Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2011, Article ID 675075, 18 pages
doi:10.1155/2011/675075

Research Article

Solution of Fractional Order System of Bagley-Torvik Equation Using Evolutionary Computational Intelligence

Muhammad Asif Zahoor Raja,[1] Junaid Ali Khan,[1] and Ijaz Mansoor Qureshi[2]

[1] *Department of Electronics Engineering, Faculty of Engineering and Technology, International Islamic University, Sector H-10, Islamabad 44000, Pakistan*
[2] *Department of Electronics Engineering, Air University, Sector E-9, Islamabad 44000, Pakistan*

Correspondence should be addressed to Muhammad Asif Zahoor Raja, asif.phdee10@iiu.edu.pk

Received 10 June 2010; Revised 12 January 2011; Accepted 23 January 2011

Academic Editor: Jyh Horng Chou

A stochastic technique has been developed for the solution of fractional order system represented by Bagley-Torvik equation. The mathematical model of the equation was developed with the help of feed-forward artificial neural networks. The training of the networks was made with evolutionary computational intelligence based on genetic algorithm hybrid with pattern search technique. Designed scheme was successfully applied to different forms of the equation. Results are compared with standard approximate analytic, stochastic numerical solvers and exact solutions.

1. Introduction

The Bagley-Torvik equation is originally formulated in the studies on behavior of real material by use of fractional calculus [1, 2]. 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 [3, 4]. Fractional dynamic systems have found many applications in various problems such as viscoelasticity, heat conduction, electrode-electrolyte polarization, electromagnetic waves, diffusion wave, control theory, and signal processing [1–10].

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

$$A\frac{d^2y(t)}{dt^2} + B\frac{d^{3/2}y(t)}{dt^{3/2}} + C\left[y(t)\right]^n = f(t), \quad 0 < t \leq T \tag{1.1}$$

with initial conditions given as

$$\frac{d^k}{dt^k}y(0) = c_k, \quad k = 0, 1 \tag{1.2}$$

whereas boundary condition at $t = t_0$, for $0 < t_0 \leq T$, is written as

$$\frac{d^k}{dt^k}y(t_0) = b_k, \quad k = 0, 1, \tag{1.3}$$

where n is the nonlinear operator of the equations, $y(t)$ is the solution of the equation, A, B, and C are constant coefficients, T is the constant representing the span of inputs within the close interval $[0, T]$, and c_k, b_k are the constants.

The general response expression (1.1) 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, μ is viscosity, and ρ is the fluid density, then (1.1) represents the motion of a large thin plate in a Newtonian fluid [3]. Similarly, linearly damped fractional oscillator with the damping term has a fractional derivative of order $\nu = 1.5$, and it can be represented by Bagley-Torvik [3, 11].

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 [12, 13], He's variational iteration method [14], Taylor collocation method [15]. Diethelm transformed the equation into first-order coupled fractional differential equation and solved the problem with Adams predictor and corrector approach [16]. Podlubny used successive approximation method to solve the equation and recently applied the Matrix Approach to Discretization of Fractional Derivatives for the same problem [3, 17]. However, there has been no advancement to apply the stochastic numerical solvers to find the solution of the equation.

In this paper, investigation and analysis are carried out for successful modeling of the equation using feed-forward artificial neural networks (ANNs). The linear combination of these networks is defined by an unsupervised error. This error is minimized with the help of appropriate unknown parameter, that is, weights. The training of weights is conducted with stochastic optimization techniques based on genetic algorithm hybridized with pattern search technique. It is well known that these techniques are reliable, successful, and efficient to avoid the possibility to get stuck in local minima or diverge to unstable situations and also maintain the diversity throughout [18, 19].

2. Basic Definition

In this section, some definitions and relations are given, which will be used in the proceeding sections. The definitions of fractional integral and derivative have been provided in the fractional calculus literature in a variety of ways, including Riemann-Liouville, Caputo, Erdélyi-Kober, Hadamard, Grünwald-Letnikov, and Riesz type. Equivalence of these definitions on some function has also been established [20–22]. However, in this paper, Riemann-Liouville definition for fractional derivative is used with lower terminal at *0*.

Definition 2.1 (Riemann-Liouville fractional order derivatives). The Riemann-Liouville derivative of ${}^{\mathrm{RL}}D^{\nu}$ of fractional order ν is normally provided in the literature as [3, 22]

$$ {}^{\mathrm{RL}}D^{\nu} f(t) = \frac{1}{\Gamma(m-\nu)} \frac{d^m}{dt^m} \int_0^t \frac{f(\tau)}{(t-\tau)^{1+\nu-m}} d\tau \quad (m-1 < \nu \le m), \tag{2.1} $$

where $\nu \in \mathcal{R}$, $m \in N$, f is the continuous function, and $\Gamma(x)$ is the gamma function defined by

$$ \Gamma(x) = \int_0^{\infty} e^{-t} t^{x-1} dt, \qquad (\mathcal{R}(x) > 0). \tag{2.2} $$

Definition 2.2 (Mittag-Leffler function). The Mittag-Leffler function (MLF) is one of the main functions that has widespread use in the field of fractional calculus. Specially, its importance is realized in providing the analytic solution for differential equation of fractional order.

The definition of classical MLF function in two parameters α and β is given as [3, 23]

$$ E_{\alpha,\beta}(t) = \sum_{k=0}^{\infty} \frac{t^k}{\Gamma(\alpha k + \beta)} \quad (\alpha > 0,\ \beta > 0). \tag{2.3} $$

It reduces to standard MLF function of one parameter by taking $\beta = 1$.

3. Mathematical Modeling

In this section mathematical modeling of Bagley-Torvik equations with feed-forward artificial neural network is presented.

The solution y of the fractional differential equation along with its ν arbitrary order derivative $d^{\nu}y/dt^{\nu}$ can be approximated by the following continuous mapping as in neural network methodology [24–26]:

$$ \begin{aligned} \widehat{y}(t) &= \sum_{i=1}^{m} \alpha_i g(w_i t + \beta_i), \\ \frac{d^{\nu}\widehat{y}}{dt^{\nu}} &= \sum_{i=1}^{m} \alpha_i \frac{d^{\nu}}{dt^{\nu}} g(w_i t + \beta_i), \end{aligned} \tag{3.1} $$

where α_i, w_i, and β_i are bounded real-valued adaptive parameters, m is the number of neurons, and f is the activation function taken as exponential function instead of normally

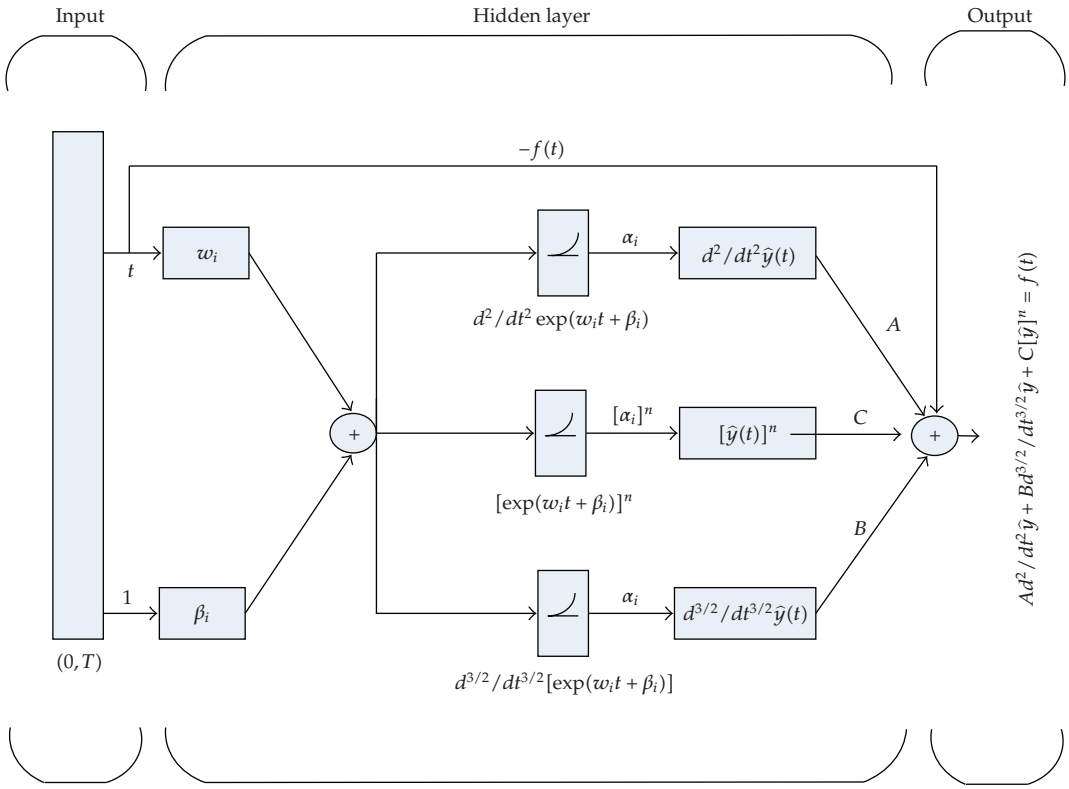

Figure 1: FDE-NN architecture of Bagley-Torvik equation.

taken log-sigmoid due to nonavailability of its fractional derivative. Fractional differential equation neural networks (FDE-NNs) for expression (1.1) can be formulated as

$$\widehat{y}(t) = \sum_{i=1}^{m} \alpha_i e^{w_i t + \beta_i}, \tag{3.2}$$

$$\frac{d^2}{dt^2}\widehat{y}(t) = \sum_{i=1}^{m} \alpha_i {w_i}^2 e^{w_i t + \beta_i}, \tag{3.3}$$

$$\frac{d^{3/2}}{dx^{3/2}}\widehat{y}(t) = \sum_{i=1}^{m} \alpha_i e^{\beta_i} t^{-3/2} E_{1,-1/2}(w_i t). \tag{3.4}$$

The mathematical model for (1.1) can be the linear combinations of the networks represented above. The FDE-NN architecture formulated for Bagley-Torvik equation can be seen in Figure 1. It is clear that the solution y can be approximated with $\widehat{y}$ subject to finding appropriate unknown weights.

4. Evolutionary Computational Intelligence

Evolutionary computational intelligence uses natural evolution as an optimization mechanism for solving various problems. The main objective of the scheme is to find a good solution

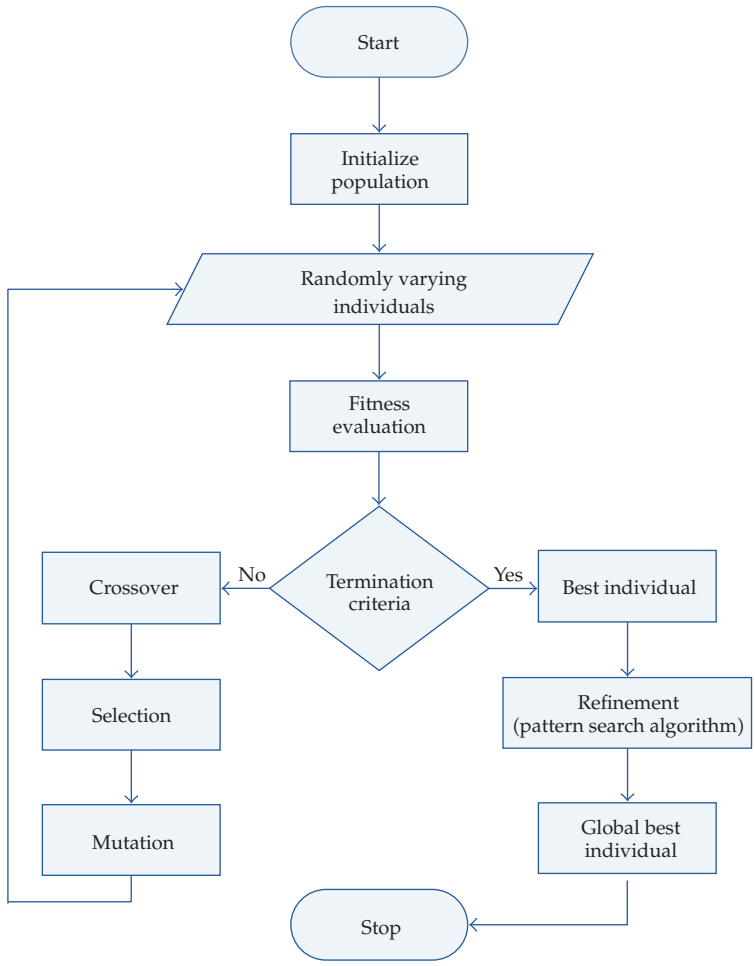

Figure 2: Generic flow diagram of evolutionary computation algorithm.

to a problem in a large search space of candidate solutions. In this section, the methodology for learning of the unknown weights of networks is given. Our intent is to use evolutionary computation based on Genetic Algorithm (GA) hybrid with Pattern Search technique (PS) to solve such problem. The main advantages of the GA algorithm are robustness in controlling parameters, not to get stuck in local minima, avoid divergence and efficient compared to other mathematical algorithms and heuristic optimization techniques [18, 19].

Pattern search algorithm is a kind of aggressive optimization method, belonging to class of direct search method [27, 28]. It can find the optimum values using stochastic searching technology based on scaled and translated integer lattice. Pattern search algorithm has a strong ability to find the local optimistic results [29, 30].

So, after the detailed study of the literature about evolutionary computation techniques [31, 32], GAs hybrid with PS algorithm are used by seeing its strengths and applicability. The general flowchart showing the process of evolutionary algorithm is given in Figure 2.

Randomly generated initial population consists of finite set of chromosomes or individual, and each chromosome consists of as many numbers of genes as the number of

unknown parameters, that is, weights in the neural networks representing the equation. The fitness of the individual is calculated based on an unsupervised error defined by linear combination of the differential equation neural networks. The fitness function to be minimized is defined as the mean of sum of square errors

$$e_j = e_1^j + e_2^j \quad j = 1, 2, \ldots, \tag{4.1}$$

where j is the generation index and e_1^j is given as

$$e_1^j = \frac{1}{s}\sum_{i=1}^{s}\left(A\frac{d^2\widehat{y}(t_i)}{dt^2} + B\frac{d^{3/2}\widehat{y}(t_i)}{dt^{3/2}} + C\left[\widehat{y}(t_i)\right]^n - f(t_i)\right)\Bigg|_j, \tag{4.2}$$

where s is the total number of steps, each step is randomly taken between $(0, T)$. The greater the value of s the more will be the accuracy, but increase, in computational complexity of the algorithm.

Similarly, e_2^j, linked with initial and boundary conditions is finally written as

$$e_2^j = \frac{1}{2}\sum_{k=0}^{1}\left[\left(\frac{d^k}{dt^k}\widehat{y}(0) - c_k\right)^2 + \left(\frac{d^k}{dt^k}\widehat{y}(t_0) - b_k\right)^2\right]\Bigg|_j. \tag{4.3}$$

It is quite evident that the weights for which fitness function, e_j, approaches zero, the solution $y(t)$ of the equation is well approximate by $\widehat{y}(t)$ given in (3.2).

Evolutionary algorithm is given in the following steps.

Step 1 (initialized population). Randomly generate bounded real values to form initial population with N number of the chromosomes or individuals. Create M subpopulation each with N/M individuals. Each individual consists of as many number of genes as the number of unknown parameter in neural network. The better search space of algorithm is subjected to enough spread of initial population.

Step 2 (initialization). Following parameter values are initialized for algorithm execution. Set the number of variable equivalent to element in the individual. Set the number of generations and the fitness limit. Set the elite count "2" and value of crossover fraction as 0.80 for reproduction. Set migration in both forward and backward direction. Start generation count, and so forth.

Step 3 (fitness evaluation). Calculate the fitness for each individual using the expressions (4.1) to (4.3).

Step 4 (ranking). Rank each individual of the populations on the basis of minimum fitness values.

Step 5 (termination criteria). Terminate the algorithm if either predefined fitness value, that is, MSE 10^{-4} is achieved or number of generation is complete. If termination criterion fulfilled, then go to Step 8, else continue.

Step 6 (reproduction). Create the next generation using. Crossover: Call for scattered function, Mutation: Call for Adaptive Feasible function, Selection: Call for Stochastic Uniform function, and Elitism. Repeat the procedure from Step 3 to Step 6 until total number of generation are complete.

Step 7. Store the global best individual of this run. Repeat the Steps 2 to 6 to have sufficient numbers of global best individual for better statistical analysis.

Step 8 (refinement). Pattern search algorithm is used for further fine-tuning by taking the best individual of GAs as start point of the algorithm. MATLAB Genetic algorithm and direct search tool box are used for pattern search algorithm. Store the refined global best individual for statistical analysis later.

5. Simulation and Results

In this section, the simulation results are presented for three different fractional order systems represented by Bagley-Torvik equation. In order to prove the applicability of the designed scheme for such systems, we have considered the equation in example 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 example II, we have tested the proposed methodology on Bagley-Torvik equation for which PMA technique fails to determine the results due to nonhomogeneous initial conditions. In example III, the strength and weakness of our scheme are analyzed by taking such equation for which the exact solution is not known; however, its numerical solution is available.

5.1. Example I

Consider the Bagley-Torvik equation [14, 33]

$$\frac{d^2y(t)}{dt^2} + \frac{d^{3/2}y(t)}{dt^{3/2}} + y(t) = 2 + 4\sqrt{t/\pi} + t^2, \quad 0 < t \leq 1, \tag{5.1}$$

subjected to the initial condition and boundary condition as

$$y(0) = \frac{d}{dt}y(0) = 0, \qquad y(1) = 1, \qquad \frac{d}{dt}y(1) = 2. \tag{5.2}$$

The exact solution of the equation is given as

$$y(t) = t^2. \tag{5.3}$$

Mathematical modeling of the above equation is done with Fractional differential equation neural network (FDE-NN) represented by (3.2) to (3.4) by taking 10 number of neurons resulting in 30 number 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

Table 1: 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

of FDE-NN. Input of the training set is taken from $t \in (0, 1)$. Therefore the fitness function is formulated as

$$e_j = \frac{1}{2}\sum_{i=1}^{2}\left[\frac{d^2\widehat{y}(t_i)}{dt_i^2} + \frac{d^{3/2}\widehat{y}(t_i)}{dt_i^{3/2}} + \widehat{y}(t_i) - 2 - 4\sqrt{\frac{t_i}{\pi}} - t_i^2\right]^2 + \frac{1}{4}\left\{\left[\widehat{y}(0)\right]^2 + \left[\widehat{y}(1) - 1\right]^2 + \left[\widehat{y}'(0)\right]^2 + \left[\widehat{y}'(1) - 2\right]^2\right\}\Bigg|_j \quad j = 1, 2, 3, \ldots, \tag{5.4}$$

where j is the number of generations, $\widehat{y}$, $d^2\widehat{y}/dt^2$, and $d^{3/2}\widehat{y}/dt^{3/2}$ are networks provided in (3.2), (3.3), and (3.4), respectively. The scheme runs iteratively in order to compute the minimum of fitness function, e_j, with a termination criteria as 3000 number of generations executed or value of the fitness function $e_j \leq 10^{-4}$ whichever comes earlier. The best individual found by global search technique is passed to a rapid local search method as a start point for further refinements of the results. The parameter settings used in genetic algorithm (GA) and Pattern Search algorithm (PS) are provided in Table 1. One of the best sets of weights of FDE-NN learned stochastically by PS, GA, and Genetic algorithm hybrid with PS (GA-PS) algorithms is provided in Table 2. Using these weights in (3.2) one can find the solution to this problem for any input time t between 0 and 1. The solutions obtained from the chromosomes given in Table 2 are presented in Table 3.

The results of other numerical solvers like Podlubny matrix approach (PMA) [3, 17] and reported results of He's variational iteration method (VIM) [14] for inputs between 0 and 1 with a step of *0.1* are also provided in Table 2. In order to determine the results by PMA technique, the library functions provided by Podlubny at MATLAB central file exchange are used [34]. Following parameter setting is employed for PMA technique as follows:

(i) the value of fractional order derivative $\nu = 1.5$;

(ii) set the value for constant coefficients, $A = B = C = 1$;

(iii) the discretization step is taken as $h = 0.01$;

(iv) total number of time steps = 100.

Table 2: FDE-NN weights trained by different solvers for example I.

i	w_i			α_i			β_i		
	PS	GA	GA-PS	PS	GA	GA-PS	PS	GA	GA-PS
1	0.227249	0.014313	0.076813	0.731960	−1.16770	−1.16770	0.769316	0.046808	0.042905
2	0.031563	−0.15713	−0.15713	0.708549	0.554705	0.304705	0.871064	−0.73877	−0.73877
3	0.121262	−0.29925	3.669499	−7.73928	−0.16469	−0.16469	0.504503	−0.12463	−8.12463
4	2.583099	0.000465	0.000465	−4.61892	−0.74905	−0.74905	−4.77791	0.050833	0.050833
5	0.564499	1.169105	1.169105	0.941493	−0.14210	−0.14210	0.702424	−0.00566	−0.00566
6	0.659389	−1.30293	−1.30293	0.964177	0.334789	0.209789	0.748012	0.248267	0.248252
7	0.434814	−0.40253	−0.90253	0.988058	0.284607	0.284607	0.199868	0.648494	0.648494
8	−0.78165	0.906189	−5.09381	0.711952	−0.07749	3.922500	0.918857	0.372823	−7.62717
9	−0.19301	0.348033	−4.64415	0.595146	0.076874	0.014374	1.270213	−1.37687	−9.37687
10	0.192005	0.975022	0.943742	0.223502	0.271679	0.271679	0.873072	1.502332	1.502332

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

Moreover, the behavior of the derivative of the solution is also analyzed with the same individual as given in Table 2. 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. It is quite evident that the accuracy level lies in the range of 10^{-2} to 10^{-3}.

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 Mittag-Leffler function. Its generic form is given in expression (2.3). 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 [35].

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. 125 total independents runs are executed for each stochastic optimizer. These optimizers are PS, GA, and GA hybrid with PS (GA-PS). The values of the absolute error are provided in Table 5 for input at 0 and 1. Mean and Standard deviation (STD) were also calculated for the best 100 runs. The value of unsupervised error, e_j, of the equation in ascending order is plotted against each independent run of the algorithm in Figure 3. Moreover, the values of the absolute error for the solution as well as its derivative at 0 and 1 are plotted in Figure 4. It can be seen from Figures 3 and 4 and Table 5 that the best results are obtained by the use of GA-PS algorithm.

5.2. Example II

Another fractional order system of Bagley-Torvik equation is taken to investigate the strength and weaknesses of the proposed stochastic algorithm. Consider the Bagley-Torvik equation [13, 15]

$$A\frac{d^2y(t)}{dt^2} + B\frac{d^{3/2}y(t)}{dt^{3/2}} + Cy(t) = C(t+1), \quad 0 < t \leq 1 \tag{5.5}$$

Table 3: Comparison of results for the solution of example I.

T	$y(t)$	$\widehat{y}(t)$					$\lvert y(t) - \widehat{y}(t)\rvert$		
		PS	GA	GA-PS	PMA	VIM	GA-PS	PMA	VIM
0.0	0.00	2.73e-02	0.06730	0.03346	0.00000	0.00000	3.34e-2	0.00000	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

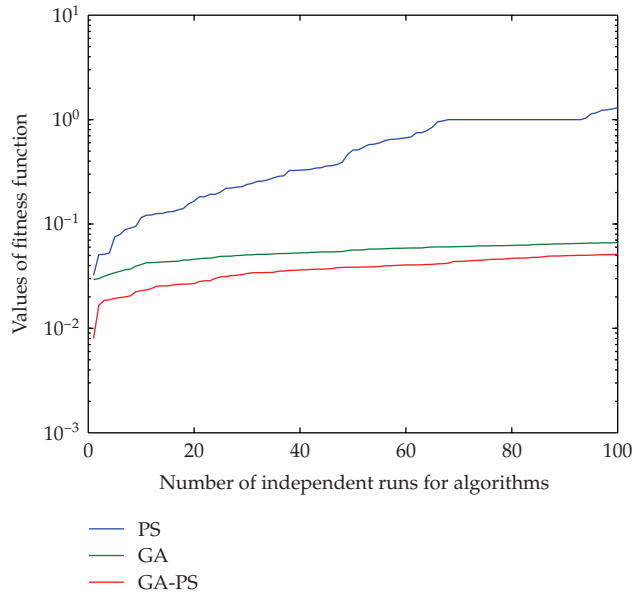

Figure 3: Comparison of different stochastic numerical solvers.

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

$$y(t) = t + 1. \tag{5.6}$$

This problem can be made simpler by use of following substitution:

$$Y(t) = y(t) - 1 - t. \tag{5.7}$$

Table 4: Comparison of results for derivative of the solution of example I.

T	$\widehat{y}'(t)$	$\widehat{y}'(t)$			$\lvert y'(t) - \widehat{y}'(t)\rvert$		
		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

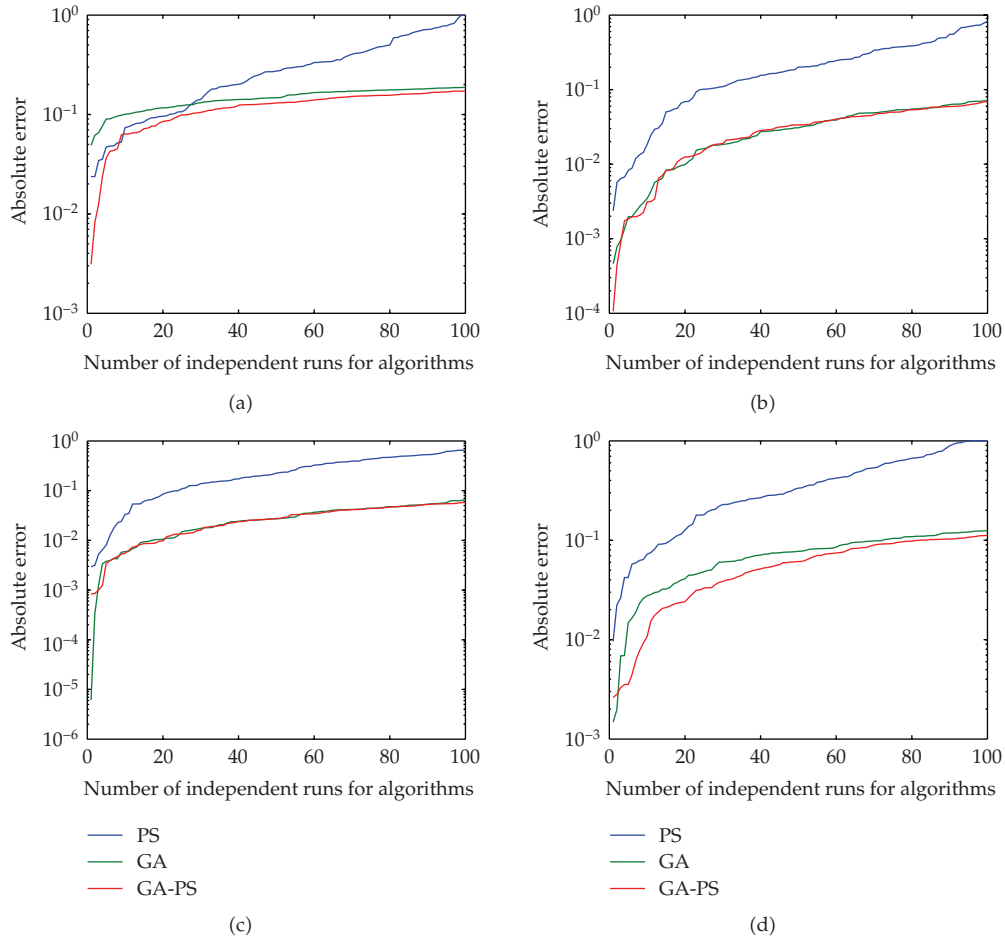

Figure 4: Comparison of FDE-NN for different stochastic numerical solvers, (a) and (b) are for the value of $|y(t) - \widehat{y}(t)|$ at $t = 0$ and $t = 1$ respectively, while (c) and (d) are for the value of $|\dot{y}(t) - \widehat{y}'(t)|$ at $t = 0$ and $t = 1$, respectively.

Table 5: Statistical analysis based on the value of absolute unsupervised error for example I.

	T	0			1		
	FDE-NN	PS	GA	GA-PS	PS	GA	GA-PS
Best	$\lvert y(t) - \widehat{y}(t)\rvert$	2.36e-2	4.87e-2	3.11e-3	2.35e-3	4.61e-4	1.06e-4
	$\lvert \dot{y}(t) - \widehat{y}'(t)\rvert$	2.91e-2	6.13e-6	8.31e-4	9.59e-3	1.48e-3	2.62e-2
Worst	$\lvert y(t) - \widehat{y}(t)\rvert$	1.00232	0.18695	0.17193	0.82603	7.19e-2	6.87e-2
	$\lvert \dot{y}(t) - \widehat{y}'(t)\rvert$	0.65725	6.63e-2	5.82e-2	1.00357	0.12491	0.11201
Mean	$\lvert y(t) - \widehat{y}(t)\rvert$	0.32789	0.14635	0.12111	0.24834	3.36e-2	3.31e-2
	$\lvert \dot{y}(t) - \widehat{y}'(t)\rvert$	0.27233	2.99e-2	2.90e-2	0.40912	7.54e-2	6.17e-2
STD	$\lvert y(t) - \widehat{y}(t)\rvert$	0.24679	3.25e-2	4.14e-2	0.20391	2.12e-2	2.00e-2
	$\lvert \dot{y}(t) - \widehat{y}'(t)\rvert$	0.19051	1.78e-2	1.70e-2	0.28778	3.35e-2	3.37e-2

Table 6: The weights trained for neural network modeling for example II.

I	w_i			α_i			β_i		
	PS	GA	GA-PS	PS	GA	GA-PS	PS	GA	GA-PS
1	−7.5259	0.45094	0.45094	0.46874	−0.6587	−0.6587	0.769316	−1.1923	0.48045
2	−7.7316	−0.0968	−0.0968	0.90230	−0.5674	−0.5674	0.871064	0.67735	0.84448
3	−5.9519	−0.3544	−0.3544	−6.6940	0.51534	0.51534	0.504503	−0.7461	0.13803
4	−1.5897	1.75859	1.75884	0.43284	−0.1467	−0.1467	−4.77791	0.33795	0.57994
5	0.20093	−0.8300	−4.5800	0.13829	0.14195	0.14171	0.702424	−1.7544	0.66264
6	0.72030	0.05750	0.05750	0.37976	0.49843	0.49062	0.748012	−0.8822	0.68844
7	−5.4554	1.10507	1.10702	0.37135	1.51126	1.50930	0.199868	−0.1709	0.00385
8	−7.4683	1.55824	1.55824	6.05083	−0.7505	−0.7505	0.918857	−1.0854	−5.9491
9	−7.8736	0.24057	0.24057	0.95588	0.22079	0.22079	1.270213	1.82921	0.83593
10	−6.9500	−0.1263	−0.1263	0.59958	−0.2013	−0.2013	0.873072	0.47301	0.63207

Equation (5.5) will be

$$A\frac{d^2Y(t)}{dt^2} + B\frac{d^{3/2}Y(t)}{dt^{3/2}} + CY(t) = 0, \qquad Y(0) = Y'(0) = 0. \tag{5.8}$$

This problem is solved on the same methodology adopted for previous example. However, the problem-specific fitness function for (5.5) by taking the values of constant coefficient as unity can be formulated as

$$e_j = \frac{1}{5}\sum_{i=1}^{5}\left[\frac{d^2\widehat{y}(t_i)}{dt_i{}^2} + \frac{d^{3/2}\widehat{y}(t_i)}{dt_i{}^{3/2}} + \widehat{y}(t_i) - (t_i + 1)\right]^2 + \frac{1}{2}\left[\widehat{y}(0) + \frac{d}{dt}\widehat{y}(0)\right]^2\Bigg|_j, \quad j = 1, 2, 3, \ldots, \tag{5.9}$$

where j is the generations index, $\widehat{y}$, $d^2\widehat{y}/dt^2$, and $d^{3/2}\widehat{y}/dt^{3/2}$ are FDE-NN networks provided in (3.2), (3.3), and (3.4), respectively. The one of unknown set of weights of FDE-NN networks trained stochastically by using PS, GA, and GA-SA is given in Table 6. These weights are used in expression (3.2) for finding the solution of the equations for some inputs between 0 and 1.

Table 7: Comparison of results for example II.

T	$y(t)$	$\widehat{y}(t)$			$\lvert y(t) - \widehat{y}(t)\rvert$		
		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

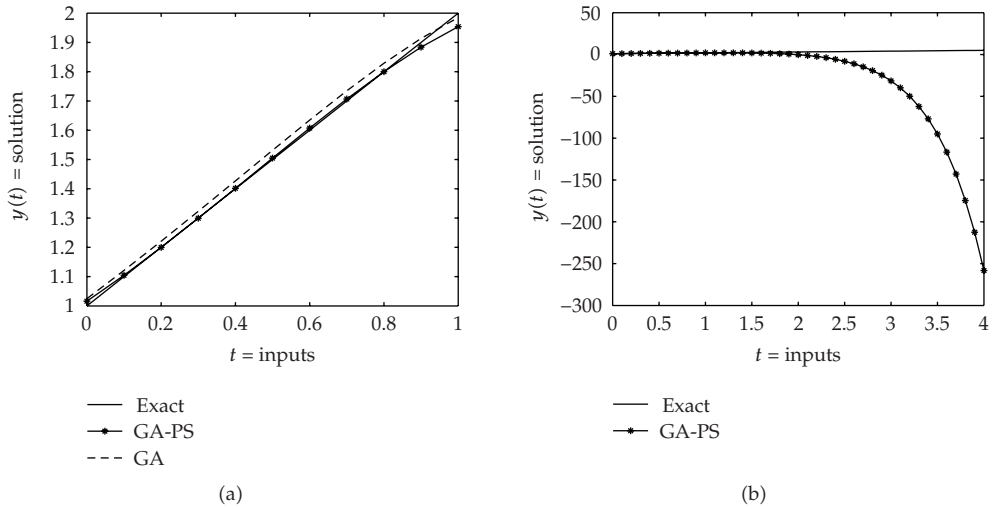

Figure 5: Comparison of the results (a) is for interval (0, 1) and (b) is for interval (0, 4).

The results are summarized in Table 7. It can be inferred from the table that the 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^{-3} to 10^{-4}.

The results computed for the algorithm are also plotted graphically in Figure 5. In Figure 5(a), the comparison of the results is 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 Figure 5(b) by using same weights, and it can be seen that error is starting to grow for inputs larger than 1 and it is tending to for inputs greater than 2. One can increase the accuracy slightly on large intervals by taking large number of neurons in FDE-NN networks or by 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 [36, 37].

5.3. Example III

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. The designed scheme is applied for such problem in order to analyze further the applicability, reliability, effectiveness, and efficiency.

Consider the fractional order system [3, 17]

$$\frac{d^2y(t)}{dt^2} + \frac{1}{2}\frac{d^{3/2}y(t)}{dt^{3/2}} + \frac{1}{2}y(t) = f(t), \quad 0 < t, \tag{5.10}$$

where, the forcing function and the initial conditions are given as

$$f(t) = \begin{cases} 8, & (0 \le t \le 1), \\ 0, & (t > 1), \end{cases} \qquad y(0) = y'(0) = 0. \tag{5.11}$$

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

$$e_j = \frac{1}{10}\sum_{i=1}^{10}\left[\frac{d^2\widehat{y}(t_i)}{dt_i{}^2} + \frac{1}{2}\frac{d^{3/2}\widehat{y}(t_i)}{dt_i{}^{3/2}} + \frac{1}{2}\widehat{y}(t_i) - 8\right]^2 + \frac{1}{2}\left[\widehat{y}(0) + \frac{d}{dt}\widehat{y}(0)\right]^2\Bigg|_j, \quad j = 1, 2, 3, \ldots. \tag{5.12}$$

The aim of our algorithm is to tune weights for which the value of fitness function as given in (5.12) is at its minimum. One such set of unknown weights of FDE-NN networks trained stochastically using PS, GA, and GA-SA is given in Figures 6(a), 6(b), and 6(c), respectively. The results obtained based on these weights are summarized in Table 8. It also includes the results of PMA algorithm provided with same set of parameters as taken in example I. It can be inferred from the results of the given scheme that it can perform comparable to the specialized state of art numerical solvers.

5.4. 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 that 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 (5.4), (5.9), and (5.12) in the interval (0, 1) randomly. It means that the mesh size used in our scheme is 0.5, 0.2, and 0.1 for examples I, II, and III, respectively. On the other hand in PMA technique the results were obtained with number of step equals to 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 Tables 3 and 8 that the results of our scheme

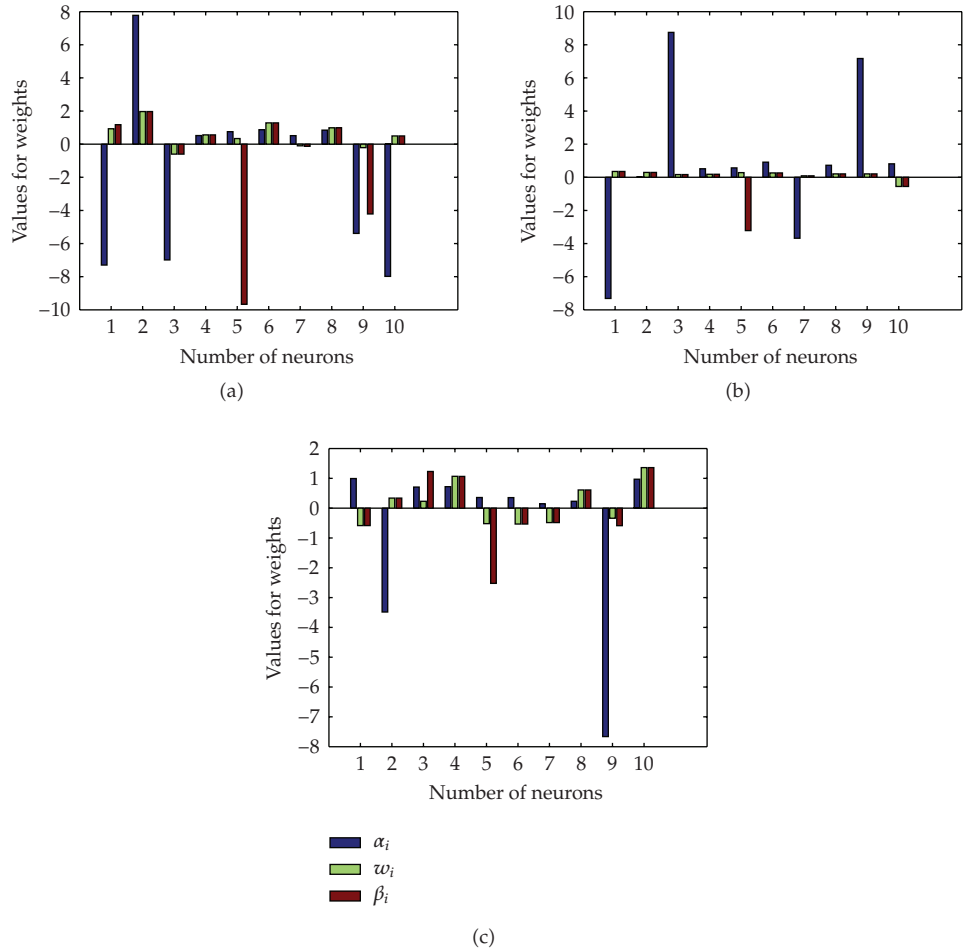

Figure 6: The weights trained for FDE-NN networks for example III, (a) is for PS, (b) is for GA, and (c) is for GA-PS algorithms.

are close 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 3 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 [14] have good approximation for the exact solution in closed vicinity of initial guess while the value of absolute error is going to rise with increase of time, for example, at $t = 1$, it is 0.225. However, the results by PMA method are better than those of GA-PS algorithm, but it has limitation as discussed earlier.

Moreover, it is a 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.

Table 8: Comparison of results for the solution of the equation in example III.

T	PMA	$\widehat{y}(t)$		
		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

Table 9: Computational complexity of the algorithms.

Span (0,T)	No. of steps	FDE-NN	Counts for MLF (Million)	Total time	Time (MLF)
(0, 1)	02	GA	12.0000	462s	371s
(0, 1)	02	PS	0.06000	7.66s	4.55s
(0, 1)	05	GA	30.0000	1149s	920s
(0, 1)	05	PS	0.15000	15.4s	9.34s
(0, 1)	10	GA	60.0000	2297s	1852s
(0, 1)	10	PS	0.30000	31,7s	18.6s
(0, 1)	20	GA	120.000	4593s	3695s
(0, 1)	20	PS	0.60000	60.8s	37.5s

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 (5.12) the algorithm required executing MLF function 20000 times for single generation, whereas the single generation using fitness functions in (5.4) and (5.9) 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 provided in Table 9. It can be seen from Table 9 that about 80% of execution time is spent on calculation of MLF function only. The time analysis provided in this paper is carried out using Dell Latitude D630 laptop with Intel(R) Core(TM) 2 Duo CPU T9300, 2.50 GHz, and MATLAB version R2008b.

6. Conclusions

On the basis of the simulations and analysis, 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 are 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.

Acknowledgment

The author would like to acknowledge Dr. Siraj-ul-Islam Ahmed for his valuable suggestion to improve the presentation of the paper.

References

[1] P. J. Torvik and R. L. Bagley, "On the appearance of the fractional derivative in the behavior of real materials," *Journal of Applied Mechanics, Transactions of ASME*, vol. 51, no. 2, pp. 294–298, 1984.

[2] R. L. Bagley and P. J. Torvik, "Fractional calculus—a different approach to the analysis of viscoelastically damped structures," *AIAA Journal*, vol. 21, no. 5, pp. 741–748, 1983.

[3] I. Podlubny, *Fractional Differential Equations*, vol. 198 of *Mathematics in Science and Engineering*, Academic Press, San Diego, Calif, USA, 1999.

[4] S. S. Ray and R. K. Bera, "Analytical solution of the Bagley Torvik equation by Adomian decomposition method," *Applied Mathematics and Computation*, vol. 168, no. 1, pp. 398–410, 2005.

[5] S. Manabe, "A suggestion of fractional-order controller for flexible spacecraft attitude control," *Nonlinear Dynamics*, vol. 29, no. 1–4, pp. 251–268, 2002.

[6] 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, p. 19, 2010.

[7] R. P. Agarwal, M. Belmekki, and M. Benchohra, "A survey on semilinear differential equations and inclusions involving Riemann-Liouville fractional derivative," *Advances in Difference Equations*, vol. 2009, Article ID 981728, p. 47, 2009.

[8] K. (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, p. 18, 2006.

[9] 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, p. 18, 2009.

[10] R. M. A. Zahoor 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.

[11] M. Naber, "Linear fractionally damped oscillator," *International Journal of Differential Equations*, vol. 2010, Article ID 197020, p. 12, 2010.

[12] Y. Hu, Y. Luo, and Z. Lu, "Analytical solution of the linear fractional differential equation by Adomian decomposition method," *Journal of Computational and Applied Mathematics*, vol. 215, no. 1, pp. 220–229, 2008.

[13] A. M. A. El-Sayed, I. L. El-Kalla, and E. A. A. Ziada, "Analytical and numerical solutions of multi-term nonlinear fractional orders differential equations," *Applied Numerical Mathematics*, vol. 60, no. 8, pp. 788–797, 2010.

[14] 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, p. 9, 2008.

[15] Y. Çenesiz, Y. Keskin, and A. Kurnaz, "The solution of the Bagley-Torvik equation with the generalized Taylor collocation method," *Journal of the Franklin Institute. Engineering and Applied Mathematics*, vol. 347, no. 2, pp. 452–466, 2010.

[16] K. Diethelm and N. J. Ford, "Numerical solution of the Bagley-Torvik equation," *BIT. Numerical Mathematics*, vol. 42, no. 3, pp. 490–507, 2002.

[17] I. Podlubny, T. Skovranek, and B. M. Vinagre Jara, "Matrix approach to discretization of fractional derivatives and to solution of fractional differential equations and their systems," in *Proceedings of the IEEE Conference on Emerging Technologies and Factory Automation (ETFA '09)*, pp. 1–6, 2009.

[18] P. García, J. D. Mingo, P. L. Carro, and A. Valdovinos, "Efficient feedforward linearization technique using genetic algorithms for OFDM systems," *Eurasip Journal on Advances in Signal Processing*, vol. 2010, Article ID 354030, p. 10, 2010.

[19] C. R. Reeves and J. E. Rowe, *Genetic Algorithms: Principles and Perspectives*, vol. 20 of *Operations Research/Computer Science Interfaces Series*, Kluwer Academic Publishers, Boston, Mass, USA, 2003.

[20] K. B. Oldham and J. Spanier, *The Fractional Calculus: Theory and Applications of Differentiation and Integration to Arbitrary Order*, vol. 11 of *Mathematics in Science and Engineering*, Academic Press, London, UK, 1974.

[21] K. S. Miller and B. Ross, *An Introduction to the Fractional Calculus and Fractional Differential Equations*, A Wiley-Interscience Publication, John Wiley & Sons, New York, NY, USA, 1993.

[22] A. K. Anatoly, H. M. Srivastava, and J. J. Trujillo, *Theory and Applications of Fractional Differential Equations*, vol. 204 of *North-Holland Mathematics Studies*, Elsevier Science, Amsterdam, The Netherlands, 2006.

[23] F. Mainardi and R. Gorenflo, "The Mittag-Leffler function in the Riemann-Liouville fractional calculus," in *Proceedings of the International Conference Dedicated to the Memory of Academician F. D. Gakhov*, A. A. Kilbas, Ed., pp. 215–225, Belarus State University, Minsk, Beloruss, 1996.

[24] R. M. A. Zahoor, J. A. Khan, and I. M. Qureshi, "Evolutionary computation technique for solving Riccati differential equation of arbitrary order," *World Academy of Science, Engineering and Technology*, vol. 58, pp. 531–536, 2009.

[25] M. A. Z. Raja, J. A. Khan, and I. M. Qureshi, "Evolutionary computational intelligence in solving the fractional differential equations," in *Proceedings of the Asian Conference on Intelligent Information and Database Systems (ACIIDS '10)*, vol. 5990 of *Lecture Notes in Computer Science*, pp. 231–240, Spinger, Hue City, Vietnam, 2010.

[26] M. A. Zahoor, Raja, J. A. Khan, and I. M. Qureshi, "Heuristic computational approach using swarm intelligence in solving fractional differential equations," in *Proceedings of the 12th Annual Genetic and Evolutionary Computation Conference (GECCO '10)*, pp. 2023–2026, Portland, Oregon, 2010.

[27] V. Torczon, "On the convergence of pattern search algorithms," *SIAM Journal on Optimization*, vol. 7, no. 1, pp. 1–25, 1997.

[28] X. Zhang and J. Ma, "Pattern search methods for finite minimax problems," *Journal of Applied Mathematics and Computing*, vol. 32, no. 2, pp. 491–506, 2010.

[29] E. D. Dolan, R. M. Lewis, and V. Torczon, "On the local convergence of pattern search," *SIAM Journal on Optimization*, vol. 14, no. 2, pp. 567–583, 2003.

[30] M. A. Taddy, H. K. H. Lee, G. A. Gray, and J. D. Griffin, "Bayesian guided pattern search for robust local optimization," *Technometrics*, vol. 51, no. 4, pp. 389–401, 2009.

[31] W. M. Spears, *Evolutionary Algorithm: The Role of Mutation and Recombination*, Natural Computing Series, Springer, Berlin, Germany, 2000.

[32] C. Grosan and A. Abraham, "Hybrid evolutionary algorithms: methodologies, architectures, and reviews," in *Studies in Computational Intelligence*, vol. 75, Springer, Berlin, Germany, 2007.

[33] E. A. Rawashdeh, "Numerical solution of semidifferential equations by collocation method," *Applied Mathematics and Computation*, vol. 174, no. 2, pp. 869–876, 2006.

[34] I. Podlubny, "Matrix approach to discretization of ODEs and PDEs of arbitrary real order," Matlab Central file exchange 22071, 2008.

[35] I. Podlubny, "Mittag-Leffler Function," Tech. Rep. 8738, Matlab Central, 2009, http://www.mathworks.com/matlabcentral/fileexchange/8738-mittag-leffler-function.

[36] 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. 2, 2011.

[37] M. A. Z. Raja, J. A. Khan, and I. M. Qureshi, "A new stochastic approach for solution of Riccati differential equation of fractional order," *Annals of Mathematics and Artificial Intelligence*. In press.