

***Abstract*—**In this article an evolutionary technique has been used for the solution of nonlinear Riccati differential equations of fractional order. In this method, genetic algorithm is used as a tool for the competent global search method hybridized with active-set algorithm for efficient local search. The proposed method has been successfully applied to solve the different forms of Riccati differential equations. The strength of proposed method has in its equal applicability for the integer order case, as well as, fractional order case. Comparison of the method has been made with standard numerical techniques as well as the analytic solutions. It is found that the designed method can provide the solution to the equation with better accuracy than its counterpart deterministic approaches. Another advantage of the given approach is to provide results on entire finite continuous domain unlike other numerical methods which provide solutions only on discrete grid of points.

***Keywords*—**Riccati Equation, Non linear ODE, Fractional differential equation, Genetic algorithm.

I. Introduction

COUNT Jacopo Francesco Riccati in the year 1720 introduced to his friend Giovanni Rizzetti the following two equations. These equations in modern symbols can be written as follow:

$$\frac{dy}{dt} = \alpha y^2 + \beta t^m \tag{1}$$

$$\frac{dy}{dt} = \alpha y^2 + \beta t + \gamma t^2 \tag{2}$$

where t is the independent variable and m, α, β and γ are the constants. This is most likely the first document witnessing the early days of the Riccaiti differential equation. The detail history and importance of the equation can be seen in literature for the interested reader [1], [2].

In this article, a new numerical technique has been presented for the solution of Riccati differential equation of arbitrary order. Its general form can be written as

$$\frac{d^\alpha y(t)}{dt^\alpha} = A(t) + B(t)y + C(t)y^2(t), \quad 0 < t \le t_1 \tag{3}$$

with initial conditions given as

$$\frac{d^k}{dt^k} y(0) = c_k, \quad k = 0,1,2,\ldots,N-1$$

and boundary conditions at $t = t_0$, for $0 < t_0 \le t_1$, is written as

$$\frac{d^k}{dt^k} y(t_0) = b_k, \quad k = 0,1,2,\ldots,N-1$$

where α is the order of the equations, $\alpha > 0$, $\alpha \in \Re$, $N = \lceil \alpha \rceil$, $y(t)$ is the solution of the equation, $A(t), B(t)$ and $C(t)$ are known functions t_1 is the constant representing the inputs span within the close interval $[0, t_1]$ for some $t_1 > 0$, c_k and b_k are the constants giving the initial and boundary condition respectively.

The different response expressions of equation (3) can be obtained by varying the order α of the equation. In the case of $\alpha = 1$, the expression (3) is known as classical first order Riccati differential equation. This equation has a paramount importance, especially for the developments of Calculus of Variations, [3] Optimal Filtering [4] and Control [5].

In case the order α of expression (3) is non-integer, then it transformed into a fractional Riccati differential equation. The value of $\alpha = \frac{1}{2}$ has special popularity. This is due to many of the model equations developed in classical fractional calculus by using this particular order of the derivative [6]. However in recent applications much more generic values of α appear in the equations [7]. Therefore researchers are interested to investigate the numerical and analytical methods to solve the Riccati differential equation of the arbitrary order. In this regard relative new analytical technique has been developed using adomian decomposition method for solving the fractional Riccati differential equations [8]. Recently some of the deterministic method have been extended for the differential equation of arbitrary order e. g. Adomian Decomposition method, [9] Homotopy perturbation method, [10] and Fractional Adams-Moulton method [11] etc. Stochastic methods have been developed for the solutions of differential equations are confined to integer order only [12], [13].

One should investigate different modification in the existing stochastic techniques such that the solution to differential equations of arbitrary order can be obtained. The aim of this article is to propose such a methodology in which the strength of feed forward neural network is utilized for modeling of the differential equations.

Raja Muhammad Asif Zahoor is with the Electronics Engineering Department, facility of Engineering and technology, Islamic International University Islamabad, Pakistan. (phone: +923009893800; fax: 303-555-5555; e-mail: asif.phdee10@iiu.edu.pk; rasifzahoor@yahoo.com).

Junaid Ali Khan is with the Electronics Engineering Department, Facility of Engineering and Technology, Islamic International University Islamabad, Pakistan. (e-mail: junaid.phdee17@iiu.edu.pk;).

I. M. Qureshi is with the Electrical Engineering Department, Air University Islamabad, Pakistan (e-mail: imq313@yahoo.com).

Open Science Index, Computer and Information Engineering Vol:3, No:10, 2009 publications.waset.org/2829.pdf

The learning of the unknown parameters in neural network has been achieved with hybrid intelligent algorithms mainly based on Genetic Algorithm (GA). The design scheme of stochastic method should be such that it is capable for finding the reliable solution of Riccati differential equation of arbitrary order.

It is necessary to introduce some definition and relations which are used in this article. In case of fractional calculus, Riemann-Liousville definitions of fractional integral or derivative with lower terminal at 0 is taken. Then the definition of fractional integral [14], [15] of function f of order α with respect to t can be given as

$$D^{-\alpha} f(t) = \frac{1}{\Gamma(\alpha)} \int_0^t (t-\xi)^{\alpha-1} f(\xi) d\xi \tag{4}$$

for $t > 0$ and α is a real and > 0

where $D = \frac{d}{dt}$ is the differential operator. So by replacing the order α in expression (4) with $-\alpha$ the fractional derivative is defined.

The fractional derivative of the exponential function

$$f(t) = e^{\lambda t} \tag{5}$$

can be written as

$$D^{\alpha} f(t) = t^{-\alpha} M_{1,1-\alpha}(\lambda t) \tag{6}$$

where $M_{1,1-\alpha}(\lambda t)$ be the classical Mittag-Leffler function of two parameters $\beta = 1$ and $\gamma = 1 - \alpha$ is defined as

$$M_{\beta,\gamma}(t) = \sum_{k=0}^{\infty} \frac{t^k}{\Gamma(\beta k + \gamma)} \qquad (\beta > 0, \gamma > 0) \tag{7}$$

II. Mathematical Model for Solving Riccati Differential Equations

In this section we are providing the introductory material for mathematical modeling of Riccati differential equations of arbitrary order with feed forward artificial neural network.

Integer order case: Take the general form of Riccati differential equation of integer order, i.e. $\alpha = n$ where n is some positive integer then equation (3) can be written as

$$\frac{d^n y}{dt^n} = a(t) + b(t) y + c(t) y^2, \quad 0 \le t \le t_1 \tag{8}$$

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. Hence the solution y of the differential equation along with its n order derivatives $\frac{d^n y}{dt^n}$ can be approximated by the following continuous mapping in neural network methodology [16] [18]

$$\hat{y}(t) = \sum_{i=1}^{m} \delta_i f(w_i t + b_i) \tag{9}$$

$$\frac{d^n \hat{y}}{dt^n} = \sum_{i=1}^{m} \delta_i \frac{d^n}{dt^n} f(w_i t + b_i) \tag{10}$$

where δ_i, w_i and b_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 given as

$$f(x) = 1/(1 + e^{-x}). \tag{11}$$

The mathematical model for the given equation (8) can be formulated the linear combinations of the networks (9) and (10). It means that the solution y is approximated with $\hat{y}$ subject to some appropriate unknown weights.

Fractional order case: The networks given in (9) and (10) could not apply directly to represent the fractional differential equations, 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 of fractional differential equation, due to its universal function approximation capability as well as its straightforward implementable fractional derivative.

Now take the $f(x) = e^x$ be the activation function then the network presented by (9) and (10) can be given as

$$\hat{y}(t) = \sum_{i=1}^{m} \delta_i e^{w_i t + b_i} \tag{12}$$

$$D^{\alpha} \hat{y}(t) = \sum_{i=1}^{m} \delta_i e^{b_i} t^{-\nu} M_{1,1-\alpha}(w_i t) \tag{13}$$

The linear combination of the networks (12) and (13) can represent approximate mathematical model of the differential equation of fractional order given in expression (3). It is named as fractional differential equation neural network (FDE-NN). The activation function for (12) is exponential function, while in case of (13) it is integer and non-integer derivative of exponential function respectively. It means that the solution y of fractional Riccati differential equation also be approximated with $\hat{y}$ subject to some appropriate unknown weights.

The mathematical model for the Riccati differential equation of arbitrary order with the help of neural networks can be formulated.

III. Evolutionary Computing

In this section the learning of the unknown weights of networks representing the equation with the help of efficient computational algorithm is narrated.

These learning algorithms are mainly based on genetic algorithm (GA) and aided with rapid local methods like active set algorithm. One of the prominent features of GAs is that unlike other algorithms, they do not get stuck in local minima. GA incorporate parallel procedure as well as structured strategy for randomly searching high aptitude points. GA consists of three fundamental operations: 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 fitness

Open Science Index, Computer and Information Engineering Vol:3, No:10, 2009 publications.waset.org/2829.pdf

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 bits at some location in the strings. The generic flow chart of the GA algorithm is given in Figure 1. [17].

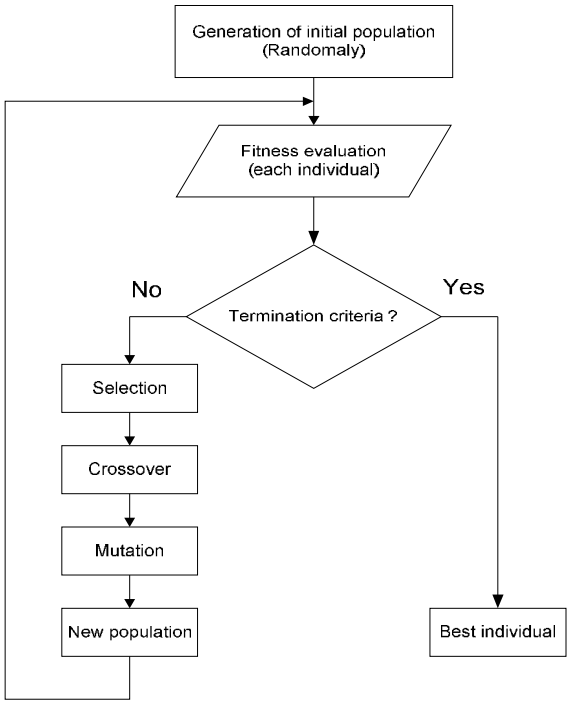

Fig. 1 Flow chart of Genetic Algorithm

Randomly generated initial population consists of finite set of chromosome each with as many numbers of genes as the unknown weights in differential equation neural network (DE-NN). The objective function to be minimized is given as the sum of errors

$$e_j = e_1^j + e_2^j \quad j = 1,2,\ldots\ldots \tag{14}$$

where e_1^j is given as

$$e_1^j = \frac{1}{n}\sum_{i=1}^{n}\left[\frac{d^\alpha \hat{y}(t_i)}{dt_i^\alpha} - A(t_i) - B(t_i)y(t_i) - C(t_i)y^2(t_i)\right]^2\Bigg\}_j \tag{15}$$

where n is the number of time steps, $d\hat{y}/dt$ and $\hat{y}$ are given by expression (9) and (10). Similarly then error e_2^j due to initial condition, for example $y(0)=1$ can be written as

$$e_2^j = [\{\hat{y}(0)\}^2 - 1]^2\Big\}_j \tag{16}$$

The iterative process for optimization continues until user define number of cycles is achieve or pre-defined level of error e_j is obtained.

Algorithm is given in the following steps:

Step 1: Randomly generate bounded real value to form initial population of P number of the individual or chromosome. Each individual represents the unknown weights of neural network. The initial population is scatter enough for better search space of the algorithm.

Step 2: Create a Q number of subpopulation each with P/Q individual.

Step 3: Initialize the algorithm and its parameter values setting for execution.
Set the number of variable equivalent to element in the individual, Set the number of generation, Set the fitness limit, Set Elite count equal to 3 and crossover fraction 0.75 for reproduction, Set Migration in forward and backward direction both, Start generation count, etc.

Step 4: Fitness calculation by using the fitness function given in expressions (14 to 16).

Step 5: Ranking is carried out for each individual of the populations on the basis of minimum fitness values. Store the best fitted candidate solution.

Step 6: Check for Termination of the algorithm, which is set as either predefine fitness value achieve i.e. MSE 10^{-8} for integer order case of the equations and 10^{-4} for non-integer order cases of the equation or Number of cycles complete.
If yes go to step 9 else continues

Step 7: Reproduction of next generation on the basis of
Crossover: Call for scattered function
Mutation: Call for Adaptive Feasible function
Selection: Call for Stochastic Uniform function
Elitism

Step 8: Repeat the procedure from step 3 to step 6 with newly generated population until total number of cycle complete.

Step 9: Refinement of result using active-set algorithm (Call *FMINCON* function of MATLAB) for local search technique by taking the best fitted individual of step 5 as start point of the algorithm. Store in the memory the refined value of fitness along with the best individual for this run of the algorithm.

Step10: Repeat the step 1 to step 9 for sufficient large number of the run to make the statistical analysis of the algorithm. Store these result in a file for analysis later.

IV. Simulation and Results

In this section the simulation and results are provided for two different problems associated with Riccati differential equation of arbitrary order to prove the applicability of our algorithm. Comparative studies are also given.

A. Problem I

Consider the Riccati differential equation

$$D^\alpha y(t) = -y^2(t) + 1, \quad 0 \le t \le 1 \tag{17}$$

with initial condition as $y(0) = 0.$

First order case. Let take the value of order $\alpha = 1.$ Then

Open Science Index, Computer and Information Engineering Vol:3, No:10, 2009 publications.waset.org/2829.pdf

the exact solution of the equation is given as

$$y(t) = \frac{e^{2t} - 1}{e^{2t} + 1}. \tag{18}$$

The adomian decomposition method (ADM) can provide the approximate solution of the equation (17) in the form of a rapidly convergent series with easily computable terms [8]. The solution in a series form is given as

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

To solve this with Differential equation neural network (DE-NN) represented by (9) and (10) with number of neurons $m = 8$ are taken. Then the total number of 24 unknown parameter (w_i , δ_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 chromosome 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 $n = 11$ in the expression (15). Therefore the fitness function is formulated as

$$e_j = \frac{1}{11}\sum_{i=1}^{11}\left[\frac{d\hat{y}(t_i)}{dt} + \{\hat{y}(t_i)\}^2 - 1\right]^2 + \{\hat{y}(0)\}^2\Bigg)_j, \tag{20}$$

$$\text{for} \quad j = 1,2,3\ldots\ldots$$

where j be the number of generation, $\hat{y}$, and $\frac{d\hat{y}}{dt}$ are networks provide in equation (9), and (10) respectively with $m = 8$ number of neurons. Our scheme runs iteratively in order to find the minimum of fitness function e_j, with stoppage criteria as 800 number of generations or fitness value of $\min |e_j| \le 10^{-8}$ whichever comes earlier. In our case the $|e_j| < 10^{-8}$ is achieved mostly before the number of cycles ends. One of the unknown weights learned by DE-NN algorithm with fitness value of $\min |e_j| = 8.6429 \times 10^{-8}$ are provided in Table 1. Using these weights in expression (9) one can find the solution to this problem for any input time t between 0 and 1.

The exact solution and approximate analytic solution represented by equations (18) and (19) respectively as well as DE-NN solution for the equation for input time $t \in (0, 2)$ with a step size of 0.2 are obtained. Results are shown in Figure 2 and given in Table 2.

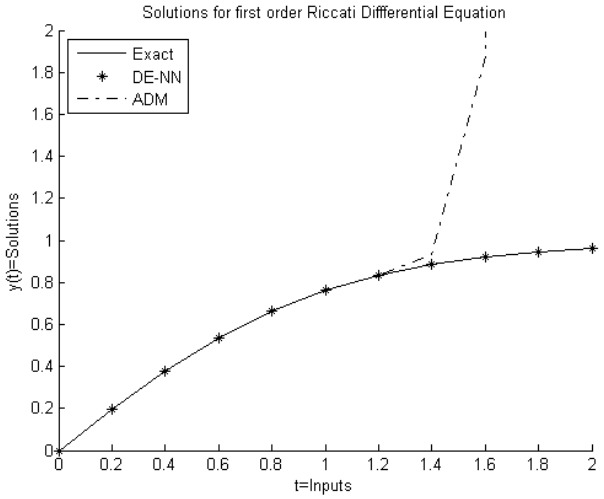

Fig. 2 Graphical representation for the solution of ODE in problem I for first order case $\alpha = 1$ and $0 \le t \le 2$.

It can be seen that solutions obtained by our algorithm are closest fit to the exact solutions.

Fractional order case. To prove the validity of our approach let us solve relative simple example of fractional differential equation with known exact solution such that comparison of results can be made. Then apply same approach for solving equation (17) for which the exact solution is extremely difficult to drive.

Consider the fractional differential equation

$$\begin{aligned} D^{\alpha} y(t) &= t^2 + \frac{2}{\Gamma(3-\alpha)} t^{2-\alpha} - y(t), \\ y(0) &= 0, \quad 0 < \alpha \le 1 \end{aligned} \tag{22}$$

whose exact solution is given by

$$y(t) = t^2. \tag{23}$$

We calculate the approximate solution by mean of fractional differential equation neural networks (FDE-NN) constructed with the networks represented in equations (12) and (13). Same procedure is adapted as in case of DE-NN but here $\max j = 2000$ number of generation is taken. The fitness function for this scheme can be provided on similar pattern to equation (20) with $\hat{y}$, and $D^{1/2}\hat{y}$ are networks provide in equation (12), and (13) respectively using $m = 8$ number of neurons. The optimize weights obtain by this stochastic scheme is provided in Table 3. Using these weights we obtained the solution of equation (22) for some inputs. The result are graphically compared in Figure 3 and provided in Table 3. It can be seen that this method can effectively approximate the solution of fractional differential equations as well.

Open Science Index, Computer and Information Engineering Vol:3, No:10, 2009 publications.waset.org/2829.pdf

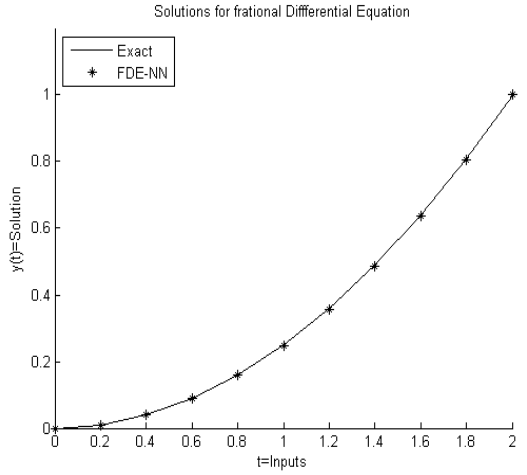

Fig. 3 Graphical representation for the solution fractional differential equation of of order $\alpha = 1/2$ and $0 \le t \le 2$.

Now the same methodology applied (FDE-NN) for solving the fraction Riccati differential equation by taking the value of order $\alpha = \frac{1}{2}$ in the expression (17). Comparison is carried out with extended ADM method which can provide the approximate analytic solution for such kind of the equation as well [8]. The solution by ADM method can be given as

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

where for simplicity we let $x = t^{1/2}$ in the above relation (21).

The approximate analytic solution represented by expression (21) and stochastic proposed solution for the equation (17) 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 1 for order $\alpha = 1/2$. The solution obtained along with fitness value at some inputs is provided in Table 4. It can be inferred from the results that our algorithm can also be provided approximate solutions of the fractional Riccati differential equation.

B. Problem II

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

$$D^{\alpha} y(t) = 2y(t) - y^2(t) + 1, \quad 0 \le t \le 1 \tag{24}$$

with intial condition as $y(0) = 0.$

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

$$y(t) = 1 + \sqrt{2}\tanh(\sqrt{2}t) + \frac{1}{2}\log(\frac{\sqrt{2}-1}{\sqrt{2}+1}). \tag{25}$$

The analytic solution of (24) by adomian decomposition technique [8] for the case $\alpha = 1/2$ is provided as

$$\begin{aligned} y(t) &= 1.12838x + 2.0x^2 + 2.05121x^3 \\ &\quad - 0.2732x^4 + \cdots\cdots + 80.4206x^{21} \end{aligned} \tag{26}$$

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 taking 6 number of neuron. DE-NN or FDE-NN method used to determine 18 numbers of unknown weights. The stoppage criterion is $\min e_j \le 10^{-6}$ or maximum number of generations is equal to 2000. The fitness function e_j used in said problem for the fractional order case $\alpha = 1/2$ is given as

$$\begin{aligned} e_j &= \frac{1}{11}\sum_{i=1}^{11}[D^{1/2}\hat{y}(t_i) + \{\hat{y}(t_i)\}^2 - 2\hat{y}(t_i) - 1]^2 + \{\hat{y}(0)\}^2 \Big)_j \\ &\quad j = 1, 2, 3\ldots\ldots \end{aligned} \tag{27}$$

Where $\hat{y}$, and $D^{1/2}\hat{y}$ are networks provide in equation (11), and (12) respectively using $m = 6$ number of neurons. Adaptive weights w_i, δ_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 (9) and (12) for finding the solution of the equations for some inputs between 0 and 1 are summarized in Table 5, where the order $\alpha = 1$ iand $\alpha = 1/2$ is taken for the equations.

V. Conclusions

On the basis of the simulation and result obtained in the last section it can be concluded that nonlinear Riccati differential equation of arbitrary order can be solved using 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. In future biological inspire method are studied to solve these problems.

References

[1] S. Bittanti, "History and Prehistory of the Riccati Equation" Proceedings of the 35th Conference on Decision and Control Kobe, Japan December 1996.
[2] Reid W. T, "Riccati DifferentialEquations", Academic Press, 1972.
[3] Goldstine H. H, "A History of the Calculus of Variations from the 17th through the 19th Century", Springer-Verlag,, 1980.

[4] Mtter S. K, "Filtering and Stochastic Control: a Historical Perspective", in IEEE Control n Systems, *pp.* 67-76, June 1996.
[5] B.D Anderson, J.B. Moore, "Optimal Control-Linear Quadratic Methods", Prentice-Hall, New Jersey, 1990.
[6] K. Diethelm, J. M. Ford, N. J. Ford, W. Weilbeer, "Pitfalls in fast numerical solvers for fractional differential equations" J. Comput. Appl. Math, 186 pp 482-503, 2006.
[7] F. Mainardi, G. Pagnini, and R. Gorenflo, "Some aspects of fractional diffusion quations of single and distributed orders" Journal of Applied Mathematics and Computation, 187 No 1, pp 295-305, 2007.
[8] S. Momani, N. Shawagfeh, "Decomposition method for solving fractional Riccati differential equations", Journal of Applied Mathematics and Computation 182, pp 1083-1092, 2006.
[9] Duan Junsheng, An Jianye, and Xu Mingyu, "Solution of system of fractional differential equations by Adomian decomposition Method", Appl. Math. J. Chinese Univ. Ser. B, 22(1) pp 7-12, 2007.
[10] Shaher Momani, and Zaid Odibat, "Comparision between the homotopy perturbation method and the variational iteration method for linear fractional partial differential equations", Computer & Mathematics with Applications, Vol 54, Issue 7-B pp 910-919. 2007
[11] L. Galeone, and R. Garrappa, "Fractional Adams-Moulton method" Mathematics and Computers in Simulation, vol. 79 issue 4 pp 1358-1367, 2008.
[12] G. Tsoulos and I. E. Lagaris, "Solving differential equations with genetic programming", Genetic programming and Evolvabe Machines, Vol. 7 No. 1 pp 33-54, 2006.
[13] Paul E. MacNeil, "Genetic algorithms and solutions of an interesting differential equation", proceedings of the 10th annual conference on Genetic and evolutionary computation, pp 1711-1712, 2008.
[14] Miller, K. S. and Ross, B., "An Introduction to the Fractional Calculus and Fractional Differential Equations" John Wiley and Sons, Inc., New York 1993.
[15] Oldham, K. B. and Spanier, J., "The Fractional Calculus" Academic Press, New York 1974.
[16] Daniel R. Rarisi et al. "Solving differential equations with unsupervised neural networks", Chemical engineering and processing, 42 pp 715-721 2003.
[17] Lucie P. Aarts and Peter Van Der Veer, "Neural Network Method for solving the partial Differential Equations" Neural Processing Letters 14 pp 261-271, 2001.
[18] A. Junaid, M. A. Z. Raja, and I. M. Qureshi, "Evolutionary computing approach for the solution of initial value problem in ordinary differential equation" Proceeding of World academy of science engineering and technology, vol. 55, pp 578-581 July 2009.

Open Science Index, Computer and Information Engineering Vol:3, No:10, 2009 publications.waset.org/2829.pdf

TABLE I
WEIGHTS OBTAINED FOR ODE IN PROBLEM I

Index i	**Unknown Weights**					
	w_i		δ_i		b_i	
	$\alpha = 1$	$\alpha = 1/2$	$\alpha = 1$	$\alpha = 1/2$	$\alpha = 1$	$\alpha = 1/2$
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

TABLE II
COMPARISON OF RESULTS FOR THE SOLUTION OF ODE IN PROBLEM FOR FIRST ORDER CASE $\alpha = 1$.

Time i	**Exact** $y(t)$	**ADM** $\varphi(t)$	**DE-NN** $\phi(t)$	**Error**	
				$y(t)-\varphi(t)$	$y(t)-\phi(t)$
0	0	0	-0.0000	0	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	0.7616	000.7616	0.7615	-2.791e-05	01.171e-04
1.2	0.8337	000.8353	0.8338	-0.0016	-1.720e-04
1.4	0.8854	000.9356	0.8857	-0.0502	-3.663e-04
1.6	0.9217	001.8762	0.9220	-0.9545	-3.089e-04
1.8	0.9468	013.5716	0.9469	-12.624	-4.592e-05
2	0.9640	126.6693	0.9639	-125.705	1.178e-04

TABLE III
WEIGHTS OBTAINED ALONG WITH THE SOLUTION OF DIFFERENTIAL EQUATIONS OF FRACTIONAL ORDER.

Index i	**Unknown Weights**			**Time**	**Exact**	**FDE-NN**	**Error**
	w_i	δ_i	b_i				
1	0.1960	0.0830	0.2050	0	0	-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	4.328 e-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

TABLE IV
THE SOLUTION OF FRACTIONAL RICCATI DE IN PROBLEM 1 HAVE ORDER $\alpha = 1/2$

Time	**FDE-NN**	**Value of Objective Function** e_j
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	0.6992	2.464e-02

TABLE V
COMPARISON OF RESULTS FOR THE SOLUTION OF ODE IN PROBLEM II.

Time	**First order Case** $\alpha = 1$			**Fractional Order Case** $\alpha = 1/2$	
	Exact $y(t)$	**DE-NN** $\varphi(t)$	**Error** $y(t) - \varphi(t)$	**FDE-NN**	**Value of Objective Function** e_j
0	0	-0.0000	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	02.398e-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	1.6895	1.6896	-6.20e-05	1.7957	2.337 e-03

Open Science Index, Computer and Information Engineering Vol:3, No:10, 2009 publications.waset.org/2829.pdf