Numerical Solution to Riccati Equations using Evolutionary Algorithm Technique Hybridized with Bernstein Polynomials

Suheel Abdullah Malik[1], Ijaz Mansoor Qureshi[2], Muhammad Amir[1], Ihsanul Haq[1], Aqdas Naveed Malik[1]

[1] Department of Electronic Engineering, Faculty of Engineering and Technology, International Islamic University, Islamabad, Pakistan
[2] Department of Electrical Engineering, Air University, Islamabad, Pakistan

Abstract: In this paper, a heuristic method based on the hybrid approach of Bernstein polynomials and evolutionary algorithms is proposed for solving Riccati nonlinear differential equations. Bernstein polynomial basis with unknown coefficients are used to construct an approximate solution of the nonlinear differential equation. The differential equation is then transformed into an equivalent global error minimization problem. A trial solution is formulated using an exclusive fitness function with unknown coefficients. One of the popular evolutionary algorithms such as genetic algorithm (GA) is used to solve the minimization problem and to obtain the unknown coefficients. The effectiveness of the proposed method is illustrated by solving the Riccati equations in contrast with some well known classical methods, as well as, the exact solutions. The comparisons of numerical solutions validate the efficacy and viability of the suggested method. The results are found to be in good agreement with the exact solutions and more accurate than some of the important classical deterministic methods.

Keywords: Riccati equation; Bernstein polynomials; Evolutionary algorithm (EA); Genetic Algorithm (GA)

1. Introduction

The systems of nonlinear ordinary differential equations (NODEs) appear in many physical problems. The majority of NODEs either do not have the exact solution or obtaining the same analytically is difficult, therefore these equations are usually solved using various approximate analytical and numerical methods.

In this paper, we present numerical solution to the well-known Riccati equation of the following form [1- 6].

$$y'(x) = P(x) + Q(x)y + R(x)y^2 \qquad (1)$$

subject to the initial condition

$$y(0) = c \qquad (2)$$

where P(x), Q(x), and R(x) are real functions and c is an arbitrary constant.

The differential equation (1) introduced by the mathematician Count Jacopo Francesco Riccati (1676–1754) is imperative due to its existence in diverse fields of engineering and science such as optimal control, random processes, quantum mechanics, diffusion problems, etc. [1,3,5]. The interested readers may refer [6] and references therein for fundamental theories and applications of the Riccati equation. To date an incredibly great number of methods have been utilized for solving various forms of the Riccati equation, see [1-14]. These methods include optimal homotopy asymptotic method (OHAM) [1], homotopy perturbation method (HPM) [2], modified homotopy perturbation method (MHPM) [3], adomian decomposition method (ADM) [4], variational iteration method (VIM) [5], multistage variational iteration method (MVIM) [6], iterative decomposition (IDM) [7], homotopy analysis method (HAM) [12] etc.

In recent years, many authors have used evolutionary computation (EC) and neural

(DOI: dx.doi.org/14.9831/1444-8939.2014/2-6/MAGNT.6)

network (NN) based techniques as an alternative for solving numerous nonlinear systems of ODEs efficiently [15-22]. Raja et al. [15] used particle swarm optimization (PSO) based NN method for solving Riccati equations of arbitrary order. Very recently Malik et al. [16] employed heuristic technique based on the hybrid genetic algorithm (HGA) for solving nonlinear singular boundary value problems arising in physiology. Khan et al. [20] used PSO based NN for solving nonlinear ODEs especially Weissinger's equation.

Bernstein polynomials (B-polynomials) based methods have been explored by many authors for solving systems of ODEs [23-28]. Among many authors Yüzbaşi [23] solved Riccati differential equation using a collocation method based on B-polynomials. Baleano et al. [26] used operational matrix method based on Bernstein polynomials for solving fractional Riccati differential equation. Bhatti and Bracken [24] used Galerkin method based on Bernstein polynomials for solving linear and nonlinear ODEs.

The prime objective of this paper is to obtain the numerical solution of the Riccati equation (1) using a novel technique based on hybrid Bernstein polynomial basis and nature inspired computation. The method employs Bernstein polynomial basis with unknown coefficients. The given ODE is transformed into an optimization problem. One of the most popular evolutionary algorithms such as genetic algorithm (GA) is utilized for solving the minimization problem and to obtain the unknown coefficients. To the best of our knowledge, none has attempted using a stochastic technique based on the hybridization of Bernstein polynomial basis with evolutionary algorithms for solving Riccati nonlinear differential equation. The efficiency and the viability of the presented method is demonstrated by solving two different forms of Riccati equation (1) in comparison with the exact solutions as well as with some well known classical deterministic method including OHAM [1], MHPM [3], ADM [4], VIM [5], MVIM [6], Bernstein polynomials based collocation method (BPCM) [23], and stochastic solver based on hybrid PSO and NN (PSO-NN) [15].

The organization of the paper is as follows: In section 2, we introduce the stochastic global search algorithm such as genetic algorithm (GA) used in this study. In section 3, we give an introduction of Bernstein polynomials. In section 4, brief description of the proposed scheme is provided. In section 5, we present numerical results followed by the discussion on our findings. Finally in section 6 we give some concluding remarks and future work.

2.1 Global Search Evolutionary Algorithm: GA

Evolutionary algorithms (EAs) are population based heuristic search methods that use the nature and biologically inspired process in an iterative manner to reach an optimal solution [29]. (GA) is one of the well-known algorithms in EAs that finds the optimal solution of a problem from a randomly generated population of individuals called chromosome.

Each individual within a population is regarded as a possible solution to the problem. The individuals within a population are evaluated using a fitness function that is specific to the problem at hand. The algorithm evolves population iteratively by means of three primary operations: selection, crossover, and mutation to reach the optimal solution [29].

The procedural steps of GA are given as follows while the values and settings of the parameters for its implementation are given in Table 1.

(DOI: dx.doi.org/14.9831/1444-8939.2014/2-6/MAGNT.6)

Algorithm 1: Genetic Algorithm (GA)

Step 1: (Population Initialization)
A population of N chromosomes is generated randomly. The length of each chromosome is chosen equal to the number of unknown coefficients.

Step 2: (Fitness Evaluation)
The fitness of each chromosome is evaluated using an exclusive fitness function.

Step 3: (Selection and Reproduction)
The chromosomes for next generation are selected on the basis of their fitness value using the crossover operation.

Step 4: Mutation
Mutation operation introduces random changes in the genes to maintain the genetic diversity to find a good solution.

Step 5: (Stoppage Criteria)
The algorithm stops if the number of generations has exceeded or a desired fitness is reached.
Else repeat steps 2 to 4.

2.2 Bernstein Polynomials and Their Properties

The Bernstein polynomials (B-polynomials) of nth degree are defined on the interval [0, T] as [23-25]

$$B_{i,n}(x) = \binom{n}{i} \frac{x^i (T-x)^{n-i}}{T^n} \qquad (3)$$

where

$$\binom{n}{i} = \frac{n!}{i!\,(n-i)!} \qquad (4)$$

There are $n+1$ nth degree B-polynomials. Usually we set $B_{i,n}(x) = 0$, if $i < 0$ or $i > n$, for mathematical convenience. Each B-polynomial is positive, i.e., $B_{i,n}(x) > 0$ and also the sum of all the B-polynomials is unity for all real $x \in [0, T]$, i.e., $\sum_{i=0}^{n} B_{i,n}(x) = 1$. B-polynomials defined over the interval form a complete basis [23-25] over the interval [0, T]. For detail of B-polynomials and their properties see [23-25]. These polynomials are quite easy to write and can be generated recursively. The ith nth degree B-polynomial and the first derivative of the nth degree B-polynomials are given by [23-25]

$$B_{i.n}(x) = \frac{(T-x)}{T} B_{i.n-1}(x) + \frac{x}{T} B_{i-1.n-1}(x) \qquad (5)$$

$$B'_{i,n}(x) = \frac{n}{T}\left(B_{i-1.n-1}(x) - B_{i.n-1}(x)\right) \qquad (6)$$

2.3 Brief Description of Proposed Method

We consider NODE given by (1) and assume that the approximate solution $\hat{y}(x)$ and its first derivate $\hat{y}'(x)$ are given by linear combinations of Bernstein polynomials (B-polynomials) basis functions of degree n as follows.

$$\hat{y}(x) = \sum_{i=0}^{n} a_i\, B_{i,n}(x), \qquad n \geq 1 \qquad (7)$$

$$\hat{y}'(x) = \sum_{i=0}^{n} a_i\, B'_{i,n}(x) \qquad (8)$$

where $a_0, a_1, \ldots, a_n$ are real valued unknown coefficients to be determined.

The unknown coefficients are achieved using the application of evolutionary algorithm such as GA. To employ GA the unknown parameters in (7) are chosen as a chromosome. The NODE (1) along with the given initial condition (2) is converted into an optimization problem by developing a trial solution using a fitness function given by

$$\varepsilon_j = \varepsilon_1 + \varepsilon_2 \qquad (9)$$

where j represents the generation index, ε_1 represents the mean of sum of square errors associated with the given NODE (1), and ε_2 represents the mean square error associated with the given initial condition (2), which are given respectively by (10) and (11) as below.

(DOI: dx.doi.org/14.9831/1444-8939.2014/2-6/MAGNT.6)

$$\varepsilon_1 = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}'(x_i) - P(x_i) - Q(x_i) - R(x_i)\hat{y}^2(x_i)\right)^2 \quad (10)$$

$$\varepsilon_2 = (y(0) - c)^2 \quad (11)$$

where *N* is the total number of steps taken in the interval $x \in [0, T]$.

The minimization of fitness function (ε_j) given by (9) is performed by employing GA for obtaining the unknown coefficients contained in the fitness function. The optimal values of the unknown coefficients corresponding to the minimum fitness are obtained and consequently the approximate solution $\hat{y}(x)$ is achieved.

3. Numerical Results and Discussion

In this section we used the proposed method to solve two different forms of Riccati nonlinear differential equation (1). To validate the effectiveness of the method comparisons are made with the exact solutions, some well known conventional methods including OHAM [1], MHPM [3], ADM [4], VIM [5], MVIM [6], B-polynomials based collocation method (BPCM) [23], as well as stochastic solver (PSO-NN) [15].

Example 1: Consider the following Riccati nonlinear differential equation [1- 6].

$$y'(x) = -y^2(x) + 1, \quad y(0) = 0 \quad (12)$$

The exact solution of (12) is give by [1-6]

$$y_{exact}(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \quad (13)$$

To apply the proposed scheme we assume the approximate solution of (12) is given by (7) with $n = 7$ (B-polynomial of degree 7) as follows.

$$\hat{y}(x) = \sum_{i=0}^{7} a_i B_{i,n}(x), \qquad n \geq 1 \quad (14)$$

The given NODE (12) is transformed into an optimization problem using the FF to find the unknown coefficients $(a_0, a_1, \ldots, a_7)$. The numerical solution is obtained in the interval $x \in (0,1)$ with a step of 0.1, therefore FF is given by

$$\varepsilon_j = \frac{1}{11}\sum_{i=1}^{11}(\hat{y}'(x_i) + \hat{y}^2(x_i) - 1)^2 + \left(y(0)\right)^2 \quad (15)$$

GA has been employed to solve the minimization problem given by (15) and to achieve the unknown coefficients$(a_0, a_1, \ldots, a_7)$. The GA is executed according to the prescribed parameter values and settings given in Table 1. The number of unknown coefficients that need to be adapted is 8 therefore the size of chromosome for GA is also chosen to be 8. The optimal values of the unknown coefficients achieved by GA are given in Table 2 for various values of *n*.

Table 1. Parameter values and settings of GA

Parameter Name	Parameter Settings/Value
Population size	[100 100]
Chromosome size	6, 7, 8 (for n = 5, 6, 7)
No. of generations	1000
Selection function	Stochastic uniform
Mutation function	Adaptive feasible
Crossover function	Heuristic
Crossover fraction	0.8

(DOI: dx.doi.org/14.9831/1444-8939.2014/2-6/MAGNT.6)

Table 2. Values of unknown coefficients obtained by GA for example 1

	Value			
	$x \in [0,1]$			$x \in [0,5]$
Coefficients	$n = 5$	$n = 6$	$n = 7$	$n = 7$
a_0	0.000001	0.000000	0.000000	-0.000007
a_1	0.199932	0.166663	0.142857	0.151233
a_2	0.401322	0.333444	0.285690	0.295511
a_3	0.562517	0.483054	0.419155	0.124550
a_4	0.677527	0.600108	0.532988	0.258568
a_5	0.761581	0.691610	0.626355	0.193401
a_6	---	0.761600	0.701599	0.211702
a_7	---	---	0.761595	0.209841

Once the unknown coefficients have been found, the numerical solution of (12) can be obtained at any value of *x* in the interval [0, 1] by using the values of these coefficients in (14). The approximate numerical results obtained by the proposed method are provided in Table 3, also exact solution and the results obtained by other methods including OHAM [1], MHPM [3], ADM [4], VIM [5], MVIM [6], BPCM [23], as well as PSO-NN method [15] are given for comparison. Moreover in Table 4, comparison of absolute errors between the proposed method and the methods used in [1,3-6,15,23] is made. From the comparison of Table 3 and Table 4, the effectiveness of the proposed method is evident. The absolute errors yielded by the proposed method are found to be comparatively smaller than OHAM [1], MHPM [3], ADM [4], VIM [5], and PSO-NN method [15], which confirms the accuracy of the proposed method. It is seen from the comparison that the classical methods like ADM and VIM give better approximation initially but as the value of *x* increases their accuracy also decreases, whereas the accuracy of the proposed scheme is seen fairly steady in the solution domain of *x*.

Moreover in Table 5 comparison of our results with $n = 5$ (B-polynomials of degree 5) is made with Bernstein polynomials based collocation method (BPCM) [23] with $n = 5$. From the comparison the results are found quite comparable to BPCM, which further confirms the efficacy of proposed method.

Table 3 Comparison of numerical results for example 1

		Proposed method $\hat{y}(x)$ with (n=7)	**Other methods**			
x	$y_{exact}\ (x)$	**GA**	**ADM**	**VIM**	**OHAM**	**PSO-NN**
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
0.1	0.099668	0.099667	0.099668	0.099668	0.099668	0.099665
0.2	0.197375	0.197374	0.197375	0.197376	0.197377	0.197400
0.3	0.291313	0.291311	0.291313	0.291321	0.291315	0.291312
0.4	0.379949	0.379947	0.379949	0.380009	0.379953	0.379954
0.5	0.462117	0.462115	0.462121	0.462401	0.462120	0.462081
0.6	0.53705	0.537049	0.537078	0.53805	0.537051	0.537021
0.7	0.604368	0.604368	0.604514	0.607262	0.604369	0.60439
0.8	0.664037	0.664036	0.664641	0.67129	0.664038	0.66370
0.9	0.716298	0.716297	0.718392	0.732603	0.716300	----
1.0	0.761594	0.761595	0.767901	0.795262	0.761595	0.761596

(DOI: dx.doi.org/14.9831/1444-8939.2014/2-6/MAGNT.6)

Table 4 Comparison of absolute errors for example 1

x	GA	ADM	VIM	OHAM	PSO-NN
0	1.66E-09	0	0	0	4.04E-09
0.1	1.17E-06	8.80E-14	1.50E-11	1.844E-07	2.99E-06
0.2	1.00E-06	1.79E-10	4.22E-09	1.180E-06	2.47E-05
0.3	1.22E-06	1.51E-08	1.56E-07	2.695E-06	6.12E-07
0.4	2.07E-06	3.49E-07	1.98E-06	3.540E-06	5.04E-06
0.5	1.95E-06	3.93E-06	1.38E-05	2.997E-06	3.62E-05
0.6	7.76E-07	2.81E-05	6.62E-05	1.747E-06	2.86E-05
0.7	1.38E-07	1.46E-04	2.43E-04	1.067E-06	2.22E-05
0.8	6.32E-07	6.05E-04	7.36E-04	1.234E-06	3.37E-04
0.9	5.63E-07	2.09E-03	1.92E-03	1.649E-06	-----
1.0	5.33E-07	6.31E-03	4.43E-03	9.540E-07	1.84E-06

Table 5 Comparison of numerical results with BPCM (with n = 5)

				Absolute Error	
x	$y_{exact}\ (x)$	**Proposed method $\hat{y}(x)$ (with n = 5)**	**BPCM (with n = 5)**	**Proposed method**	**BPCM**
0.0	0.000000	0.000001	0.000000	7.902E-07	0.000000
0.2	0.197375	0.197464	0.197427	8.855E-05	5.173E-05
0.4	0.379949	0.379956	0.379975	6.795E-06	2.597E-05
0.6	0.537050	0.537061	0.537090	1.100E-05	4.066E-05
0.8	0.664037	0.664100	0.664024	6.369E-05	1.239E-05
1.0	0.761594	0.761581	0.762346	1.312E-05	7.514E-04

The influence of the change in degree of B-polynomials i.e. change in n on the performance of the proposed scheme is analyzed next. We used $n = 5, 6$ in (7) for evaluating the performance, therefore the number of unknown coefficients to be tailored are 6 and 7 respectively for $n = 5$ and $n = 6$. GA has been used for solving the FF such as given by (15) and to achieve the unknown coefficients corresponding to each value of n mentioned. GA is implemented with the same parameter settings prescribed in Table 1 except a change in the chromosome size which is now chosen as 6 for $n = 5$ and 7 for $n = 6$ respectively.

The optimal values of the unknown coefficients acquired by GA are given in Tables 2 and using these values the approximate solution $\hat{y}(x)$ is obtained from (7). The approximate solutions obtained using the proposed method for different values of n = 5, 6, are presented in Table 6, also exact solution and solution with n = 7 obtained by the proposed method are given for comparison. From the comparison of Table 6 the improvement in the approximate solution is observed with the increase in n (i.e. increase in degree of B-polynomials).

To further investigate the effect of change in the degree of B-polynomials (i.e. change in n), a comparison of average absolute errors, computational time, and number of generations utilized by GA is given in Table 7. From the comparison of Table 7, it is observed that the accuracy of the solution drastically improves with the increase in n from 5 to 7, but at the cost of high computational time and more number of generations.

(DOI: dx.doi.org/14.9831/1444-8939.2014/2-6/MAGNT.6)

Table 6 Comparison of numerical results for various values of *n*

		Proposed method $\hat{y}(x)$			Absolute errors $\lvert(y_{exact} - \hat{y}(x))\rvert$		
x	$y_{exact}\,(x)$	*n = 5*	*n = 6*	*n = 7*	*n = 5*	*n = 6*	*n = 7*
0.0	0.000000	0.000001	0.000000	0.000000	7.90E-07	4.78E-07	1.66E-09
0.1	0.099668	0.099713	0.099673	0.099667	4.54E-05	5.43E-06	1.17E-06
0.2	0.197375	0.197464	0.197382	0.197374	8.85E-05	6.77E-06	1.00E-06
0.3	0.291313	0.291373	0.291313	0.291311	6.06E-05	6.05E-07	1.22E-06
0.4	0.379949	0.379956	0.379947	0.379947	6.79E-06	2.43E-06	2.07E-06
0.5	0.462117	0.462102	0.462119	0.462115	1.53E-05	1.64E-06	1.95E-06
0.6	0.537050	0.537061	0.537056	0.537049	1.10E-05	6.23E-06	7.76E-07
0.7	0.604368	0.604422	0.604371	0.604368	5.42E-05	3.52E-06	1.38E-07
0.8	0.664037	0.664100	0.664033	0.664036	6.37E-05	3.77E-06	6.32E-07
0.9	0.716298	0.716317	0.716295	0.716297	1.89E-05	2.37E-06	5.63E-07
1.0	0.761594	0.761581	0.761600	0.761595	1.31E-05	6.08E-06	5.33E-07

Table 7 Effect of change in the degree *n*

Degree of B-polynomials (*n*)	No. of Generations	Computational Time (sec)	Average Absolute Error
5	240	11	3.439E-05
6	281	15	3.575E-06
7	339	17	9.140E-07

We now investigate the reliability and accuracy of the proposed scheme in the larger interval $x \in [0, 5]$. The FF is formulated and GA is used for its minimization to achieve the values of the unknown coefficients. GA is executed with the same parameter settings as prescribed in Table 2. The values of unknown parameters achieved by GA are provided in Table 2. The comparison of our approximate solution is graphically made with two well-known classical methods ADM and VIM and the exact solution in Fig. 1. The comparison of results in Fig. 1 evidently shows that the proposed scheme is quite capable of yielding the numerical solution of the Riccati equation (12) with a significant accuracy in the larger domain of *x*, while ADM and VIM diverge after $x = 1.4$ and $x = 1.1$ respectively.

(DOI: dx.doi.org/14.9831/1444-8939.2014/2-6/MAGNT.6)

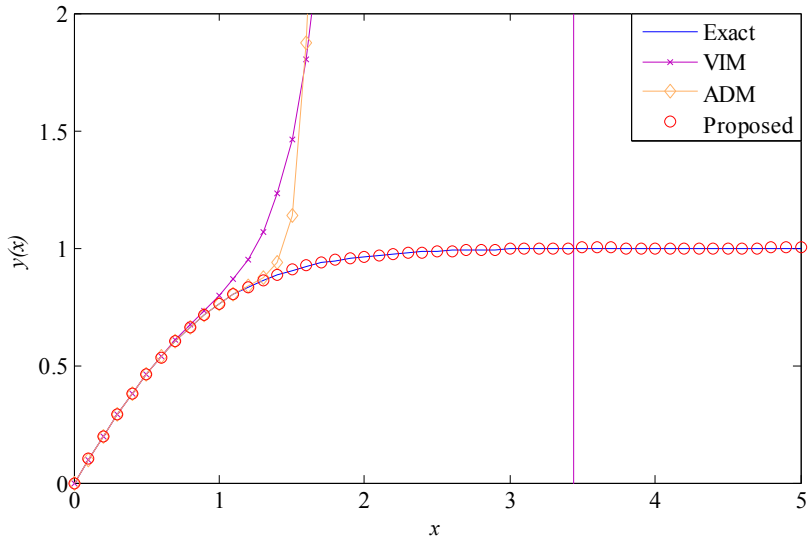

Fig. 1 Comparison of numerical solutions for $x \in [0,5]$

Example 2: We now consider another Riccati NODE as follows [1-6].

$$y'(x) = -y^2(x) + 2y(x) + 1, \quad y(0) = 0 \tag{16}$$

the exact solution is given by

$$y_{exact}(x) = 1 + \sqrt{2}tanh\left(\sqrt{2}t + \frac{1}{2}log\left(\frac{\sqrt{2}-1}{\sqrt{2}+1}\right)\right) \tag{17}$$

The approximate solution of (16) is obtained using the proposed scheme with $n = 5, 7$ (B-polynomial of degree 5, 7), in the interval $x \in (0,1)$ and $x \in (0,2)$ with a step of 0.1 for each interval of x. The FF is formulated for each interval of x separately, for instance FF for the interval $x \in (0,2)$ given by

$$\varepsilon_j = \frac{1}{21}\sum_{i=1}^{21}\left((\hat{y}'(x_i) + \hat{y}^2(x_i) - 2\hat{y}(x_i) - 1)\right)^2 + \left(y(o)\right)^2 \tag{18}$$

The FF given by (18) is solved using GA for finding the optimal values of unknown coefficients corresponding to the minimum ε_j. GA is implemented according to the prescribed parameter values and settings given in Table 1. The values of the unknown coefficients achieved by GA are provided in Table 8. Using the values of unknown coefficients in (7) yields the approximate numerical solution of (16) straightforward.

The numerical solutions obtained by the proposed method are presented in Table 9 for the $x \in (0,1)$ and in Fig. 2 for the $x \in (0,2)$, also the exact solutions and the approximate solutions reported by methods including OHAM [1], MHPM [3], VIM [5], MVIM [6], and BPCM [23] are given for the sake of comparison.

(DOI: dx.doi.org/14.9831/1444-8939.2014/2-6/MAGNT.6)

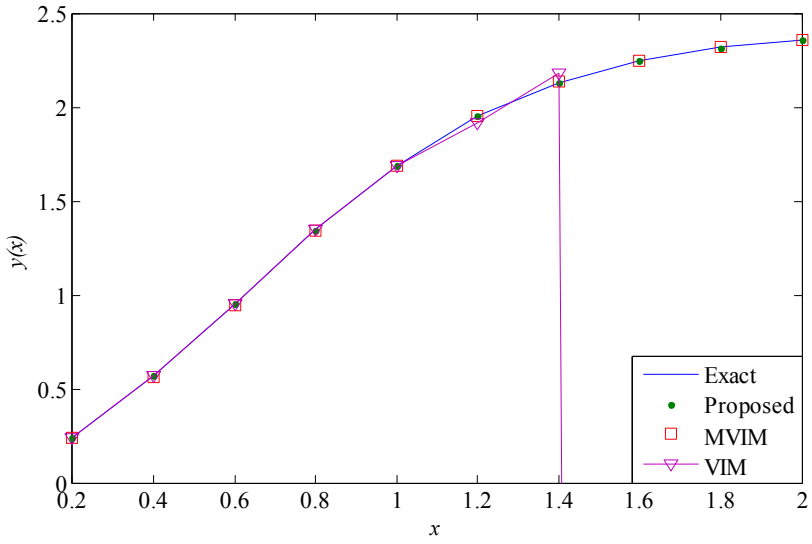

Fig. 4 Comparison of numerical solutions for $x \in (0, 2)$

Table 8 Optimal values of coefficients obtained by GA for example 2

	Value		
	$x \in [0, 1]$		$x \in [0, 2]$
Coefficient	$n = 5$	$n = 7$	$n = 7$
a_0	0.000076	0.000002	0.000330
a_1	0.200673	0.142854	0.289078
a_2	0.487779	0.334228	0.697442
a_3	0.969543	0.577790	1.725555
a_4	1.384721	0.893010	2.115500
a_5	1.689281	1.204383	2.201938
a_6		1.471732	2.315525
a_7		1.689531	2.358231

Further Table 10 and Table 11 show a comparison of absolute errors yielded by our method and other methods including OHAM [1], MHPM [3], VIM [5], MVIM [6], PSO-NN [15], and BPCM [23] in contrast to exact solutions. The comparison clearly reveals that the proposed method with $n = 7$ provides the solution with significantly smaller absolute errors as compared to OHAM, MHPM, and VIM. Moreover, it is noticeable that VIM gives good accuracy only in the short interval $x \in (0,1.4)$ after that it diverges drastically, while our method provides the approximate solution with a high degree of accuracy in the interval $x \in (0,2)$. Further, it is seen that the proposed scheme with $n = 7$ gives solutions that are quite comparable to stochastic solver PSO-NN, which also testifies the accuracy of the proposed scheme. The comparison further reveals that numerical results by our method with $n = 5$ are fairly comparable to those reported by BPCM with $n = 5$, which show the accuracy of the proposed method.

(DOI: dx.doi.org/14.9831/1444-8939.2014/2-6/MAGNT.6)

Table 9 Comparison of numerical results for example 2

		Proposed method $\hat{y}(x)$		**Other methods**				
x	$y_{exact}(x)$	$n = 5$	$n = 7$	**BPCM[23]** $n = 5$	**MHPM[3]**	**VIM[5]**	**OHAM[1]**	**PSO-NN[15]**
0	0	0.000076	0.000002	0	0	0.000000	0	6.58E-09
0.1	0.110295	0.109928	0.110350	---	0.110294	0.110295	0.110328	0.110328
0.2	0.241977	0.241161	0.242024	0.241283	0.241965	0.241977	0.242273	0.241997
0.3	0.395105	0.394495	0.395138	---	0.395106	0.395113	0.396178	0.395101
0.4	0.567812	0.567624	0.567875	0.567187	0.568115	0.567845	0.570281	0.567797
0.5	0.756014	0.755923	0.756097	---	0.757564	0.756086	0.759942	0.756008
0.6	0.953566	0.953149	0.953631	0.952653	0.958259	0.953666	0.957123	0.953580
0.7	1.152949	1.152145	1.152997	---	1.163459	1.153037	1.150521	1.152973
0.8	1.346364	1.345546	1.346429	1.345970	1.365240	1.346379	1.326366	1.346374
0.9	1.526911	1.526482	1.526980	---	1.554960	1.526411	1.468058	1.526890
1.0	1.689498	1.689281	1.689531	1.682362	1.723810	1.686027	1.546858	1.689459

Table 10 Comparison of absolute errors for example 2 (for $x \in [0, 1]$)

	Proposed Method		**Other Methods**				
x	$n = 5$	$n = 7$	**BPCM** $n = 5$	**MHPM**	**VIM**	**OHAM**	**PSO-NN**
0	7.58E-05	2.04E-06	0	0	0	0	6.58E-09
0.1	3.67E-04	5.48E-05	---	1.20E-06	1.97E-07	3.24E-05	3.28E-05
0.2	8.16E-04	4.68E-05	6.94E-04	1.18E-05	2.00E-07	2.97E-04	2.02E-05
0.3	6.10E-04	3.36E-05	---	1.15E-06	8.15E-06	1.07E-03	3.85E-06
0.4	1.88E-04	6.24E-05	6.25E-04	3.03E-04	3.28E-05	2.47E-03	1.52E-05
0.5	9.09E-05	8.30E-05	---	1.55E-03	7.16E-05	3.93E-03	6.39E-06
0.6	4.17E-04	6.47E-05	9.13E-04	4.69E-03	9.98E-05	3.56E-03	1.38E-05
0.7	8.04E-04	4.77E-05	---	1.05E-02	8.80E-05	2.43E-03	2.40E-05
0.8	8.17E-04	6.51E-05	3.94E-04	1.89E-02	1.53E-05	2.00E-02	1.03E-05
0.9	4.29E-04	6.90E-05	---	2.80E-02	5.00E-04	5.89E-02	2.13E-05
1.0	2.18E-04	1.16E-05	7.13E-03	3.43E-02	3.47E-03	1.43E-01	3.94E-05

Table 11 Comparison of absolute errors for example 2 (for $x \in [0, 2]$)

x	**Proposed (**with $n = 7$**)**	**MVIM [6]**	**VIM [5]**
0.2	2.38E-03	2.362E-03	1.033E-06
0.4	7.93E-04	5.189E-03	3.335E-05
0.6	5.03E-04	6.725E-03	9.982E-05
0.8	1.85E-03	5.800E-03	1.545E-05
1.0	2.66E-03	3.116E-03	3.471E-03
1.2	4.83E-04	4.109E-04	3.631E-02
1.4	6.76E-04	1.256E-03	4.780E-02
1.6	6.41E-04	1.855E-03	5.323E+01
1.8	8.45E-04	1.799E-03	5.341E+03
2.0	4.59E-04	1.470E-03	2.864E+05

(DOI: dx.doi.org/14.9831/1444-8939.2014/2-6/MAGNT.6)

4. Conclusions and Future Work

A novel stochastic technique based on hybrid approach of Bernstein polynomials and evolutionary algorithms GA and DE has been presented for numerical solution of Riccati equations. The effectiveness of the proposed technique has been demonstrated by numerically solving different forms of Riccai nonlinear differential equation. On the basis of the comparisons made with the exact solutions and some classical approximate numerical methods, as well as stochastic solver, it can be concluded that the proposed method is handy and viable for solving the Riccati nonlinear differential equations. Moreover, the presented method has shown supremacy on some of the well known classical methods like VIM, MHPM, ADM and OHAM in terms of accuracy.

In future we seek to use the proposed methodology in combination with other nature inspired computation and memetic computing approach for solving other such nonlinear ODEs and coupled systems of ODEs arising in engineering and science.

References

1. Mabood F, Izani bin Md Ismail A, Hashim I. Application of optimal homotopy Asymptotic method for the approximate solution of Riccati equation. Sains Malaysiana 2013; 42(6): 863–867.
2. Abbasbandy S. Homotopy perturbation method for quadratic Riccati differential equation and comparison with Adomian's decomposition method. Appl Math Comput 2006; 172: 485–490.
3. Odiba Z, Momani S. Modified homotopy perturbation method: application to quadratic Riccati differential equation of fractional order. Chaos, Solitons and Fractals 2008; 36:167–174.
4. Momani S, Shawagfeh N. Decomposition method for solving fractional Riccati differential equations. Appl Math Comput 2006; 182:1083–1092.
5. Batiha B, Noorani MSM, Hashim I. Application of variational iteration method to a general Riccati equation. Int Math Forum 2007; 2(56): 2759 – 2770.
6. Batiha B. A numeric-analytic method for approximating quadratic Riccati differential equation. Int J Appl Math Res 2012; 1 (1): 8-16
7. Taiwo OA, Osilagun JA. Approximate solution of generalized Riccati differential equations by iterative decomposition algorithm. J Res Recent Trends 2012; 20-27.
8. Geng FZ, Li XM. A new method for Riccati differential equations Based on reproducing kernel and quasilinearization methods. Abstr Appl Anal 2012; doi: 10.1155/2012/603748
9. Azimzadeh Z, Babolian E, Vahidi AR. A comparative study of mumerical methods for solving the Riccati equation. World Appl Sci J 2012; 16 (2): 227-231.
10. Chaharpashlou R, Ghorbani A. Parametric iteration method: a useful analytic method for solving Riccati differential equations, Stud Nonlinear Sci 2012; 3(2):78-85.
11. Abbasbandy S. A new application of He's variational iteration method for quadratic Riccati differential equation by using Adomian's polynomials. J Comput Appl Math 2007; 207:59-63
12. Tan Y, Abbasbandy S. Homotopy analysis method for quadratic Riccati differential equation. Comm Nonlinear Sci and Numer Simul 2008; 13:539–546
13. Khan NA, Asmat Ara A, Khan NA. Fractional-order Riccati differential equation: analytical approximation and numerical results. Advances Diff equations 2013;185.
14. Magdy AEl-Tawil, Bahnasawi AA, Abdel-Naby A. Solving Riccati differential equation using Adomian's decomposition method. Appl Math Comput 2004; 157:503–514.
15. Raja MAZ, Khan JA, Qureshi IM. A new stochastic approach for solution of Riccati differential equation of fractional order Ann Math Artif Intell 2011;doi:10.1007/s10472-010-9222-x.
16. Malik SA, Qureshi IM, Zubair M, Haq I. Solution to force-free and forced duffing-van der pol oscillator using memetic computing,
J Basic Appl Sci Res 2012; 2(11):11136-11148.

(DOI: dx.doi.org/14.9831/1444-8939.2014/2-6/MAGNT.6)

17. Malik SA, Qureshi IM, Amir M, Haq I. Nature inspired computational technique for the numerical solution of nonlinear singular boundary value problems arising in physiology. The Sci World J 2014; doi:10.1155/2014/837021
18. Malik SA**,** Qureshi IM, Zubair M, Amir M. Hybrid heuristic computational approach to the Bratu problem. Res J Recent Sci 2013; 2(10):1-8.
20. Malik SA, Qureshi IM, Amir M, Haq I. Numerical solution to nonlinear biochemical reaction model using hybrid polynomial basis differential evolution technique. Advanced Stud Bio 2014;6(3): 99-113.
20. Khan JA, Raja RMA, Qureshi IM. Swarm intelligence for the problems of non-linear ordinary differential equations and its application to well known Wessinger's equation. European J Sci Res 2009; 34(4): 514-525.
21. Caetano C, Reis Jr JL, Amorim J,3 Lemes MR, Pino Jr AD. Using neural networks to solve nonlinear differential equations in atomic and molecular physics. Int J Quantum Chem 2011; 111: 2732–2740.
22. Raja MAZ, Samar R. Numerical treatment of nonlinear MHD Jeffery-Hamel problem using neural networks optimized with interior point algorithm. Neurocomuting 2014;124: 178-193.
23. Yüzbaş Ş. Numerical solutions of fractional Riccati type differential equations by means of the Bernstein polynomials. Appl Math Comput 2013; 219: 6328–6343.
24. Bhatti MI, Bracken P. Solutions of differential equations in a Bernstein polynomial basis. J Comput Appl Math 2007; 205: 272 – 280.
25. Ordokhani Y, Far SD. Approximate solutions of differential equations by using the Bernstein polynomials. ISRN Appl.Math.2011;doi:10.5402/2011/787694.
26. Baleanu D, Alipour M, Jafari H, The Bernstein operational matrices for solving the fractional quadratic Riccati differential equations with the Riemann-Liouville derivative. Abstr Appl Anal 2013; doi10.1155/2013/461970.
27. Pandya1 BM, Joshi DC. Solution of a volterra's population model in a Bernstein polynomial basis. Appl Math Sci 2011; 5 (69): 3403 – 3410.
28. Dascioglu AA, Isler N. Bernstein collocation method for solving nonlinear differential equations. Math Comput Appl 2013; 18(3):293-300.
29. Mitchell M. Genetic algorithms: an overview. Complexity 1995; 1 (1): 31-39.

(DOI: dx.doi.org/14.9831/1444-8939.2014/2-6/MAGNT.6)