Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

Numeric treatment of nonlinear second order multi-point boundary value problems using ANN, GAs and sequential quadratic programming technique

Zulqurnain Sabir[a*] and Muhammad Asif Zahoor Raja[b]

[a] *Department of Mathematics, Unique Science College (for girls) Dina, Pakistan*
[b] *Department of Electrical Engineering, COMSATS Institute of Information Technology, Attock Campus, Attock Pakistan*

CHRONICLE

Article history:
Received December 2 2013
Received in Revised Format Mach 17 2014
Accepted March 24 2014
Available online March 29 2014

Keywords:
Artificial neural networks
Genetic Algorithm
Sequential Quadratic Programming Technique
Hybrid computing techniques

ABSTRACT

In this paper, computational intelligence technique are presented for solving multi-point nonlinear boundary value problems based on artificial neural networks, evolutionary computing approach, and active-set technique. The neural network is to provide convenient methods for obtaining useful model based on unsupervised error for the differential equations. The motivation for presenting this work comes actually from the aim of introducing a reliable framework that combines the powerful features of ANN optimized with soft computing frameworks to cope with such challenging system. The applicability and reliability of such methods have been monitored thoroughly for various boundary value problems arises in science, engineering and biotechnology as well. Comprehensive numerical experimentations have been performed to validate the accuracy, convergence, and robustness of the designed scheme. Comparative studies have also been made with available standard solution to analyze the correctness of the proposed scheme.

1. Introduction

Multi-point boundary value problems (BVPs) or non-local boundary value problems were introduced by Ilin and Moiseev (1987). These BVPs have become the most important area among researchers. These days, because of its widespread applications in engineering as to model the physical problems, including vibration happening in a wire of uniform cross section and combined of material with changed densities, through porous media applications, BVPs are used in fluid flow and in elastic stability. In addition, these problems have wide applications in applied physics particularly in heat conduction (Cannon, 1984; Cannon et al., 1987) and other problems in nonlinear elasticity (Timoshenko, 1961). By using perturbation techniques, these problems have been normally solved but these techniques have some confines, e.g. the approximate solution consists of a series of very small parameters, which poses difficulty since the popularity of nonlinear problems has no minor parameters.

* Corresponding author. Tel: +923009521379
E-mail: zulqurnainsabir@gmail.com, uma_dina08@yahoo.com (Z. Sabir)

doi: 10.5267/j.ijiec.2014.3.004

These multi-point BVPs are generally limited to second order equation. Agarwal and Kinguradze (Das et al., 2010) solved higher order linear ordinary differential equation with singularity at numerous points in 2004. Cheung and Ren (Das et al., 2010) discussed multi-point BVPs in 2005. Tatari and Dehghan (Das et al., 2010) used Adomian decomposition method (ADM) to find the solution of multi-point BVP in 2006. Gupta (1992, 1994, 1997) is believed to be the first who used functional analytical technique to demonstrate the existence of the solution monitored by Eloe and Henderson (1994), Ma (1999, 2001), Liu (2002), Webb (2001), etc. Deimling, (1985), Guo and Lakshmikantham (1988) and Krasnosel'skij (1964) used commonly fixed-point theorem on cones for positive solution. The source to apply this theorem can be initiated with semi linear elliptic equations on annuli (Erbe & Wang, 1994).

Stochastic numerical techniques have been used extensively by many researchers in different fields to solve linear and nonlinear differential equations (Raja & Samar, 2012; Khan, 2009). Interest in this area is significantly enhanced with effective numerical treatment for fractional differential equation (Khan et al., 2012). The analysis is carried out further by constructing different types of neural networks models with/without satisfying the boundary conditions, exactly. Furthermore, the effect of change on number on neurons on accuracy and convergence has also been analyzed.

In the present study, computational intelligent techniques are developed for solving a nonlinear multi-point BVPs representing to model the equation with the help of universal approximation capability of feed-forward artificial neural networks (ANN) trained in an unsupervised manner with genetic algorithms (GA) technique, as a tool for effective global search optimizer, hybrid with sequential quadratic programming technique (SQPT), an efficient local search approach. The proposed schemes are evaluated on four variants of the problems to validate the correctness of the approaches. The reliability and effectiveness of the proposed scheme is analyzed based on sufficient large set of data generated from independent execution of the optimizers, i.e., GA, SQPT and GA-SQPT.

The organization of the paper is as follows: in the second section, the system model of the problem is given in details. Design methodology consists of feed-forward artificial neural networks modeling and learning procedure based on GA, SQPT and GA-SQPT is narrated in section three. Numerical experimentation of number of variant of the problem is provided in section four. Concluding remarks and few future research directions are provided in the last section.

2. General form of Multi-point BVPs

Mathematically, this problem is a second-order ordinary differential equation (ODE). The generic form or general form of multi-point boundary value problems is written as follows,

$$\begin{cases} \dfrac{d^2y}{dx^2} + h(y, y') = q(x), \quad 0 \le x \le 1, \\ y(0) = A \text{ and } \; y(1) = \displaystyle\sum_{i=1}^{m} \alpha_i y(\eta_i) + \gamma, \end{cases} \tag{1}$$

where $\eta_i \in (0,1), i = 0,1,2,..., m$, α and γ are constants

The Eq. (1) represents a standard form of multi-point BVP. The aim of studying the generalized form of the nonlinear multi-point BVP and its solutions is to develop a broader understanding of the general equations and to develop analytical tools to analyze these equations to cover more applications as they arise in the future.

3. Proposed Methodology

The brief description of designed methodology will be presented for the solution of the multi-point BVPs. In this section, the procedure consists of the development of two feed-forward unsupervised neural networks models of the equation. Moreover, brief introductory material for the optimization algorithms i.e. evolutionary computing and active set programming techniques are also given.

3.1 Neural Network Mathematical Modeling

Mathematical model for the second order multi-point boundary value problems is constructed with the strength of feed-forward ANN, in the form of following continuous mapping for the solution $y(x)$, and its first *dy/dx*, second d^2y/dx^2 respectively, For the activation function log-sigmoid function $g(x) = 1/(1+e^{-x})$ is used in the neural network modeling, given as:

$$\begin{cases} \hat{y}(x) = \sum_{i=1}^{k} \alpha_i g(w_i x + \beta_i) = \sum_{i=1}^{n} \alpha_i \dfrac{1}{1+e^{-(w_i x+\beta_i)}}, \\ \dfrac{d\hat{y}}{dx} = \sum_{i=1}^{k} \alpha_i \dfrac{d}{dx} g(w_i x + \beta_i) = \sum_{i=1}^{n} \alpha_i w_i \dfrac{e^{-(w_i x+\beta_i)}}{1+e^{-(w_i x+\beta_i)^2}}, \\ \dfrac{d^2\hat{y}}{dx^2} = \sum_{i=1}^{k} \alpha_i \dfrac{d^2}{dx^2} g(w_i x + \beta_i) = \sum_{i=1}^{n} \alpha_i w_i^2 \left(\dfrac{2e^{-2(w_i x+\beta_i)}}{\left(1+e^{-(w_i x+\beta_i)}\right)^3} - \dfrac{e^{-(w_i x+\beta_i)}}{\left(1+e^{-(w_i x+\beta_i)}\right)^2} \right), \end{cases} \tag{2}$$

where the vectors α, w and β are real-valued bounded unknown parameters, i.e., weights W = [α, w, β], *n* is the number of neurons and *g* is the activation function generally taken as log-sigmoid function.

The suitable combination of the networks given in the set (2) can be used to model the differential equation (1). An objective function or fitness function is developed in an unsupervised manner and it is defined by sum of two mean square error, e_1 is error function associated with differential equation (1) and it is given as:

$$\begin{cases} e = e_1 + e_2 \\ e_1 = \dfrac{1}{N} \sum_{m=1}^{N} \left(\dfrac{d^2\hat{y}_m}{dx^2} + h(\hat{y}_m, \hat{y}'_m) \dfrac{d\hat{y}_m}{dx} - q_m \right)^2, 0 \le x \le 1 \\ N = \dfrac{1}{h}, \hat{y}_m = \hat{y}(x_m), q_m = q(x_m), x_m = mh \end{cases} \tag{3}$$

Similarly, e_2 is the error function associated with boundary conditions for Eq. (1) is given as:

$$e_2 = \frac{1}{2}\left((\hat{y}_0 - A)^2 + (\hat{y}_N - B)^2 \right) \tag{4}$$

A second form of feed-forward artificial neural networks modeling is also constructed such that initial and boundary conditions are exactly satisfied in order to solve boundary value problems of the multi-point BVPs. The governing equations of alternative form of artificial neural networks model for the solution $y(x)$, and its first *dy/dx*, second d^2y/dx^2 respectively, written as:

$$\begin{cases} \breve{y}(x) = P(x) + Q(x)\hat{y}(x) \\ \dfrac{d\breve{y}}{dx} = \dfrac{dP}{dx} + \dfrac{dQ}{dx}\hat{y}(x) + Q(x)\dfrac{d\hat{y}}{dx} \\ \dfrac{d^2\breve{y}}{dx} = \dfrac{d^2P}{dx^2} + \dfrac{d^2Q}{dx^2}\hat{y}(x) + Q(x)\dfrac{d^2\hat{y}}{dx^2} \end{cases} \tag{5}$$

where $\hat{y}(x)$, $d\hat{y}/dx$, $d^2\hat{y}/dx^2$ are the networks as shown in Eq. (6). The updated neural network can be developed for solving the general nonlinear multi-point boundary value problem (1) that satisfies boundary conditions exactly by taking the approximate values of $P(x)$ and Q(x).

An objective function or fitness function is developed in an unsupervised manner for introducing error function base on the sum of mean square error as:

$$\begin{cases} \varepsilon = \dfrac{1}{N}\displaystyle\sum_{m=1}^{N}\left(\dfrac{d^2\tilde{y}_m}{dx^2} + h(\tilde{y}_m, \tilde{y}'_m) - q_m\right)^2, \ x \in (0,1) \\ N = \dfrac{1}{h}, \tilde{y}_m = \tilde{y}(x_m), q_m = q(x_m), x_m = mh \end{cases} \tag{6}$$

One can only deduce that with the availability of weights of neural networks such that the objective function given in Eq. (3) or Eq. (6) approaches zero, the proposed solution will approximate the exact solution of the problem closely.

3.2 Learning Methodology

The procedure for training the designed parameters of the networks is accomplished with the help of genetic algorithms (GAs), sequential quadratic programming technique (SQPT) and hybrid computing approach GA-SQPT. The recent researches in genetic algorithms (GAs) has emphasized that the initial proposals were found incompetent of solving tough problems in a valuable and efficient way. For a large numbers of optimization problems, the time of execution of first-generation GAs increases significantly while solution quality decreases. Moreover the things such as selection schemes, encoding procedures, and self-adaptive and knowledge based operators play an important role in the optimization of well convex and stochastic nature problems. Since the beginning of the GAs, it has used for optimization of different fields of interest. In addition to the optimization problems, it is also used as an effective technique for achieving modeling, forecasting, control and simulation objectives. Traditional optimization approach start with a single candidate and search iteratively for the optimal result by using static heuristics. While, the GA techniques uses a population of candidate to seek a number of areas of a solution space, simultaneously and adaptively. The most famous methods that go beyond easy local search are GAs, simulated annealing (SA), and tabu search (TS) etc. Genetic algorithms work on a population of individuals. All individual is a potential solution to a given problem and is normally encoded as a fixed-length binary string, which is an analogy with an actual chromosome. After an initial population is randomly or heuristically generated, the algorithm evolves population through sequential and iterative application of selection, crossover, and mutation operators. A new generation is shaped after every each iteration. The strongest feature of the GA is that do not get stuck in local minimum.

Sequential quadratic programming techniques is basically a local search method which belongs to the class of efficient constrained optimization technique. The method is based on mathematical model for solving the Karush-Kuhn-Tucker (KKT) equations to get the desired Lagrange multipliers simply and consequently the global candidate solution. The SQP technique has been used in many practical important optimization problems since their origin. The interesting reader is referred to these literatures to known the history, importance, subject term, working and detail procedural step of the algorithm. The necessary procedure adopted for GA-SQPT is given in following steps:

Step 1: *Initialization*: Initial values of parameters are set in this step with random assignment and declarations. These setting are also tabulated in Table1 for important parameter of GAs.

Step 2: *Fitness Evaluation*: Calculate the fitness of each individual or chromosome of population using the Eq. (2) and Eq. (5), for first and second type of modeling, respectively.

Step 3: *Termination Criteria:* Terminate the algorithm when either of following criteria matches:

- Predefined fitness values $e \leq 10^{-15}$ is achieved.

- Predefine number of generations are executed.
- Any of termination setting given in Table 1 for GA is fulfilled.

If termination criterion meets, then go to step 5

Step 4: *Reproduction:* Create next generation on the basis of Crossover: Call for scattered function, Mutation: Call for Adaptive Feasible function, Selection: Call for Stochastic Uniform function and Elitism account is set 4 etc.

Repeat the procedure from step 2 to step 4 with newly produced population

Step 5: *Improvements:* Sequential quadratic programming technique is used for further refinement of results by taking final adaptive weights of GAs as initial weights (start point) of SQPT algorithm. SQPT is applied as per setting of parameters given in Table 2. Store also the refined final weights of the algorithm.

Step 6: *Neurons Analysis*: Repeat step 1 to 5 for by taking size of initial weights, i.e. 15, 30 and 45 for k = 5, 10 and 15 neurons, respectively. These results are used for detail analysis of algorithm later.

The parameter setting used in this paper for GA algorithm in table 1 and SQPT algorithm are given in Table 2.

Table 1
Parameters Setting for GA in simulations

Parameters	Setting	Parameters	Setting
Population Creation	Linear feasible	Bounds	[-20, 20]
Initial Population range	[-1, 1]	Derivative	Approximate by solver
Population	300 individuals	TolFun	10^{-20}
Generations	700	StallGenLimit	100
Scaling faction	Rank	Mutation	Adaptive feasible
Selection function	Stochastic Uniform	Migration interval	20
Crossover fraction	0.80	Elite Count	02
Crossover function	Heuristic	TolCon	10^{-20}
Migration fraction	0.2	Other	Defaults

Table 2
Parameter Settings of "fmincon" function for SQPT in simulations

Parameters	Settings/Values	Parameters	Settings/Values
Algorithm	active-set	Initial Weights	best weights of GA
FinDiffType	Central	Initial Weight Vector length	30
TolX	10^{-20}	Total Initial Weight Vectors	100
Iterations	700	Bounds	a_i, w_i, b_i ϵ (±20) $\forall$ i 1 to n
MaxFunEvals	200000	TolCon	Zero
TolFun	10^{-20}	Derivative	Approximate by solver
TolCon	0	Other	as defaults

3. Simulations and Results

In this section, three problems of multi-point boundary value problems will be presented with their multi-point boundary conditions. We compare the results of all the numerical problems for different number of neurons with exact solutions that will be provided with each problem. The error analysis showed graphically the high reliability of purposed scheme.

Problem I: consider the following nonlinear multi-point boundary value problem (Das et al., 2010)

$$\begin{cases} \dfrac{d^2y}{dx^2}+\dfrac{x^2(1-x)}{2}\dfrac{dy}{dx}-y^2(x)=x^3+2 \\ y(0)=0, \quad \mathrm{y}(1)=\displaystyle\sum_{i=0}^{4}\left(\frac{1}{1+i}\right)y\left(\frac{i}{5}\right)+0.708667 \end{cases} \tag{7}$$

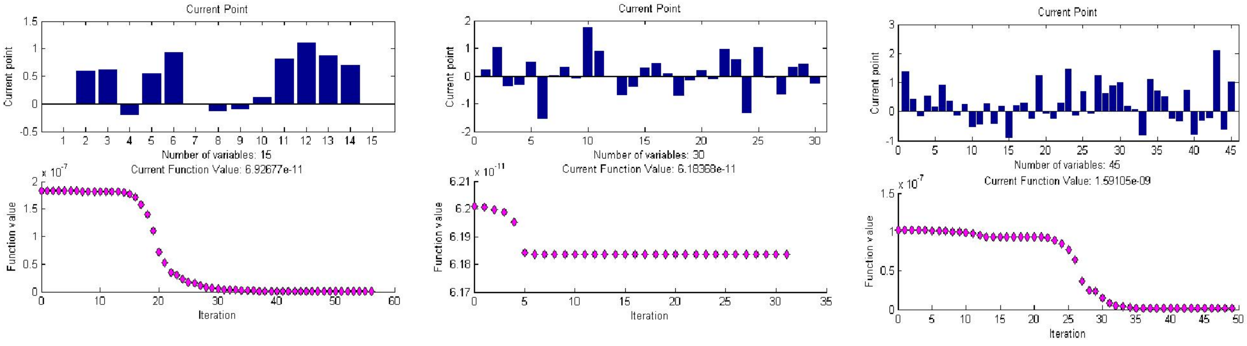
The exact solution of the problem is given as:

$$y(x)=x^2 \tag{8}$$

The proposed methodologies are applied to solve the problem by taking 10 neurons in the hidden layer of neural networks thus with a total 30 design parameter or weights, $W(\alpha,w,\beta)$.The fitness functions e and ε is formulated for this case for r ϵ (0, 1] with a step size h = 0.1 as:

$$\begin{cases} e=\dfrac{1}{10}\displaystyle\sum_{m=1}^{10}\left(\frac{d^2\hat{y}_m}{dx^2}+\frac{x_m^2(1-x_m)}{2}\frac{d\hat{y}_m}{dx}-\hat{y}_m^2-x_m^3-2\right)^2 \\ \quad +\dfrac{1}{2}\left((\hat{y}_0)^2+\left(\hat{y}_{10}-\displaystyle\sum_{i=0}^{4}\left(\frac{1}{1+i}\right)y\left(\frac{i}{5}\right)+0.708667\right)^2\right) \\ \varepsilon=\dfrac{1}{10}\displaystyle\sum_{m=1}^{10}\left(\frac{d^2\tilde{y}_m}{dx^2}+\frac{x_m^2(1-x_m)}{2}\frac{d\tilde{y}_m}{dx}-\tilde{y}_m^2-x_m^3-2\right)^2 \end{cases} \tag{9}$$

Set of trained weights with respective fitness for GA-SQPT algorithm are shown in Fig. 1



(a) (b) (c)

Fig. 1. Set of optimal weights along with parameters for problem 1 by GA-SQPT using 5, 10 and 15 numbers of neurons

The proposed solution $\hat{y}(x)$ and $\tilde{y}(x)$ of the Eq. (7) is obtained using the optimal weights of algorithms in first equation of the set (2) and (5), respectively. The results due to GA-SQPT for the case $k = 5$ are given in simplify form as:

$$\begin{aligned}\hat{Y}_{GA-SQPT}(x)=&\frac{-3.67842896132752}{1+\mathrm{e}^{-(2.34877564743t+3.219305585838)}}-\frac{1.14752400515832}{1+\mathrm{e}^{-(4.4035182757t-0.46497078475872)}}\\&-\frac{2.39639235002620}{1+\mathrm{e}^{-(-0.4350829368t-2.29597874248666)}}-\frac{2.35054880126743}{1+\mathrm{e}^{-(0.52111789400447t-1.78365289124987)}}-\frac{2.29409857048802}{1+\mathrm{e}^{-(2.0465687282029t-4.25093661709093)}}\end{aligned} \tag{10}$$

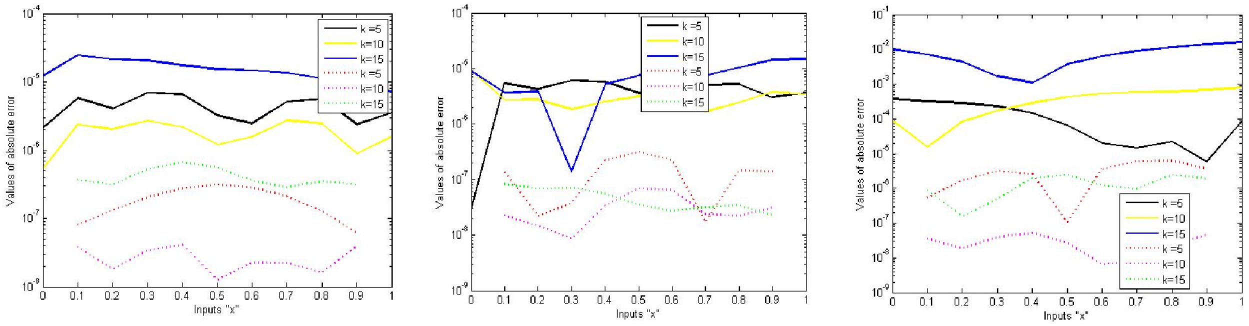
$$\begin{aligned}\tilde{y}_{GA-SQP}(x)=&\ x+x(x-1)\\&[\frac{-0.00916357067839}{1+\mathrm{e}^{-(0.5913937719t-0.617416045561)}}-\frac{-0.20873044942255}{1+\mathrm{e}^{-(0.53952271524t-0.93546047665)}}\\&-\frac{0.00455641085491}{1+\mathrm{e}^{-(-0.13418836737t+-0.10888915790)}}+\frac{0.10796315561423}{1+\mathrm{e}^{-(0.81207439935t+1.10333363137)}}+\frac{0.8652393033873}{1+\mathrm{e}^{-(0.69277039883t-0.02118837096)}}]\end{aligned} \tag{11}$$

To find the solution of the problems, we have applied the GA, SQPT and hybrid approach GA-SQPT by invoking the Matlab built in functions using the parameters setting as given in Table 1 and Table 2.

The results calculated for the both neural networks models optimized with GA-SQPT for different number of neurons, i.e., *k* = 5, 10 and 15, are given in the Table 3 for inputs t between 0 and 1 with a step size of 0.1. The results obtained with exact solution given in Eq. (8) are also given in the table for the same inputs. It is seen that the proposed solution match with exact solution to five to seven decimal space of the accuracy. The values of absolute error (AE) are determined for each algorithm, i.e., GA, SQPT and GA-SQPT, for both models and results are shown graphically in Fig 2

Table 3
Proposed results of the problem 1 using neural networks models with different neurons optimized with GA-SQPT

X	$Y(x)$ Exact		$\tilde{y}(x)$			$\hat{y}(x)$	
		GA	*SQPT*	*GA-SQPT*	*GA*	*SQPT*	*GA-SQPT*
0.00	0	0	0	0	0	0	0
0.10	0.01	0.010005785	0.010002356	0.010024446	0.010000081	0.01000	0.010000
0.20	0.04	0.040004076	0.040002018	0.040021325	0.040000132	0.04000	0.040000
0.30	0.09	0.09000689	0.090002685	0.090020551	0.090000205	0.09000	0.089999
0.40	0.16	0.160006592	0.160002162	0.160017598	0.160000277	0.16000	0.159999
0.50	0.25	0.250003202	0.250001204	0.250015241	0.25000031	0.25000	0.249999
0.60	0.36	0.360002446	0.36000157	0.360014615	0.360000284	0.36000	0.360000
0.70	0.49	0.490005155	0.490002722	0.490013618	0.490000212	0.49000	0.490000
0.80	0.64	0.640005664	0.640002441	0.640011193	0.640000127	0.64000	0.640000
0.90	0.81	0.810002392	0.810000901	0.810008764	0.810000061	0.81000	0.810000
1.00	1	1.000003505	1.0000016	1.000007151	1	1	1



(a) (b) (c)

Fig. 2. Results based on AE for two models in case of problem I (solid and dotted lines for neural network with/without satisfying boundary condition exactly) (a) for GA (b) for SQP and (c) for GA-SQP

It is seen that the values of AE are determined for each algorithm, i.e., GA, SQPT and GA-SQPT, for both models and results are shown graphically in Fig 3. In 3(a), the values of AE for GA lies in the range 10^{-05} to 10^{-06}, 10^{-05} to 10^{-06}, and 10^{-05} to 10^{-06} for k=5, 10 and 15 respectively using first fitness function, while for fitness second function these values lies in the range 10^{-07} to 10^{-08}, 10^{-06} to 10^{-07}, and 10^{-05} to 10^{-06} for k = 5, 10 and 15, respectively. Similarly it is inferred from Fig 3(b) the value of AE for SQPT lies in the range 10^{-05} to 10^{-07}, 10^{-05} to 10^{-06}, and 10^{-05} to 10^{-06} for k = 5, 10 and 15 respectively using first fitness function, while for second fitness function these values lies in the range 10^{-06} to 10^{-07} 10^{-06} to 10^{-07}, and 10^{-07} to 10^{-08} for k = 5, 10 and 15, respectively. Whereas, from Fig 3(c) the values of AE for GA-SQPT lies in the range 10^{-02} to 10^{-05}, 10^{-02} to 10^{-03}, and 10^{-03} to 10^{-05} for k = 5, 10 and 15 respectively using first fitness function, while for second fitness function these values lies in the range 10^{-06} to 10^{-08}, 10^{-05} to 10^{-07}, and 10^{-06} to 10^{-07} for k = 5, 10 and 15, respectively.

Problem II: Consider the following nonlinear multi-point boundary value problem (Das et al., 2010)

$$\begin{cases} \dfrac{d^2y}{dx^2}+xy(x)\dfrac{dy}{dx}-2y^2(x)=x^3-x^2+2 \\ y(0)=0, \quad \mathrm{y}(1)=\displaystyle\sum_{i=0}^{4}\left(\frac{1}{1+i}\right)y\left(\frac{i}{5}\right)+0.252 \end{cases} \tag{12}$$

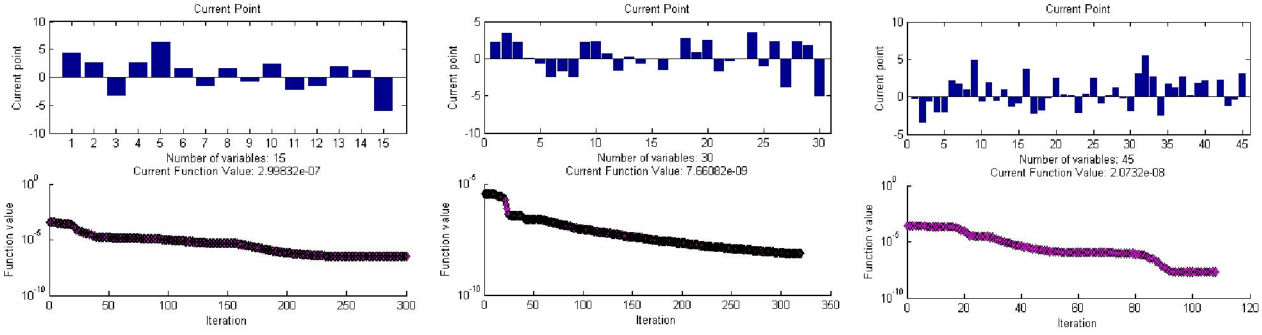
The exact solution of the problem is given as:

$$y(x)=x(x-1) \tag{13}$$

The proposed methodologies are applied to solve the problem by taking 10 neurons in the hidden layer of neural networks thus with a total 30 design parameter or weights, $\mathbf{W}\ (\alpha, w, \beta)$. The fitness functions e and ε is formulated for this case for $r \in (0, 1]$ with a step size $h = 0.1$ as:

$$\begin{cases} e=\dfrac{1}{10}\displaystyle\sum_{m=1}^{10}\left(\frac{d^2\hat{y}_m}{dx^2}+x_m\hat{y}_m\frac{d\hat{y}_m}{dx}-2\hat{y}_m^2-x_m^3-x_m^2-2\right)^2 \\ \quad +\dfrac{1}{2}\left((\hat{y}_0)^2+\left(\hat{y}_{10}-\displaystyle\sum_{i=0}^{4}\left(\frac{1}{1+i}\right)y\left(\frac{i}{5}\right)+0.252\right)^2\right) \\ \varepsilon=\dfrac{1}{10}\displaystyle\sum_{m=1}^{10}\left(\frac{d^2\tilde{y}_m}{dx^2}+x_m\tilde{y}_m\frac{d\tilde{y}_m}{dx}-2\tilde{y}_m^2-x_m^3-x_m^2-2\right)^2 \end{cases} \tag{14}$$

Set of trained weights with respective fitness for GA-SQPT algorithm are shown in Fig. 3



(a) (b) (c)

Fig. 3. Set of optimal weights along with parameters for problem 1 by GA-SQPT using 5, 10 and 15 numbers of neurons

The results due to GA-SQPT for the case $k = 5$ are given in simplify form as:

$$\begin{aligned} \hat{Y}_{GA-SQP}(x)=\ &\frac{2.208836555}{1+\mathrm{e}^{(3.346487248t+2.243395633)}}+\frac{0.071721746}{1+\mathrm{e}^{(-0.763290739t-2.46820952)}}-\frac{1.810734908}{1+\mathrm{e}^{(-2.483154936t+2.243058379)}} \\ &-\frac{2.349373568}{1+\mathrm{e}^{(0.636566587t-1.617782428)}}-\frac{0.209680387}{1+\mathrm{e}^{(-0.775217641t+0.026655226)}}-\frac{1.522652644}{1+\mathrm{e}^{(-0.111253639t+2.697695214)}} \\ &+\frac{0.789955082}{1+\mathrm{e}^{(2.417955151t-1.836772022)}}-\frac{0.394007752}{1+\mathrm{e}^{(-0.005227503t+3.564844228)}}-\frac{1.128676534}{1+\mathrm{e}^{(2.389719539t-3.905731664)}} \\ &+\frac{2.34355798}{1+\mathrm{e}^{(1.69498294t-5.121411805)}} \end{aligned} \tag{15}$$

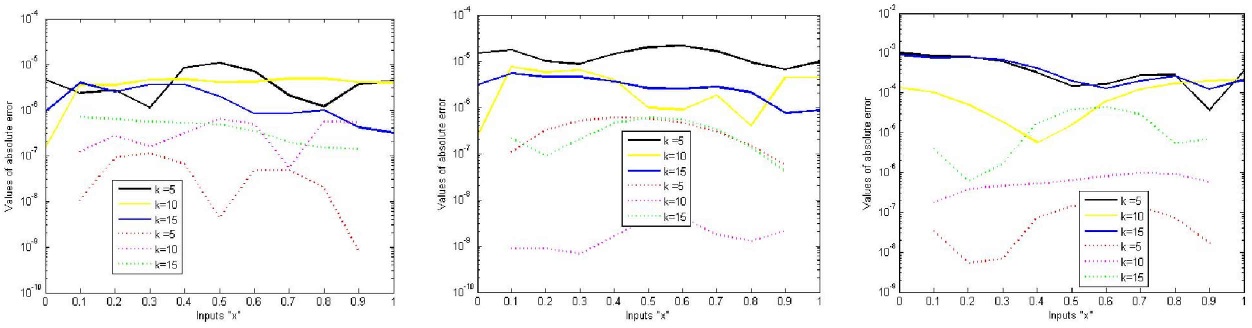
$$\tilde{y}_{GA-SQP}(x) = x^2 - x + x(x-1)$$
$$\Big[\frac{-0.216506509}{1+e^{(0.472487834t-0.246514322)}} - \frac{0.109795413}{1+e^{(0.095063987t-0.487915835)}}$$
$$-\frac{0.482970033}{1+e^{(-0.305821484t+0.263233801)}} - \frac{0.018294812}{1+e^{(0.944712492t+0.484968391)}}$$
$$-\frac{0.230717011}{1+e^{(0.316310172t-0.346091359)}} + \frac{0.426353657}{1+e^{(-0.201150105t+0.031234018)}} \qquad (16)$$
$$-\frac{0.427353059}{1+e^{(-0.69348194t+0.274893474)}} + \frac{0.28357977}{1+e^{(0.842677833t+0.550224316)}}$$
$$-\frac{0.230940307}{1+e^{(-0.576745358t--0.000596585)}} - \frac{0.250772371}{1+e^{(0.529882133t-0.372938855)}}\Big]$$

The results obtained for the both neural networks models of GA-SQPT for k = 5, 10 and 15, are also presented in the Table 4 along with exact solution using Eq. (13).

Table 4
Proposed results of the problem using neural networks models with different neurons optimized with GA-SQPT

X	$y(x)$		$\tilde{y}(x)$			$\hat{y}(x)$	
	Exact	k=5	k=10	k=15	k=5	k=10	k=15
0	0	0	0	0	0	0	0
0.1	-0.09	-0.08999763	-0.08999647	-0.08999603	-0.08999998	-0.08999988	-0.08999929
0.2	-0.16	-0.16000270	-0.15999643	-0.15999751	-0.15999990	-0.160000271	-0.15999938
0.3	-0.21	-0.20999887	-0.20999540	-0.20999636	-0.20999988	-0.210000158	-0.20999944
0.4	-0.24	-0.23999158	-0.23999535	-0.23999641	-0.23999993	-0.23999969	-0.23999948
0.5	-0.25	-0.24998915	-0.24999592	-0.24999801	-0.25000000	-0.249999367	-0.24999952
0.6	-0.24	-0.23999295	-0.23999582	-0.23999915	-0.24000004	-0.239999515	-0.23999965
0.7	-0.21	-0.20999795	-0.20999516	-0.20999915	-0.21000004	-0.210000055	-0.20999980
0.8	-0.16	-0.15999881	-0.15999515	-0.15999901	-0.16000002	-0.160000553	-0.15999985
0.9	-0.09	-0.08999632	-0.08999595	-0.08999958	-0.08999999	-0.090000531	-0.0899998
1	0	0	0	0	0	0	0

The values of absolute error AE are determined for each algorithm, i.e., GA, SQPT and GA-SQPT for both models and results are shown graphically in Fig 4.



(a) (b) (c)

Fig. 4. Results based on AE for two models in case of problem II (solid and dotted lines for neural network with/without satisfying boundary condition exactly) (a) for GA (b) for SQPT and (c) for GA-SQPT

Problem III: Consider the following nonlinear multi-point boundary value problem (Das et al., 2010)

$$\begin{cases} \dfrac{d^2y}{dx^2}+y(x)\dfrac{dy}{dx}=\left(\cos x-1\right)\sin x \\ y(0)=0, \quad \mathrm{y}(1)=\sum\limits_{i=0}^{4}\left(\dfrac{1}{1+i}\right)y\left(\dfrac{i}{5}\right)+0.3277 \end{cases} \tag{17}$$

The exact solution of the problem is given as:

$$y(x)=\sin x \tag{18}$$

The proposed methodologies are applied to solve the problem by taking 10 neurons in the hidden layer of neural networks thus with a total 30 design parameter or weights, $\mathbf{W}\ (\alpha, w, \beta)$. The fitness functions e and ε is formulated for this case for $r \in (0, 1]$ with a step size $h = 0.1$ as:

$$\begin{cases} e=\dfrac{1}{10}\sum\limits_{m=1}^{10}\left(\dfrac{d^2\hat{y}_m}{dx^2}+\hat{y}_m\dfrac{d\hat{y}_m}{dx}-(\cos x_m-1)\sin x_m\right)^2 \\ \quad+\dfrac{1}{2}\left(\left(\hat{y}_0\right)^2+\left(\hat{y}_{10}-\sum\limits_{i=0}^{4}\left(\dfrac{1}{1+i}\right)y\left(\dfrac{i}{5}\right)+0.3277\right)^2\right) \\ \varepsilon=\dfrac{1}{10}\sum\limits_{m=1}^{10}\left(\dfrac{d^2\tilde{y}_m}{dx^2}+\tilde{y}_m\dfrac{d\tilde{y}_m}{dx}-(\cos x_m-1)\sin x_m\right)^2 \end{cases} \tag{19}$$

The proposed solution $\hat{y}(x)$ and $\tilde{y}(x)$ of the Eq. (17) is obtained using the optimal weights of algorithms in first equation of the set (2) and (5), respectively. The results due to GA-SQP for the case $k = 15$ are given in simplify form as:

$$\begin{aligned} \hat{Y}_{GA-AST}(x)=&\frac{-0.081445879}{1+\mathrm{e}^{(-2.768026281t-0.07889657)}}+\frac{1.033908368}{1+\mathrm{e}^{(1.577421678t+0.764886671)}}+\frac{1.99603985}{1+\mathrm{e}^{(-2.261775246t+1.861122461)}}-\frac{1.159325859}{1+\mathrm{e}^{(-0.069874053t-1.296706406)}} \\ &-\frac{0.298371076}{1+\mathrm{e}^{(-0.764074296t-0.48929052)}}-\frac{2.389750629}{1+\mathrm{e}^{(1.329328249t-1.775691856)}}-\frac{0.408425054}{1+\mathrm{e}^{(-0.452390437t-0.437945435)}}+\frac{1.081451478}{1+\mathrm{e}^{(-0.69183918t-0.116530885)}} \\ &-\frac{1.118420805}{1+\mathrm{e}^{(-0.337217405t-1.205051142)}}+\frac{0.4761533}{1+\mathrm{e}^{(0.335142006t+0.432716274)}}-\frac{0.899806552}{1+\mathrm{e}^{(-3.539588497t-0.950303091)}}+\frac{0.418222187}{1+\mathrm{e}^{(0.200102379t-0.184532206)}} \\ &-\frac{0.262434241}{1+\mathrm{e}^{(0.608412449t+0.339126605)}}+\frac{0.337546906}{1+\mathrm{e}^{(-2.253966554t-0.045011958)}}+\frac{0.18163671}{1+\mathrm{e}^{(0.432918626t+0.136772046)}} \end{aligned} \tag{20}$$

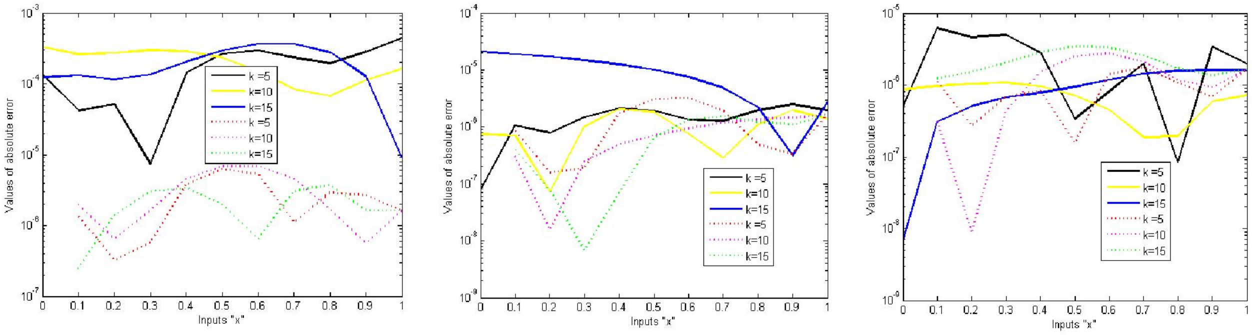
$$\begin{aligned} \tilde{y}_{GA-SQP}(x)=x^2-0.158527384x+x(x-1)=[&\frac{0.564010455}{1+\mathrm{e}^{(0.758597564t-0.91034693)}}+\frac{0.534950738}{1+\mathrm{e}^{(-0.324178311t-0.361743894)}} \\ &-\frac{0.332320715}{1+\mathrm{e}^{(-0.257771472t-0.794779984)}}+\frac{0.392079392}{1+\mathrm{e}^{(0.011730424t-0.310811717)}}+\frac{0.126893795}{1+\mathrm{e}^{(0.021418674t-0.927117635)}}-\frac{0.666059302}{1+\mathrm{e}^{(-0.219926712t0.695773998)}} \\ &-\frac{0.438332404}{1+\mathrm{e}^{(-0.321418936t+0.273684792)}}-\frac{0.337776689}{1+\mathrm{e}^{(-0.278016426t-0.166833512)}}-\frac{0.246601324}{1+\mathrm{e}^{(-0.075388873t-0.250126438)}}+\frac{0.350453409}{1+\mathrm{e}^{-(1.171431708t-0.507170459)}} \\ &-\frac{0.128862184}{1+\mathrm{e}^{(-0.406160924t-0.195300395)}}-\frac{0.265027029}{1+\mathrm{e}^{(0.47739276t+0.196942455)}}+\frac{0.596502792}{1+\mathrm{e}^{(0.226194573t-0.353944167)}}+\frac{0.453471338}{1+\mathrm{e}^{(0.527981159t+0.856344939)}} \\ &+\frac{0.773546968}{1+\mathrm{e}^{(1.62352718t+0.590627788)}}] \end{aligned} \tag{21}$$

The results obtained for the both neural networks models of GA-SQPT for k = 5, 10 and 15, are also presented in the Table 4 along with exact solution using Eq. (18). The proposed solution approximately matches with exact solution up to four to six decimal places of the accuracy.

Table 4
Proposed results of the problem using neural networks models with different neurons optimized with GA-SQPT

X	$y(x)$	$\tilde{y}(x)$			$\hat{y}(x)$		
	Exact	k =5	k=10	k=15	k=5	k=10	k=15
0	0	0	0	0	0	0	0
0.1	0.099833	0.099827	0.099832	0.099834	0.099832	0.099834	0.099835
0.2	0.198669	0.198665	0.198668	0.19867	0.198669	0.198669	0.198671
0.3	0.29552	0.295515	0.295519	0.295521	0.29552	0.295521	0.295522
0.4	0.389418	0.389416	0.389417	0.389419	0.389418	0.38942	0.389421
0.5	0.479426	0.479425	0.479425	0.479426	0.479426	0.479428	0.479429
0.6	0.564642	0.564642	0.564642	0.564644	0.564644	0.564645	0.564646
0.7	0.644218	0.644216	0.644217	0.644219	0.644219	0.64422	0.64422
0.8	0.717356	0.717356	0.717356	0.717358	0.717357	0.717357	0.717358
0.9	0.783327	0.78333	0.783327	0.783329	0.783328	0.783328	0.783328
1	0.841471	0.841473	0.841472	0.841473	0.841473	0.841473	0.841473

The values of absolute error AE are determined for each algorithm, i.e., GA, SQP and GA-SQP for both models and results are shown graphically in Fig 5 as follows,



(a) (b) (c)

Fig. 5. Results based on AE for two models in case of problem III (solid and dotted lines for neural network with/without satisfying boundary condition exactly) (a) for GA (b) for SQPT and (c) for GA-SQPT

4. Conclusions

Following conclusions are drawn for our research studies:

- A new soft computing approach has been developed effectively for solving multi-point boundary value problems, in-particularly multi-point BVPs and its variants using neural networks optimized with genetic algorithm, sequential quadratic programming technique and their hybrid combination.
- The sufficient low value of absolute error has established the correctness of the designed scheme.
- By taking fewer number of neurons, the performance of method were efficient for finding the solution, however, it increased the number of neurons, one can get slightly more accurate solution with immense reliability but at the cost of more computations.

Future research directions

Following are few suggested research directions for interested readers:

- One can look for more accurate neural networks modeling by using alternate activation functions like radial basis, Mexican Hat, wavelets hat, etc.
- Change of optimization algorithm can also help us improve the accuracy and the convergence of results. In this regards, particle swarm optimization, differential evolution, genetic programming, hill climbing etc., are possible choices.

- Better hardware platform like parallel, grid and cloud computing, can help in decreasing computational time of the algorithm, considerably. Consequently, we may look for other complicated nonlinear problems to solve by our design scheme.

Acknowledgement

Authors like to thank the anonymous referees for constructive comments on this paper.

References

Cannon, J. R. (1984). The one-dimensional heat equation, volume 23 of Encyclopedia of Mathematics and its Applications.

Cannon, J. R., Esteva, S. P., & Van Der Hoek, J. (1987). A Galerkin procedure for the diffusion equation subject to the specification of mass. *SIAM Journal on Numerical Analysis*, *24*(3), 499-515.

Deimling, K. (1985). Nonlinear Functional Analysis. *Springler-Verlag, Berlin*.

Das, S., Kumar, S., & Singh, O. P. (2010). Solutions of Nonlinear Second Order Multi-point Boundary Value Problems by Homotopy Perturbation Method. *Applied Mathematics: An International Journal*, *5*(10), 1592-1600.

Erbe, L. H., & Wang, H. (1994). On the existence of positive solutions of ordinary differential equations. *Proceedings of the American Mathematical Society*, *120*(3), 743-748.

Gupta, C. P. (1992). Solvability of a three-point nonlinear boundary value problem for a second order ordinary differential equation. *Journal of Mathematical Analysis and Applications*, *168*(2), 540-551.

Gupta, C. P. (1994). A note on a second order three-point boundary value problem. *Journal of Mathematical Analysis and Applications*, *186*(1), 277-281.

Gupta, C. P., & Trofimchuk, S. I. (1997). A sharper condition for the solvability of a three-point second order boundary value problem. *Journal of Mathematical Analysis and Applications*, *205*(2), 586-597.

Guo, D., & Lakshmikantham, V. (1988). *Nonlinear Problems in Abstract Cones* (Vol. 5). San Diego: Academic press.

Eloe, P. W., & Henderson, J. (1994). Multipoint boundary value problems for ordinary differential systems. *Journal of Differential Equations*, *114*(1), 232-242.

Ilin, V. A., & Moiseev, E. I. (1987). Nonlocal boundary-value problem of the first kind for a Sturm-Liouville operator in its differential and finite difference aspects. *Differential Equations*, *23*(7), 803-810.

Khan, J. A., Zahoor, R. M. A., & Qureshi, I. M. (2009, December). Swarm intelligence for the solution of problems in differential equations. In *Environmental and Computer Science, 2009. ICECS'09. Second International Conference on* (pp. 141-147). IEEE.

Khan, A., Raja, M. A. Z., & Ijaz Mansoor Qureshi, J. (2012). An application of evolutionary computational technique to non-linear singular system arising in polytrophic and isothermal sphere. *Global Journal of Researches In Engineering*, *12*(1-I).

Krasnosel'skij, M. A. (1964). *Positive solutions of operator equations* (p. 381). L. F. Boron (Ed.). Groningen: P. Noordhoff.

Liu, B. (2002). Positive solutions of a nonlinear three-point boundary value problem. *Computers & Mathematics with Applications*, *44*(1), 201-211.

Ma, R. (1999). Positive solutions of a nonlinear three-point boundary-value problem. *Electronic Journal of Differential Equations*, *1999*(34), 1-8.

Ma, R. (2001). Positive solutions for second-order three-point boundary value problems. *Applied Mathematics Letters*, *14*(1), 1-5.

Raja, M. A. Z., & Samar, R. (2012). Neural network optimized with evolutionary computing technique for solving the 2-dimensional Bratu problem. *Neural Computing and Applications*, 1-12.

Timoshenko, S. P., & Gere, J. M. (1961). Theory of elastic stability. 1961.

Webb, J. R. L. (2001). Positive solutions of some three point boundary value problems via fixed point index theory. *Nonlinear Analysis: Theory, Methods & Applications*, *47*(7), 4319-4332.