Journal of Computational and Applied Mathematics 311 (2017) 314–323

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Evolutionary computational intelligence in solving a class of nonlinear Volterra–Fredholm integro-differential equations

Bothayna S.H. Kashkaria [a], Muhammed I. Syam [b,*]

[a] Department of Mathematics, Faculty of Science, AL Faisaliah, King Abdulaziz University, Jeddah, Saudi Arabia

[b] Department of Mathematical Sciences, UAE University, College of Science, P. O. Box 17551, Al-Ain, United Arab Emirates

ARTICLE INFO

Article history:
Received 23 May 2016
Received in revised form 15 June 2016

Keywords:
Volterra–Fredholm integro-differential equations
Computational intelligence
Artificial neural network

ABSTRACT

In this paper, a stochastic computational intelligence technique for solving a class of nonlinear Volterra–Fredholm integro-differential equations with mixed conditions is presented. The strength of feed forward artificial neural networks is used to accurately model the integro-equation. Comparisons with the exact solution and other numerical techniques are presented to show the efficiency of the proposed method. Theoretical and numerical results are presented. Analysis for the presented method is given.

1. Introduction

Many physical phenomena and engineering problems are governed by mathematical models involving differential and integral equations. For example: applications related to geophysical fluid dynamics problems, fluid dynamics, potential theory, astronomy, biology, economics, electrostatics, studies of edge effect in elastic shells, modeling oceanic and atmospheric circulation, chemical reactors theory, convection diffusion processes, and optimal control; among many other areas of applied mathematics and engineering [1–3]. For many years, nonlinear Volterra–Fredholm integro-differential equations have drawn the attention of many researchers and practitioners who devised various techniques for their numerical solutions; among them the works in [4–13,2,3,14–16].

Interests in accurately approximating the solutions of Volterra–Fredholm integro-differential equations have been the focus of attention of many scientists. There are numerous special purpose techniques to adequately deal with such problems; for example the Taylor expansion approach [4–7], the triangular functions [8], the reproducing kernel Hilbert space [9], the Legendre–Galerkin method [10], the second kind Chebyshev wavelet [11], the Adomian decomposition method [12], the maximum principle [13], the sequential approach [14], the hybrid function method [17], the Legendre matrix method [18], the Legendre wavelets method [19], the Tau method [20], the compact finite difference method [21], the Sinc–Galerkin method [22], the Haar function method [23], the Cas wavelet method [24], the differential transform method [25], the sine–cosine wavelet methods [26], the Bessel matrix method [27], the Shannon wavelets approximation [28], the Bernstein polynomial method [29], the Bernstein collocation procedure [30], the product integration and Lagrangian interpolation methods [31], and iterative method [32]. In addition, scientists consider several numerical techniques to solve system of integro-differential equations such as the variational iteration method [15] and the differential transform method [16].

In the recent years, the approximate solution of boundary value problems using the artificial neural network (ANN) has been the focus of attention of many scientists such as [33–36]. Their approaches based on artificial intelligence using neural

* Corresponding author.
E-mail addresses: bkashkari@kau.edu.sa (B.S.H. Kashkaria), m.syam@uaeu.ac.ae (M.I. Syam).

http://dx.doi.org/10.1016/j.cam.2016.07.027
0377-0427/

networks optimized global and local search techniques. However, they are relatively less exploited in this domain in spite of their efficiency.

In this paper, numerical treatment for a class of non-linear Volterra integro-differential problems is presented using the evolutionary computational intelligence. We investigate the numerical solution of a class of high-order nonlinear Volterra–Fredholm integro-differential equations of the form

$$\sum_{j=1}^{r} p_j(x)y^{(j)}(x) = f(x) + \lambda_1 \int_a^x K_1(x,t)g_1(t,y(t))dt + \lambda_2 \int_a^b K_2(x,t)g_2(t,y(t))dt, \quad a < x < b \tag{1.1}$$

subject to the mixed conditions

$$\sum_{j=0}^{r-1} \left(a_{ij}y^{(j)}(a) + b_{ij}y^{(j)}(b) + c_{ij}y^{(j)}(c)\right) = \mu_{ij} \tag{1.2}$$

for $i = 0, 1, \ldots, r-1$, $a \le x \le b$, where $f(x), K_1(x,t), K_2(x,t), g_1(t,y(t)), g_2(t,y(t))$, and $p_j(x)$, $j = 1, \ldots, r$, are smooth functions on $a \le x, t \le b$, i.e., they have derivatives as much as the discussion required and $p_r(x) > 0$ for all $x \in [a,b]$. Also, $\lambda_1, \lambda_2, \mu_{ij}$ for $i, j = 0, \ldots, r-1$ are constants and r is a nonnegative integer. Problem (1.1)–(1.2) was studied by Darania and Ivaz [4] using a Taylor expansion approach. The same approach is used by Yalcinbas and Sezer [5,6] when $g_1(t,y(t)) = g_2(t,y(t)) = y(t)$ and Maleknejad and Mahmoudi [7] when $g_1(t,y(t)) = y^p(t)$ and $g_2(t,y(t)) = y(t)$.

In Section 2, the mathematical modeling of the problem is presented. In Section 3, we present the idea of the evolutionary computational intelligence method while in Section 4, analytical results are presented. In Section 5, simulation and numerical results for a number of examples are presented and discussed. Finally, some conclusion and future work are drawn.

2. Mathematical modeling

In this section, we present the description of the designed model along with the fitness function formulation. Following the approach of Raja [36], the approximate solution of problem (1.1)–(1.2) is written as

$$y_n(x) = \sum_{i=1}^{n} \alpha_i f(w_i x + \beta_i) \tag{2.1}$$

where α_i, w_i, and β_i are the weights, n is the number of neurons, and f is the log-sigmoid transfer function which is given by

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

Simple calculations imply that

$$\begin{aligned}
y_n(x) &= \sum_{i=1}^{n} \alpha_i \left(\frac{1}{1+e^{-(w_i x+\beta_i)}}\right),\\
y_n'(x) &= \sum_{i=1}^{n} \alpha_i w_i \left(\frac{e^{-(w_i x+\beta_i)}}{\left(1+e^{-(w_i x+\beta_i)}\right)^2}\right),\\
&\vdots\\
y_n^{(r)}(x) &= \sum_{i=1}^{n} \alpha_i \frac{d^r}{dx^r}\left(\frac{1}{1+e^{-(w_i x+\beta_i)}}\right).
\end{aligned}$$

Substitute the approximation of the derivatives in Eq. (1.1). Let

$$x_i = a + ih, \quad i = 0, 1, \ldots, m$$

where $h = \frac{b-a}{m}$. Then, $\{x_0, x_1, \ldots, x_m\}$ is a uniform partition of $[a,b]$. The error ϵ_1 is given by

$$\begin{aligned}
\epsilon_1 = \frac{1}{m+1} \sum_{i=0}^{m} \Bigg(&\sum_{j=1}^{r} p_j(x) y_n^{(j)}(x_i) - f(x_i) - \lambda_1 \int_a^{x_i} K_1(x_i,t)g_1(t,y_n(t))dt\\
&- \lambda_2 \int_a^b K_2(x_i,t)g_2(t,y_n(t))dt \Bigg)^2.
\end{aligned} \tag{2.2}$$

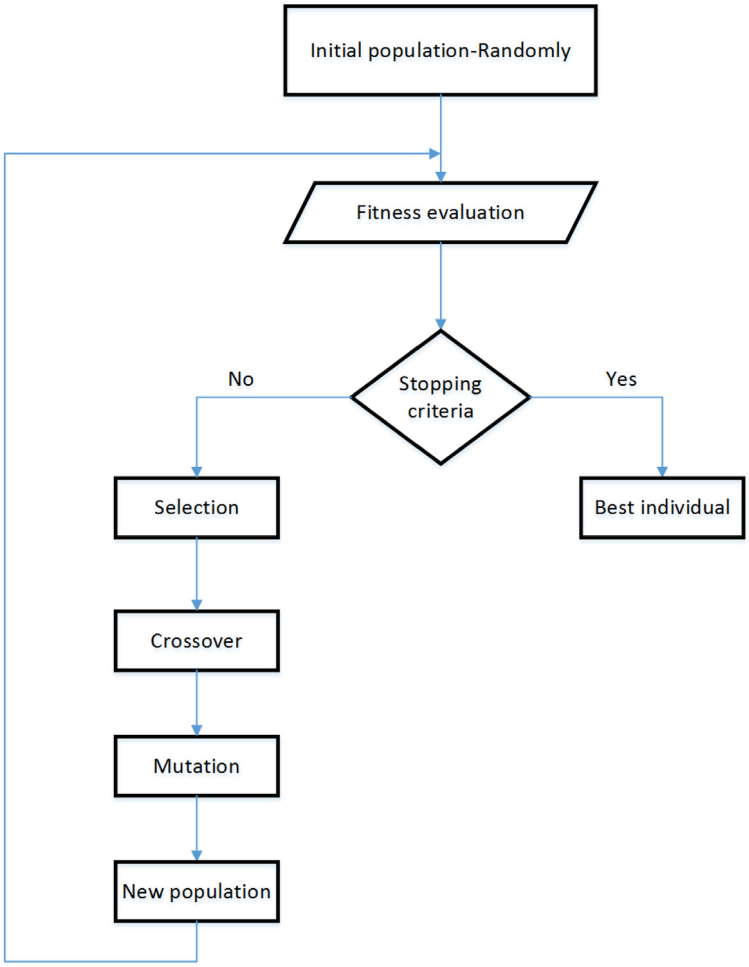

Fig. 1. Flowchart of GA.

The second error is generated from the mixed conditions

$$\epsilon_2 = \frac{1}{r+1}\left(\sum_{j=0}^{r-1}\left(a_{ij}y_n^{(j)}(x_0) + b_{ij}y_n^{(j)}(x_n) + c_{ij}y_n^{(j)}(c)\right) - \mu_{ij}\right)^2. \tag{2.3}$$

The fitness function, ϵ, for the model has been constructed in an unsupervised manner as

$$\epsilon = \epsilon_1 + \epsilon_2. \tag{2.4}$$

Our aim is to find the weights α_i, w_i, and β_i, for $i = 1, 2, \ldots, n$, so that ϵ approaches to zero. As ϵ approaches to zero, y_n tends to y and so that the approximate solution for the non-linear Volterra integro-differential problem (1.1)–(1.2) is found. The fitness function depends only on the weights.

3. Evolutionary computing

In this section, we present an efficient computational method to compute the weights of networks representing the problem (1.1)–(1.2) based on genetic algorithm and active set algorithm. The main advantage of such technique is unlike other techniques, it does not stuck in local minimum. This technique consists of three main steps:

1. Selection
2. Crossover
3. Mutation.

These steps can be done by built in functions in Matlab. The genetic algorithm (GA) works iteratively using its operators randomly based on fitness function. The generic flow chart of the algorithm is given in Fig. 1. For more details, see [37].

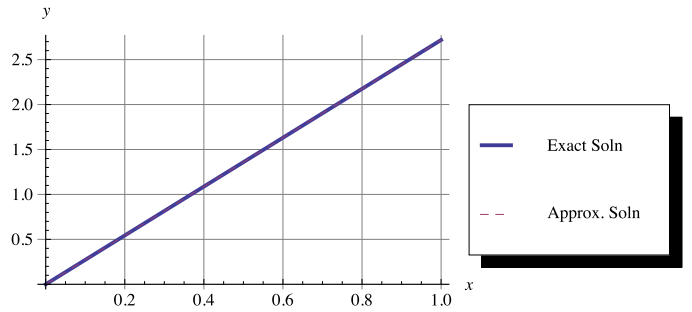

Fig. 2. Exact and Approx. Solu. of Example 1.

The algorithm of the proposed technique will be as follows:

Algorithm 1:

Input: $error = 10^{-9}$ % maximum error
$L = 1500$ % maximum number of cycles
$n = 10$ % number of nodes in the partition of $[a, b]$ is 11
$E = 3$ % Elite count
$F = \frac{3}{4}$ % crossover fraction
$M_1 = 12$ % Migration in forward direction
$M_2 = -12$ % Migration in Backward direction

Step 1: Generate randomly initial population of P number of chromosome
Step 2: Choose s such that $\frac{P}{S} =$ integer % s : number of subpopulations
Step 3: Let $count = 1$ % $count$: number of cycles
Step 4: Compute the fitness function % using Eq. (6)
Step 5: Ranking over all subpopulations
Step 6: Store the best fitted solution
Step 7: Let $count = count + 1$
Step 8: If $\epsilon < error$ or $count \geq L$, go to step 13 % stopping criteria
else do steps 7–10
Step 9: Crossover % call scattered function (Matlab)
Step 10: Mutation % call adaptive feasible function (Matlab)
Step 11: Selection % call stochastic uniform function (Matlab)
Step 12: Elitism
Step 13: Go to step 4
Step 14: Refinement of result using active set Alg. % call Fmincon function (Matlab)
Step 15: Repeat steps 1–13 for large number of runs
Step 16: Stop

In the next section, we implement the proposed numerical technique for three examples to show the efficiency of Algorithm (1).

4. Analytical results

The existence and the uniqueness of the exact solution of problem (1.1)–(1.2) are investigated herein. Since we can use the shooting method which required converting the boundary value problem to initial value problem, we discuss in the next theorems the existence and uniqueness of the solution to

$$y^{(j)}(a) = \mu_i \tag{4.1}$$

for $j = 0, 1, \ldots, r-1$. In this section, we assume that f and $p_j \in C[a, b]$, for $j = 1, 2, \ldots, r-1$, $p_r = 1, K_1 \in D = \{(x, t) : x, t \in [a, b], x \geq t\}$, $K_2 \in C[a, b]^2$, and $g_1, g_2 \in C[a, b] \times \Re$. Define the following norm on $C^{r-1}[a, b]$:

$$\|y\|_T = \max\left\{\|y\|_\infty, \left\|y'\right\|_\infty, \ldots, \left\|y^{(r-1)}\right\|_\infty\right\}$$

where $\|\cdot\|_\infty$ is the Chebyshev norm.

Theorem 4.1. *Let $y \in C^r[a, b]$. Then, y is a solution to problem (1.1)–(1.2) if and only if y is a solution to the integral equation*

$$y(x) = \sum_{j=0}^{r-1} \frac{\mu_j}{j!}(x-a)^j + \frac{1}{(r-1)!}\int_a^x (x-t)^{r-1} h(t, y, y', \ldots, y^{(r-1)}, (Ty)(t), (Sy)(t))dt \tag{4.2}$$

where

$$(Ty)(x) = \lambda_1 \int_a^x K_1(x,t) g_1(t, y(t))dt, \qquad (Sy)(x) = \lambda_2 \int_a^b K_2(x,t) g_2(t, y(t))dt$$

and

$$h(x, y, y', \ldots, y^{(r-1)}, (Ty)(x), (Sy)(x)) = f(x) - \sum_{j=1}^{r-1} p_j(x) y^{(j)}(x) + (Ty)(x) + (Sy)(x).$$

Proof. Since $y \in C^r[a,b]$, by Taylor's theorem,

$$y(x) = \sum_{j=0}^{r-1} \frac{\mu_j}{j!}(x-a)^j + \frac{1}{(r-1)!}\int_a^x (x-t)^{r-1} y^{(r)}(t)dt.$$

If y is a solution to problem (1.1)–(1.2), then

$$\begin{aligned} y(x) &= \sum_{j=1}^{r-1} \frac{\mu_j}{j!}(x-a)^j + \frac{1}{(r-1)!}\int_a^x (x-t)^{r-1} y^{(r)}(t)dt \\ &= \sum_{j=0}^{r-1} \frac{\mu_j}{j!}(x-a)^j + \frac{1}{(r-1)!}\int_a^x (x-t)^{r-1} h(t, y, y', \ldots, y^{(r-1)}, (Ty)(t), (Sy)(t))dt \end{aligned}$$

which implies that y satisfies the integral equation (4.2).

If y satisfies the integral equation (4.2), then differentiate equation (4.2) r times to get

$$\begin{aligned} y'(x) &= \sum_{j=1}^{r-1} \frac{\mu_j}{(j-1)!}(x-a)^{j-1} + \frac{1}{(r-2)!}\int_a^x (x-t)^{r-2} h(t, y, y', \ldots, y^{(r-1)}, (Ty)(t), (Sy)(t))dt, \\ y''(x) &= \sum_{j=2}^{r-1} \frac{\mu_j}{(j-2)!}(x-a)^{j-2} + \frac{1}{(r-3)!}\int_a^x (x-t)^{r-3} h(t, y, y', \ldots, y^{(r-1)}, (Ty)(t), (Sy)(t))dt, \\ &\vdots \\ y^{(r)}(x) &= h(x, y, y', \ldots, y^{(r-1)}, (Ty)(x), (Sy)(x)). \end{aligned}$$

Thus, y satisfies Eq. (1.1). Direct substitution shows that $y^{(j)}(a) = \mu_i$ for $j = 0, 1, \ldots, r-1$ which completes the proof. It is enough to show that the integral equation (4.2) has a unique solution to prove that problem (1.1)–(1.2) has a unique solution.

Theorem 4.2 (*Existence*). *Let $f, p_i \in C[a,b]$, for $i = 0, 1, \ldots, r-1$, $K_1 \in C[D]$, $K_2 \in C[a,b]^2$, $g_1, g_2 C[a,b] \times \Re$, and there are positive real numbers G_1^* and G_2^* such that*

$$|g_1(x,y)| \le G_1^* \|y\|_T \quad \text{and} \quad |g_2(x,y)| \le G_2^* \|y\|_T .$$

Then for any $\epsilon > 0$ and

$$\chi = \min\left\{ b, a + \ln\left[\epsilon\, r! \left(\sum_{j=1}^{r-1} \|p_j\|_\infty + (b-a)\left[|\lambda_1| \|K_1\|_\infty G_1^* + |\lambda_2| \|K_2\|_\infty G_2^* \right] \right) \right] \right\},$$

there exists $y : [a, \chi] \to \Re$ solving (4.1) and (4.2).

Proof. Let

$$G = \left\{ y \in C^{r-1}[a,b] : \left\| y - \sum_{j=0}^{r-1} \frac{\mu_j}{j!} x^j - \frac{1}{(r-1)!}\int_a^x (x-t)^{r-1} f(t)dt \right\|_T < \epsilon \right\}.$$

Then, G is a closed subset of the Banach space of all $(r-1)$ continuous differentiable functions on $[a,b]$ equipped with the T-norm. Since $y(x) = \sum_{j=0}^{r-1} \frac{\mu_j}{j!} x^j - \frac{1}{(r-1)!}\int_a^x (x-t)^{r-1} f(t)dt \in G$. Thus, $G \neq \phi$. Define the operator Υ on G by

$$\Upsilon[y](x) = \sum_{j=0}^{r-1} \frac{\mu_j}{j!}(x-a)^j + \frac{1}{(r-1)!}\int_a^x (x-t)^{r-1} h(t, y, y', \ldots, y^{(r-1)}, (Ty)(t), (Sy)(t))dt. \tag{4.3}$$

Eq. (4.1) can be written as

$$\Upsilon[y] = y. \tag{4.4}$$

Our aim is to show that Eq. (4.3) has a fixed point in G. Since $f \in C[a,b]$, $K_1 \in C[D], K_2 \in C[a,b]^2$, and $g_1, g_2 \in C[a,b] \times \Re$, then Υ is a continuous function. To end the proof, we want to show that Υ is self-mapping on G. First, For any $i \in \{0, 1, \ldots, r-1\}$ and $x \in [a,b]$,

$$\begin{aligned}
&\left| \Upsilon[y](x) - \sum_{j=0}^{r-1} \frac{\mu_j}{j!} x^j - \frac{1}{(r-1)!} \int_a^x (x-t)^{r-1} f(t) dt \right| \\
&= \left| \frac{1}{(r-1)!} \int_a^x (x-t)^{r-1} h(t, y, y', \ldots, y^{(r-1)}, (Ty)(t), (Sy)(t)) dt - \frac{1}{(r-1)!} \int_a^x (x-t)^{r-1} f(t) dt \right| \\
&= \left| \frac{1}{(r-1)!} \int_a^x (x-t)^{r-1} \left[\sum_{j=1}^{r-1} p_j(t) y^{(j)}(t) + (Ty)(t) + (Sy)(t) \right] dt \right| \\
&\leq \frac{(x-a)^r}{r!} \left(\sum_{j=1}^{r-1} \|p_j\|_\infty + (b-a) \left[|\lambda_1| \, \|K_1\|_\infty G_1^* + |\lambda_2| \, \|K_2\|_\infty G_2^* \right] \right) \|y\|_T \\
&\leq \frac{(\chi-a)^r}{r!} \left(\sum_{j=1}^{r-1} \|p_j\|_\infty + (b-a) \left[|\lambda_1| \, \|K_1\|_\infty G_1^* + |\lambda_2| \, \|K_2\|_\infty G_2^* \right] \right) \|y\|_T \\
&\leq \epsilon \left\| y^{(j)} \right\|_\infty
\end{aligned}$$

which implies that $\left\| \Upsilon[y](x) - \sum_{j=0}^{r-1} \frac{\mu_j}{j!} x^j - \frac{1}{(r-1)!} \int_a^x (x-t)^{r-1} f(t) dt \right\|_T \leq \epsilon$. Thus, $\Upsilon[y] \in G$ if $y \in G$; that is, Υ maps G into itself. Based on Banach's fixed point theorem, it follows that the proof completes.

Theorem 4.3 (*Uniqueness*). *Let $f, p_i \in C[a,b]$, for $i = 0, 1, \ldots, r-1$, $K_1 \in C[D], K_2 \in C[a,b]^2$, $g_1, g_2 C[a,b] \times \Re$ be Lipschitz functions in the variable y with Lipschitz constants L_1 and L_2, respectively. If*

$$\frac{\left(\sum_{j=1}^{r-1} \|p_j\|_\infty + \lambda_1 \|K_1\|_\infty L_1 + \lambda_2 \|K_2\|_\infty L_2 \right) (b-a)^r}{r!} < 1,$$

then problem (4.1) and (4.2) has a unique solution.

Proof. Let y_1 and y_2 be two solutions to problem (4.1)–(4.2); then

$$\begin{aligned}
&\left| h(t, y_2, y_2', \ldots, y_2^{(r-1)}, (Ty_2)(t), (Sy_2)(t)) dt - h(t, y_1, y_1', \ldots, y_1^{(r-1)}, (Ty_1)(t), (Sy_1)(t)) dt \right| \\
&\leq \left| \sum_{j=1}^{r-1} p_j(x) (y_2^{(j)}(x)) - y_1^{(j)}(x) \right| + \left| \lambda_1 \int_a^x K_1(x,t) \left(g_1(t, y_2(t)) - g_1(t, y_1(t)) \right) dt \right| \\
&\quad + \left| \lambda_2 \int_a^b K_2(x,t) \left(g_2(t, y_2(t)) - g_2(t, y_1(t)) \right) dt \right| \\
&\leq \left(\sum_{j=1}^{r-1} \|p_j\|_\infty + \lambda_1 \|K_1\|_\infty L_1 + \lambda_2 \|K_2\|_\infty L_2 \right) \|y_2 - y_1\|_\infty
\end{aligned}$$

which implies that

$$\begin{aligned}
|y_2(x) - y_1(x)| &\leq \frac{\left(\sum_{j=1}^{r-1} \|p_j\|_\infty + \lambda_1 \|K_1\|_\infty L_1 + \lambda_2 \|K_2\|_\infty L_2 \right) \|y_2 - y_1\|_T}{(r-1)!} \int_a^x (x-t)^{r-1} \\
&\leq \frac{\left(\sum_{j=1}^{r-1} \|p_j\|_\infty + \lambda_1 \|K_1\|_\infty L_1 + \lambda_2 \|K_2\|_\infty L_2 \right) (b-a)^r}{r!} \|y_2 - y_1\|_\infty .
\end{aligned}$$

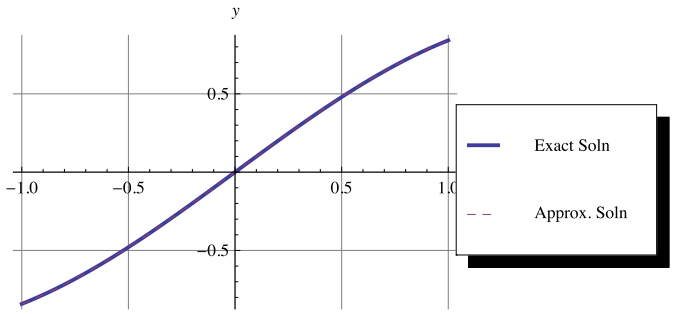

Fig. 3. Exact and Approx. Solu. of Example 2.

Thus,

$$\|y_2 - y_1\|_\infty \leq \frac{\left(\sum_{j=1}^{r-1} \|p_j\|_\infty + \lambda_1 \|K_1\|_\infty L_1 + \lambda_2 \|K_2\|_\infty L_2\right)(b-a)^r}{r!} \|y_2 - y_1\|_\infty .$$

Since

$$\frac{\left(\sum_{j=1}^{r-1} \|p_j\|_\infty + \lambda_1 \|K_1\|_\infty L_1 + \lambda_2 \|K_2\|_\infty L_2\right)(b-a)^r}{r!} < 1,$$

then $y_1 = y_2$ which completes the proof.

From Theorem 4.1, problem (1.1) and (4.1) has a unique solution. To prove the existence of solution to problem (1.1)–(1.2), it is enough to force the solution to satisfy the condition (1.2).

5. Simulation and numerical results

In this section, we implement the proposed numerical technique for three examples. Assume that the approximate solution of problem (1.1)–(1.2) is written as

$$y_n(x) = \sum_{i=1}^{8} \frac{\alpha_i}{1 + e^{-(w_i x + \beta_i)}}. \tag{5.1}$$

Then the total number of neurons are 24. We start with initial population consisting of a set of 180 chromosomes and 9 subpopulations such that each chromosome consists of 24 genes.

Example 1. Consider the Volterra integro-differential problem appearing in [38]

$$y'(x) = xe^x + e^x - x + \int_0^1 xy(t)dt, \quad x \in (0, 1)$$
$$y(0) = 0.$$

The exact solution is

$$y(x) = xe.$$

The exact and the approximate solution generated by the proposed method is presented in Fig. 2. Table 1 presents a comparison between the error in our results and the ones obtained by CAS wavelet method (CASM) [24], differential transformation method (DTM) [39], homotopy perturbation method (HPM) [40], and sequential method (SM) [14].

Example 2. Consider the Volterra integro-differential problem appearing in [41]

$$y'''(x) + xy''(x) - \sin(x)y = f(x) + \frac{1}{2}\int_{-1}^{x} \sin(x+t)y(t)dt - \int_{-1}^{1} \cos(x+t)y(t)dt, \quad x \in (-1, 1)$$
$$y(0) = 0, \qquad y'(0) = 1, \qquad y''(0) = 0$$

Table 1
Comparison between the error in our results in Example 1 and other methods.

x	CASM in [24]	DTM in [39]	HPM in [40]	SM in [14]	Our method
0.1	1.3492×10^{-3}	1.0012×10^{-2}	0.2315×10^{-5}	1.0179×10^{-7}	1.1134×10^{-14}
0.2	1.1596×10^{-3}	2.7865×10^{-2}	0.9259×10^{-5}	4.8277×10^{-7}	2.8971×10^{-14}
0.3	5.6715×10^{-3}	5.0873×10^{-2}	0.2083×10^{-4}	1.0178×10^{-6}	4.2008×10^{-14}
0.4	5.9311×10^{-2}	7.5536×10^{-2}	0.3704×10^{-4}	1.6193×10^{-6}	8.6510×10^{-14}
0.5	1.3233×10^{-2}	9.7189×10^{-2}	0.5787×10^{-4}	2.3089×10^{-6}	9.4326×10^{-14}
0.6	4.3929×10^{-2}	1.0955×10^{-1}	0.8333×10^{-4}	3.0935×10^{-6}	1.2221×10^{-13}
0.7	1.4120×10^{-2}	1.0413×10^{-1}	0.1134×10^{-3}	3.9780×10^{-6}	3.4469×10^{-13}
0.8	1.3451×10^{-2}	6.9451×10^{-2}	0.1481×10^{-3}	4.9957×10^{-6}	3.5284×10^{-13}
0.9	1.3205×10^{-2}	1.0003×10^{-2}	0.1875×10^{-3}	6.1354×10^{-6}	4.5461×10^{-13}

Table 2
Comparison between the error in our results in Example 2 and other methods.

x	ILM in [38]	Our method
0.2	9.8863×10^{-13}	1.0031×10^{-14}
0.4	5.1985×10^{-12}	3.1820×10^{-14}
0.6	1.3041×10^{-11}	5.5672×10^{-14}
0.8	2.4772×10^{-11}	7.5291×10^{-14}
1	4.0591×10^{-11}	8.2221×10^{-14}

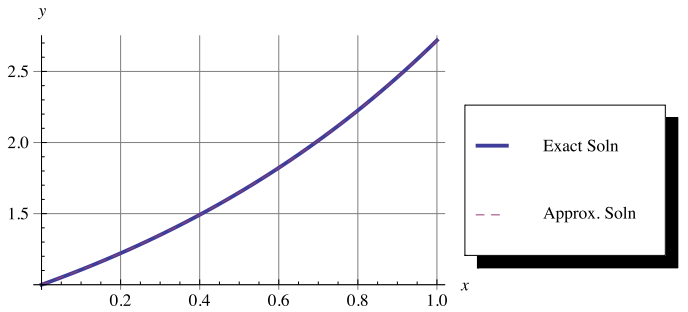

Fig. 4. Exact and Approx. Solu. of Example 3.

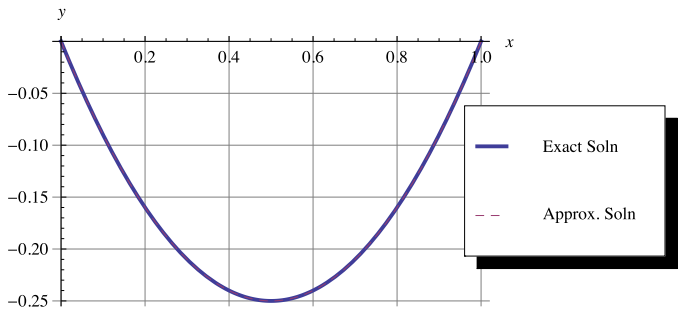

Fig. 5. Exact and Approx. Solu. of Example 4.

where

$$f(x) = \cos(x)\,(-1.25 + .125\sin(2) - 0.25x + 0.5\sin(x)\cos(x)) + \sin(x)\,(-x - \sin(x) + 0.5\sin(2) - 1.125 - 0.125\cos(2))\,.$$

The exact solution is

$$y(x) = \sin(x).$$

The exact and the approximate solution generated by the proposed method is presented in Fig. 3. Table 2 presents a comparison between the error in our results and the ones obtained by Improved Legendre method (ILM) [38].

Table 3
Comparison between the error in our results in Example 3 and VIM.

x	VIM in [42]	Our method
0.1	1.27×10^{-5}	2.23×10^{-11}
0.2	4.36×10^{-5}	3.51×10^{-11}
0.3	8.17×10^{-5}	1.10×10^{-10}
0.4	1.16×10^{-4}	2.17×10^{-10}
0.5	1.38×10^{-4}	5.32×10^{-10}
0.6	1.39×10^{-4}	5.82×10^{-10}
0.7	1.17×10^{-4}	3.09×10^{-10}
0.8	7.47×10^{-5}	6.24×10^{-11}
0.9	2.59×10^{-5}	2.17×10^{-11}

Table 4
The error in our results in Example 4.

x	Our method
0.1	1.1×10^{-15}
0.2	1.9×10^{-15}
0.3	3.1×10^{-15}
0.4	4.6×10^{-15}
0.5	2.2×10^{-14}
0.6	8.4×10^{-15}
0.7	4.2×10^{-15}
0.8	1.7×10^{-15}
0.9	1.2×10^{-15}

Example 3. Consider the Volterra integro-differential problem appearing in [42]

$$y^{(4)}(x) = 1 + \int_0^x e^{-t} y^2(t)dt, \quad x \in (0, 1)$$
$$y(0) = 1, \quad y'(0) = 1, \quad y(1) = e, \quad y'(1) = e.$$

The exact solution is

$$y(x) = e^x.$$

The exact and the approximate solution generated by the proposed method are presented in Fig. 4. Table 3 presents a comparison between the error in our results and the ones obtained by variational iteration method (VIM) [42].

Example 4. Consider the Volterra integro-differential problem appearing in [43]

$$y''(x) + 3y(x) = f(x) + \int_0^x \sin(x + t)y(y)dt, \quad x \in (0, 1)$$
$$y(0) = y(1) = 0,$$

where

$$f(x) = -2 + 3x - 3x^2 - (x^2 - x - 2)\cos(2x) + (2x - 1)\sin(2x) + \sin x - 2\cos x.$$

The exact solution is

$$y(x) = x^2 - x.$$

The error in [43] is about 10^{-6} in most of the interval [0, 1]. The exact and the approximate solution generated by the proposed method is presented in Fig. 5. Table 4 presents the error in our results.

Conclusion and future work: In this paper, we have solved a class of non-linear Volterra integro-differential problems subject to either boundary or initial conditions. The method of solution is based on the evolutionary computational intelligence. The numerical results for given examples and comparisons with other researchers demonstrate the efficiency and accuracy of the present method. It should be noted that applying Theorem 4.3 causes two difficulties. Firstly, the Lipschitz constant is not easy to find and, secondly, the necessary condition does not satisfy several cases of the problem (1.1)–(1.2). Therefore, a further discussion on weaker necessary conditions should be followed in the future work to cover wider range of problem (1.1)–(1.2). In addition, generalization to fraction derivative case will be investigated.

Acknowledgment

This project was funded by the Deanship of Scientific Research (DSR), King Abdulaziz University, Jeddah, under Grant No.(363 - 780 - D 1435). The authors, therefore, acknowledge with thanks DSR technical and financial support.

References

[1] L.M. Delves, J.L. Mohamed, Computational Methods for Integral Equations, Cambridge University Press, 1985.
[2] M.F. El-sayed, M. Syam, Electrohydrodynamic instability of a dielectric compressible liquid sheet streaming into an ambient stationary compressible gas, Arch. Appl. Mech. 77 (2007) 613–626.
[3] M. Syam, Modified Broyden-variational method for solving nonlinear elliptic differential equations, Choas Solitons Fractals 32 (2) (2007) 392–404.
[4] P. Darania, K. Ivaz, Numerical solution of nonlinear Volterra-Fredholm intgro-differntial equations, Comput. Math. Appl. 56 (2008) 2197–2209.
[5] S. Yalcinbas, Taylor polynomial solution of nonlinear Volterra–Fredholm integral equations, Appl. Math. Comput. 127 (2002) 195–206.
[6] S. Yalcinbas, M. Sezer, The approximate solution of high-order linear Volterra–Fredholm integro-differential equations in terms of Taylor polynomials, Appl. Math. Comput. 112 (2000) 291–308.
[7] K. Maleknejad, Y. Mahmoudi, Taylor polynomial solution of high-order nonlinear Volterra–Fredholm integro-differential equations, Appl. Math. Comput. 145 (2003) 641–653.
[8] E. Baboliana, Z. Masouria, S. Hatamzadeh-Varmazyarb, Numerical solution of nonlinear Volterra–Fredholm integro-differential equations via direct method using triangular functions, Comput. Math. Appl. 58 (2) (2009) 239–247.
[9] S. Momani, O. Abu Arqub, T. Hayat, H. Al-Sulami, A computational method for solving periodic boundary value problems for integro-differential equations of Fredholm–Volterra type, Appl. Math. Comput. 240 (2014) 229–239.
[10] M. Fathy, M. El-Gamel, M.S. El-Azab, Legendre–Galerkin method for the linear Fredholm integro-differential equations, Appl. Math. Comput. 243 (2014) 789–800.
[11] L. Zhu, Q. Fan, Solving fractional nonlinear Fredholm integro-differential equations by the second kind Chebyshev wavelet, Commun. Nonlinear Sci. Numer. Simul. 17 (6) (2012) 2333–2341.
[12] A. Wazwaz, R. Rach, J. Duan, Adomian decomposition method for solving the Volterra integral form of the Lane–Emden equations with initial values and boundary conditions, Appl. Math. Comput. 219 (10) (2013) 5004–5019.
[13] M. Syam, M.N. Anwar, A computational method for solving a class of non-linear singularly perturbed Volterra integro-differential boundary value problems, J. Math. Comput. Sci. 3 (1) (2013) 73–86.
[14] M.I. Berenguer, M.V. Munoz, A.I. Gullem, M. Galan, A sequential approach for solving the Fredholm integro-differential equation, Appl. Numer. Math. 62 (2012) 297–304.
[15] J. Saberi-Nadjafi, M. Tamamgar, The variational iteration method: A highly promising method for solving the system of integro-differential equations, Comput. Math. Appl. 56 (2008) 346–351.
[16] A. Arikoglu, I. Ozkol, Solutions of integral and integro-differential equation systems by using differential transform method, Comput. Math. Appl. 56 (9) (2008) 2411–2417.
[17] C.H. Hsiao, Hybrid function method for solving Fredholm and Volterra integral equations of the second kind, J. Comput. Appl. Math. 230 (2009) 56–68.
[18] S. Yalcinbas, M. Sezer, H.H. Sorkun, Legendre polynomial solutions of high-order linear Fredholm integro-differential equations, Appl. Math. Comput. 210 (2009) 334–349.
[19] N. Lakestani, B.N. Saray, M. Dehghan, Numerical solution for the weakly singular Fredholm integro-differential equations using legendre multiwavelets, J. Comput. Appl. Math. 235 (2011) 3291–3303.
[20] J.P. Mahmoud, M.Y.R. Ardabili, S. Shahmorad, Numerical solution of the system of Fredholm integro-differential equations by the tau method, Appl. Math. Comput. 168 (2005) 465–478.
[21] J. Zhao, R.M. Corless, Compact finite difference method for integro-differential equations, Appl. Math. Comput. 177 (2006) 271–288.
[22] M. Zarebnia, Sinc numerical solution for the Volterra integro-differential equation, Commun. Nonlinear Sci. Numer. Simul. 15 (2010) 700–706.
[23] M.H. Reihani, Z. Abadi, Rationalized haar functions method for solving Fredholm and Volterra integral equations, J. Comput. Appl. Math. 200 (2007) 12–20.
[24] H. Danfu, S. Xufeng, Numerical solution of integro-differential equations by using CAS wavelet operational matrix of integration, Appl. Math. Comput. 194 (2007) 460–466.
[25] P. Darania, A. Ebadian, A method for the numerical solution of the integro-differential equations, Appl. Math. Comput. 188 (2007) 657–668.
[26] M.T. Kajani, M. Ghasemi, E. Babolian, Comparison between the homotopy perturbation method and the sinecosine wavelet method for solving linear integro-differential equations, Comput. Math. Appl. 54 (2007) 1162–1168.
[27] S. Yuzbasi, N. Sahin, M. Sezer, Bessel polynomial solutions of high-order linear Volterra integro-differential equations, Comput. Math. Appl. 62 (2011) 1940–1956.
[28] K. Maleknejad, M. Attary, An efficient numerical approximation for the linear class of Fredholm integro-differential equations based on cattanis method, Commun. Nonlinear Sci. Numer. Simul. 16 (2011) 2672–2679.
[29] B.N. Mandal, S. Bhattacharya, Numerical solution of some classes of integral equations using Bernstein polynomials, Appl. Math. Comput. 190 (2007) 1707–1716.
[30] O.R. Isik, M. Sezer, Z. Guney, A rational approximation based on Bernstein polynomials for high order initial and boundary values problems, Appl. Math. Comput. 217 (2011) 9438–9450.
[31] C. Allouch, P. Sablonniere, D. Sbibih, M. Tahrichi, Product integration methods based on discrete spline quasi-interpolants and application to weakly singular integral equations, J. Comput. Appl. Math. 233 (2010) 2855–2866.
[32] W. Yulan, T. Chaolu, P. Jing, New algorithm for second-order boundary value problems of integro-differential equation, J. Comput. Appl. Math. 229 (2009) 1–6.
[33] J.A. Khan, M.A.Z. Raja, I.M. Qureshi, Stochastic computational approach for complex non-linear ordinary differential equations, Chinese Phys. Lett. 28 (2) (2011) 020206–020209.
[34] D.R. Parisi, M.C. Mariani, M.A. Laborde, Solving differential equations with unsupervised neural networks, Chem. Eng. Process. 42 (8–9) (2003) 715–721.
[35] M.A.Z. Raja, Solution of one-dimension Bratu equation arising in fuel ignition model using ANN optimized with PSO and SQP, Connect. Sci. 6 (3) (2014) 195–214.
[36] M.A.Z. Raja, J.A. Khan, I.M. Qureshi, Swarm intelligent optimized neural networks for solving fractional differential equations, Int. J. Innovative Comput. Inform. Control 7 (11) (2011) 6301–6318.
[37] P. Aarts, P. Veer, Neural network method for solving the partial differential equations, Neural Process. Lett. 14 (2001) 261–271.
[38] S. Yüzbas, M. Sezer, B. Kemancı, Numerical solutions of integro-differential equations and application of a population model with an improved Legendre method, Appl. Math. Model. 37 (2013) 2086–2101.
[39] P. Darania, A. Ebadian, A method for the numerical solution of the integro-differential equations, Appl. Math. Comput. 188 (2005) 657–668.
[40] E. Yusufoglu, Improved homotopy perturpation method for solving Fredholm type integro-differential equations, Chaos Solitons Fractals 41 (2009) 28–37.
[41] S.M. Hosseini, S. Shahmorad, Numerical solution of a class of integro-differential equations by the tau method with an error estimation, Appl. Math. Comput. 136 (2003) 559–570.
[42] M. Noor, S. Mohyud-Din, A reliable approach for higher-order integro-differential equations, Appl. Appl. Math. 3 (6) (2008) 188–199.
[43] M. Gachpazan, A. Kerayechian, H. zeidabadi, Finite element method for solving linear Volterra integro-differential equations of the second kind, J. Inform. Comput. Sci. 9 (4) (2014) 289–297.