Journal of Interdisciplinary Mathematics
ISSN: 0972-0502 (Print), ISSN: 2169-012X (Online)
Vol. 28 (2025), No. 1, pp. 139–146
DOI : 10.47974/JIM-1804

Designing feed forward fully fuzzy neural network to solve fuzzy singular perturbed Volterra integro-differential equations

Ghassan Salih Salman †
Khalid Mindeel Mohammed Al-Abrahemee *
Department of Mathematics
College of Education
AL-Qadisiyah University
AL-Qadisiyah 58002
Iraq

Abstract

Recently, the study of singular Volterra integro-differential equations has been of increasing interest for a long time. Our paper has a design for a fast feed-forward neural network to adduce a new method for solving one-dimensions fuzzy singular perturbed integro-differential equations. Employing a multi-layer that has one hidden layer with five units and one linear output unit. And the sigmoid activation for every unit is the hyperbolic tangent function and the Levenberg-Marquardt training algorithm.We compared our exact solution in illustrative examples with the results of numerical experiments, confirming the efficiency and accuracy of our presented scheme.

Subject Classification (2010): *34K28.*

Keywords: *Singular perturbed volterra integro-differential equations, Feed forward neural network, Hyperbolic tangent function, Fuzzy sets.*

1. Introduction

It is known that a huge amount of work is done for analyzing and improving numerical methods to solve one-dimensional integro-differential equation In this paper, we contemplate the one-dimensional singular perturbed fuzzy Volterra integro-differential equations

$$\varepsilon\mu'(x) = \mathrm{F}(\mu, x, \varepsilon) + \int_0^{\mathrm{x}} \Phi(x, \mathrm{t})\mu(\mathrm{t})\mathrm{dt},\ x \in [\mathrm{a}, \mathrm{b}]$$

† *ORCID-ID:* 0000-0003-3841-1465
* *ORCID-ID:* 0000-0002-4705-6349

† *E-mail:* qayrawan2007@gmail.com
* *E-mail:* Khalid.mohammed@qu.edu.iq (Corresponding Author)

where ε represents a fixed perturbation parameter such that $0<\varepsilon\leq 1$.

Here, ε is the tiny parameter that gives to the problem its singularly perturbed nature; the kernel Φ and the data function F are two known smooth functions. Beneath suitable status for F and Φ, for each $\varepsilon > 0$, the equation above has just one continuous solution on $[0, x]\times[0, y]$, [1-8].

Many methods have been developed to find solutions to integral and integro-differential equations. Some of these methods used basis functions to express the analytic form of the solution to transform the original problem, usually into an algebraic equation system. The techniques that we discuss now are artificial neural networks originally inspired by the operatizing of the human brain combined with intelligence. These are the ingredients to make a great number of artificial neurons that can be processed in an aligned way, so any problems related to aspects of classification can be solved now by the architecture of any identified ANN. Based on its position on the network, there are three types of computational neurons: input, output and hidden [9-13].

2. A Fuzzy Neural Network Structure

The construction of the model displays how the feed-forward neural network transforms the n inputs $(x_1, x_2, \dots x_i, \dots x_n)$ into the s fuzzy outputs:

$([O_1]_\gamma [O_2]_\gamma, [O_3]_\gamma, \dots, [O_k]_\gamma, \dots, [O_s]_\gamma)$ all around the m hidden fuzzy neurons $([h_{d1}]_\gamma, [h_{d2}]_\gamma, [h_{d3}]_\gamma, \dots, [h_{dj}]_\gamma, \dots, [h_{dm}]_\gamma)$ such that $[q_j]_\gamma$ and $[z_k]_\gamma$ are fuzzy biases for fuzzy neurons $[h_{dj}]_\gamma, [O]_\gamma$ respectively, $[w_{ji}]_\gamma$ are the fuzzy weights that connects crisp neuron x_i to fuzzy neuron $[h_{dj}]_\gamma$, along with $[v_{kj}]_\gamma$ are the fuzzy weight that connects fuzzy neuron $[h_{dj}]_\gamma$ to fuzzy neuron $[O]_\gamma$.

Input unit

$$x = x_i, i = 1,2,3,4\dots, n$$

Hidden unit:

$$[h_{dj}]_\gamma = [[h_{dj}]^l_\gamma, [h_{dj}]^u_\gamma] = \Gamma([N_{tj}]_\gamma) = [\Gamma[N_{tj}]^l_\gamma, \Gamma[[N_{tj}]^u_\gamma]$$

$$[h_{dj}]_\gamma = \Gamma([N_{tj}]_\gamma), j = 1,2,3,\dots, m \text{ Where}$$

$$[N_{tj}]^l_\gamma = \sum_{i=1}^{n} x_i \, [w_{ji}]^l_\gamma + [q_j]^l_\gamma$$

$$[N_{tj}]^u_\gamma = \sum_{i=1}^{n} x_i \, [w_{ji}]^u_\gamma + [q_j]^u_\gamma$$

Output unit:

$$[O]_\gamma = [[O]^l_\gamma, [O]^u_\gamma] = \Gamma([N_{tjk}]_\gamma), k = 1,2,3,\dots, s \text{ Where}$$

$$[N_{tj}]^l_\gamma = (\sum_{j\in a}[v_{kj}]^l_\gamma [h_{dj}]^l_\gamma + \sum_{j\in b}[v_{kj}]^l_\gamma [h_{dj}]^u_\gamma) + [z_k]^l_\gamma$$

$$[N_{tj}]^u_\gamma = (\sum_{j\in c}[v_{kj}]^u_\gamma [h_{dj}]^u_\gamma + \sum_{j\in d}[v_{kj}]^u_\gamma [h_{dj}]^l_\gamma) + [z_k]^u_\gamma$$

Such that $[h_{dj}]^u_\gamma \geq [h_{dj}]^l_\gamma \geq 0$ where

$$a = \{ j: [v_{kj}]^l_\gamma \geq 0 \}, b = \{ j: [v_{kj}]^l_\gamma < 0 \},$$

$$c = \{ j: [v_{kj}]^u_\gamma \geq 0 \}, d = \{ j: [v_{kj}]^u_\gamma < 0 \}$$

3. The method illustration

Consider the first order of one dimension fuzzy singularly perturbed Volterra integro-differential equation for ODEs

$$\varepsilon\mu'(\mu) = \mathrm{F}(\mu, x, \varepsilon) + \int_0^{\mathrm{x}} \Phi(x, \mathrm{t})\mu(\mathrm{t})\mathrm{dt}\ , x \in [\mathrm{a}, \mathrm{b}]$$

where ε is perturb paremter (0<ε<<1)

With the fuzzy (BC): $\mu(a) = [A]_\gamma, \mu(b) = [B]_\gamma$,such that $[\mathrm{A}]_\gamma$ and $[\mathrm{B}]_\gamma$ are fuzzy numbers in E^1 with γ − level sets:

$[A]_\gamma = [[A]_\gamma^l, [A]_\gamma^u]$ and $[\mathrm{B}]_\gamma = [[B]_\gamma^l, [B]_\gamma^u]$, Over the rectangular region $x, t \in [a, b]$, let Φ be the kernel function, with a specified function denoted by F. Consider the fuzzy solution of the first order of fuzzy singular perturbed Volterra Iintegro-diffrintial equation be: $\mu(x) = [[\mu^l(x, \gamma)], [\mu^u(x, \gamma)]]$,

As well as it have the equivalent system

$$\varepsilon[\mu'(x)]_\gamma^l = [F(\mu, x)]_\gamma^l + \int_0^x [\Phi(x, t)\mu(t)]_\gamma^l\, dt, \text{ Where}$$

$$[\mu']_\gamma^l(\boldsymbol{a}) = [A]_\gamma^l, [\mu']_\gamma^u(\boldsymbol{a}) = [A]_\gamma^u$$

$$\varepsilon[\mu'(x)]_\gamma^u = [F(\mu, x)]_\gamma^u + \int_0^x [\Phi(x, t)\mu(t)]_\gamma^u dt, \text{ Where}$$

$$[\mu']_\gamma^l(\boldsymbol{b}) = [B]_\gamma^l, [\mu']_\gamma^u(\boldsymbol{b}) = [B]_\gamma^u$$

For $\gamma \in [0,1]$. Suppose $\Phi(x, t)$ be continuous in $[0,1]$ and mutate its sing in finite points for a certain fix *t*.

For this problem, the fuzzy trial solution can be written as:

$$[\mu_\mathrm{t}(x, \mathrm{p})]_{\boldsymbol{\gamma}} = \frac{\mathrm{b}[\mathrm{A}]_\gamma - \mathrm{a}[\mathrm{B}]_\gamma}{\mathrm{b}-\mathrm{a}} + \frac{[\mathrm{B}]_\gamma - [\mathrm{A}]_\gamma}{\mathrm{b}-\mathrm{a}}\, x + (x - \mathrm{a})(x - \mathrm{b})\,[O(x, \mathrm{p}, \varepsilon)]_{\boldsymbol{\gamma}}$$

where $O_{utp}(x, \mathrm{p}, \varepsilon)$ is output of the feed forward neural network with one input for x and parameter p.

The amount of error which must be minimized is given as:

$$\mathrm{E}(p, \varepsilon) = \sum_{i=1}^{g} (E_{i\boldsymbol{\gamma}}^{\boldsymbol{l}}(p) + E_{i\boldsymbol{\gamma}}^{\boldsymbol{u}}(p))$$

Where

$$\mathbb{E}_{i\gamma}^l(p, \varepsilon) = \left[\left[\frac{d\mu_\mathrm{t}(x_i, \mathrm{p})}{\mathrm{d}x}\right]_{\boldsymbol{\gamma}}^l - \frac{1}{\varepsilon}\left[\sum_{x_i \in D} F(x_i, \mu_t(x_i), \varepsilon) + \int_0^{x_i} \Phi(x_i, t, \varepsilon)\mu_t(t)dt\right]_\gamma^l\right]^2$$

$$\mathbb{E}_{i\gamma}^u(p, \varepsilon) = \left[\left[\frac{d\mu_\mathrm{t}(x_i, \mathrm{p})}{\mathrm{d}x}\right]_{\boldsymbol{\gamma}}^u - \frac{1}{\varepsilon}\left[\sum_{x_i \in D} F(x_i, \mu_t(x_i), \varepsilon) + \int_0^{x_i} \Phi(x_i, t, \varepsilon)\mu_t(t)dt\right]_\gamma^u\right]^2$$

where $\{x_i\}_{\mathrm{i}=1}^{\mathrm{g}}$ are discrete points [a,b], $E_{i\boldsymbol{\gamma}}^{\boldsymbol{l}}$ and $E_{i\boldsymbol{\gamma}}^{\boldsymbol{u}}$ are squared errors for the lower and upper limits of the γ-level sets, respectively.

With the aim of driving the minimized error function for the problem:

Then we could calculate:

$$[\mu_\mathrm{t}(x, p)]_\gamma^l = \frac{\mathrm{b}[\mathrm{A}]_\gamma^l - \mathrm{a}[\mathrm{B}]_\gamma^l}{\mathrm{b}-\mathrm{a}} + \frac{[\mathrm{B}]_\gamma^l - [\mathrm{A}]_\gamma^l}{\mathrm{b}-\mathrm{a}}\, x + (x - \mathrm{a})(x - \mathrm{b})[O(x, \mathrm{p}, \varepsilon)]_{\boldsymbol{\gamma}}^l$$

$$[\mu_\mathrm{t}(x, p)]_\gamma^u = \frac{\mathrm{b}[\mathrm{A}]_\gamma^u - \mathrm{a}[\mathrm{B}]_\gamma^u}{\mathrm{b}-\mathrm{a}} + \frac{[\mathrm{B}]_\gamma^u - [\mathrm{A}]_\gamma^u}{\mathrm{b}-\mathrm{a}}\, x + (x - \mathrm{a})(x - \mathrm{b})[O(x, \mathrm{p}, \varepsilon)]_{\boldsymbol{\gamma}}^u$$

Then we get:

$$\frac{[\mathrm{d}\mu_t(x,p)]_\gamma^l}{\mathrm{dx}} = \frac{[\mathrm{B}]_\gamma^l - [\mathrm{A}]_\gamma^l}{\mathrm{b-a}} + (x-\mathrm{a})(x-\mathrm{b})\frac{d[O(x,\mathrm{p},\varepsilon)]_\gamma^l}{\mathrm{dx}} + (2x-(a+b))O(x,\mathrm{p},\varepsilon))]_\gamma^l$$

$$\frac{[\mathrm{d}\mu_t(x,p)]_\gamma^u]}{\mathrm{dx}} = \frac{[\mathrm{B}]_\gamma^u - [\mathrm{A}]_\gamma^u}{\mathrm{b-a}} + (x-\mathrm{a})(x-\mathrm{b})\frac{d[O(x,\mathrm{p},\varepsilon)]_\gamma^u}{\mathrm{dx}} + (2x-(\mathrm{a}+\mathrm{b}))[O(x,\mathrm{p},\varepsilon)]_\gamma^u$$

Then $\mathbb{E}_{i\gamma}^l(p,\varepsilon)$ *and* $\mathbb{E}_{i\gamma}^u(p,\varepsilon)$ can be rewritten as:

$$E_{i\gamma}^l(p,\varepsilon) = \frac{[\mathrm{B}]_\gamma^l - [\mathrm{A}]_\gamma^l}{\mathrm{b}-\mathrm{a}} + (x-\mathrm{a})(x-\mathrm{b})\frac{d[O(x,\mathrm{p},\varepsilon)]_\gamma^l}{\mathrm{d}x} + (2x-(\mathrm{a}+\mathrm{b}))[O(x,\mathrm{p},\varepsilon)]_\gamma^l$$

$$-\frac{1}{\varepsilon}\left[\sum_{x_i\in D} F(x_i,\mu_t(x_i),\varepsilon) + \int_0^{x_i}\Phi(x_i,t,\varepsilon)\left(\frac{\mathrm{b}[\mathrm{A}]_\gamma^l - \mathrm{a}[\mathrm{B}]_\gamma^l}{\mathrm{b-a}} + \frac{[\mathrm{B}]_\gamma^l - [\mathrm{A}]_\gamma^l}{\mathrm{b-a}}\,\mathrm{t} + (\mathrm{t}-\mathrm{a})(\mathrm{t}-\mathrm{b})[O(x,\mathrm{p},\varepsilon)]_\gamma^l\right)dt\right]_\gamma^{l\,]^2}$$

$$E_{i\gamma}^u(p,\varepsilon) = \frac{[\mathrm{B}]_\gamma^u - [\mathrm{A}]_\gamma^u}{\mathrm{b-a}} + (x-\mathrm{a})(x-\mathrm{b})\frac{d[O(x,\mathrm{p},\varepsilon))]_\gamma^u}{\mathrm{d}x} + (2x-(\mathrm{a}+\mathrm{b}))[O(x,\mathrm{p},\varepsilon)]_\gamma^u - \frac{1}{\varepsilon}\left[\sum_{x_i\in D} F(x_i,\mu_t(x_i),\varepsilon) + \int_0^{x_i}\Phi(x_i,t,\varepsilon)\left(\frac{\mathrm{b}[\mathrm{A}]_\gamma^u - \mathrm{a}[\mathrm{B}]_\gamma^u}{\mathrm{b-a}} + \frac{[\mathrm{B}]_\gamma^u - [\mathrm{A}]_\gamma^u}{\mathrm{b-a}}\,\mathrm{t} + (\mathrm{t}-\mathrm{a})(\mathrm{t}-\mathrm{b})[O(x,\mathrm{p},\varepsilon)]_\gamma^u\right)dt\right]_\gamma^{u\,]^2}$$

4. Numerical Analysis

The section now is to describe a numerical result and give a solution to the model problem. For this model, we used a multi-layer feed-forward neural network with just one hidden layer with five hidden neurons and a unique linear output unit. The exact analytic solution $\mu(x_i)$ for every test problem is already known [14-15]. Finally, by computing the mean square error (MSE) we tested the accuracy of the solutions.

Example: Consider the FSPVIDE

$$\varepsilon\mu'(x) = [(\gamma^5+2\gamma)x^3,(6-3\gamma^3)x^3]\ \frac{\varepsilon}{24x}(36-5x^4) + \frac{1}{2}\int_0^x (x^2+t^2)\mu(t)dt,$$

with fuzzy B.C $\mu(0)=[0,0],\mu(1)=[\gamma^5+2\gamma,6-3\gamma^3]$ and $0\le t\le x\le 1$, $0<\varepsilon<<1,\gamma\in[0,1]$.

The exact solution is

$$[\mu(x)]_\gamma^l = (\gamma^5+2\gamma)x^3,\ [\mu(x)]_\gamma^u = (6-3\gamma^3)x^3$$

Then a fuzzy trial solution for this example is:

$$[\mu_t(x)]_\gamma = [\gamma^5+2\gamma,6-3\gamma^3]x + x(x-1)[O(x,\mathrm{p},\varepsilon)]_\gamma$$

where $O(x,\mathrm{p},\varepsilon)$ is the output of the feed forward.From the fuzzy trial solution we have

$$[\mu_t(x)]_\gamma^l = (\gamma^5+2\gamma)x + x(x-1)[O(x,\mathrm{p},\varepsilon)]_\gamma^l$$

$$[\mu_t(x)]_\gamma^u = (6-3\gamma^3)x + x(x-1)[O(x,\mathrm{p},\varepsilon)]_\gamma^u$$

The following function is the error function that have to be minimized which described as:

$\mathbb{E}(p) = \sum_{i=1}^{g}(\mathbb{E}_{i\gamma}^{l}(p) + \mathbb{E}_{i\gamma}^{u}(p))$ where

$$\mathbb{E}_{i\gamma}^{l}(p,\varepsilon) = \left[\left[\frac{d\mu_t(x_i,p)}{dx}\right]_{\gamma}^{l} - \frac{1}{\varepsilon}\left[\sum_{x_i\in D} F(\mu_t(x_i),x_i,\varepsilon) + \int_0^{x_i}\Phi(x_i,t,\varepsilon)\mu_t(t)dt\right]_{\gamma}^{l}\right]^2$$

$$\mathbb{E}_{i\gamma}^{u}(p,\varepsilon) = \left[\left[\frac{d\mu_t(x_i,p)}{dx}\right]_{\gamma}^{u} - \frac{1}{\varepsilon}\left[\sum_{x_i\in D} F(\mu_t(x_i),x_i,\varepsilon) + \int_0^{x_i}\Phi(x_i,t,\varepsilon)\mu_t(t)dt\right]_{\gamma}^{u}\right]^2$$

Then we get:

$$\left[\frac{d\mu_t(x_i,p)}{dx}\right]_{\gamma}^{l} =$$

$$(\gamma^5 + 2\gamma) + x(x-1)\frac{d[O(x,p,\varepsilon)]_{\gamma}^{l}}{dx} + (2x-1)[O(x,p,\varepsilon)]_{\gamma}^{l} \left[\frac{d\mu_t(x_i,p)}{dx}\right]_{\gamma}^{u} = (6 -$$

$$3\gamma^3) + x(x-1)\frac{d[O_{utp}(x,p,\varepsilon)]_{\gamma}^{u}}{dx} + (2x-1)[O(x,p,\varepsilon)]_{\gamma}^{u}$$

Then error function can be rewritten as:

$$\mathbb{E}_{i\gamma}^{l}(p,\varepsilon) = \left[\left[(\gamma^5 + 2\gamma) + x(x-1)\frac{d[O(x,p,\varepsilon)]_{\gamma}^{l}}{dx} + (2x-1)[O(x,p,\varepsilon)]_{\gamma}^{l}\right] -\right.$$

$$\frac{1}{\varepsilon}\sum_{x_i\in D}((\gamma^5 + 2\gamma){x_i}^3)\frac{\varepsilon}{24x_i}(36 - 5{x_i}^4) + \frac{1}{2}\int_0^{x_i}(x^2+t^2)[(\gamma^5 + 2\gamma)t +$$

$$\left.t(t-1)[O(t,p,\varepsilon)]_{\gamma}^{l}dt]\right]^2$$

$$\mathbb{E}_{i\gamma}^{u}(p,\varepsilon) = \left[(6 - 3\gamma^3) + x(x-1)\frac{d[O(x,p,\varepsilon)]_{\gamma}^{u}}{dx} + (2x-1)[O(x,p,\varepsilon)]_{\gamma}^{u} -\right.$$

$$\frac{1}{\varepsilon}\sum_{x_i\in D}((6 - 3\gamma^3){x_i}^3)\frac{\varepsilon}{24x_i}(36 - 5{x_i}^4) + \frac{1}{2}\int_0^{x_i}(x^2+t^2)[(6 - 3\gamma^3)t +$$

$$\left.t(t-1)[O(t,p,\varepsilon)]_{\gamma}^{u}dt]\right]^2$$

The error function that had to be minimized for this state is simply evaluated:

$$\mathbb{E} = \sum_{i=1}^{11}(\mathbb{E}_{i\gamma}^{l} + \mathbb{E}_{i\gamma}^{u})$$

Showing the approximate solution in order to compare it with the exact solution for a set of training points for the proposed network in the following tables and figures, and showing the accuracy and efficiency of the proposed method in Tables 1, 2, 3 and 4 .

Table 1

The analytic and FFNN solution of the example where $\varepsilon=10^{-5}$, $\gamma = 0.1$

input	Analytic solution		Solution of FFNN $\mu_t(x)$	
x	$[\mu_a(x)]_{\gamma}^{l}$	$[\mu_a(x)]_{\gamma}^{u}$	$[\mu_t(x)]_{\gamma}^{l}$	$[\mu_t(x)]_{\gamma}^{u}$
0.0	0.000000000000	0.000000000000	0.000000000000	0.000000000000
0.1	0.001568070000	0.004971000000	0.001568874000	0.004976338270
0.2	0.012544560000	0.039768000000	0.012544520986	0.039764399665

0.3	0.042337890000	0.134217000000	0.042337230050	0.134211775530
0.4	0.100356480000	0.318144000000	0.100356404528	0.318146449360
0.5	0.196008750000	0.621375000000	0.196008706649	0.621375442800
0.6	0.338703120000	1.073736000000	0.338703107111	1.073734399270
0.7	0.537848010000	1.705053000000	0.537848050063	1.705053775340
0.8	0.802851840000	2.545152000000	0.802851899630	2.545152885730
0.9	1.143123030000	3.623859000000	1.143123038856	3.623854992160
1	1.568070000000	4.971000000000	1.568070000000	4.971000000000

Table 2
Accuracy of solutions of the example where $\varepsilon = 10^{-5}$, $\gamma = 0.1$

The error $[\mathbb{E}(x)]_\gamma = \left\lvert [\mu_a(x)]_\gamma - \mu_t(x)]_\gamma \right\rvert$	
$[\mathbf{error}]_\gamma^{\mathbf{l}}$	$[\mathbf{error}]_\gamma^{\mathbf{u}}$
0	0
8.03999999999866E-07	5.33826999999862E-06
3.9014230001147E-08	3.60033500001328E-06
6.59949999995413E-07	5.22447000000925E-06
7.54725000134959E-08	2.44935999993423E-06
4.33509999842041E-08	4.42800000000076E-07
1.28889999784221E-08	1.60073000010641E-06
4.00633002328732E-08	7.75340000203073E-07
5.96299999378047E-08	8.85729999122731E-07
8.85640005776622E-09	4.00784000031962E-06
2.22044604925031E-16	0
MSE= 5.08801978064075E-13	MSE= 6.90543707633391E-11

Table 3
The analytic and FFNN solution of the example where $\varepsilon = 10^{-5}$, $x = 0.1$

input	Analytic solution		Solution of FFNN $\mu_t(x)$	
γ	$[\mu_a(x)]_\gamma^{\mathbf{l}}$	$[\mu_a(x)]_\gamma^{\mathbf{u}}$	$[\mu_t(x)]_\gamma^{\mathbf{l}}$	$[\mu_t(x)]_\gamma^{\mathbf{u}}$
0.0	0.000000000000	0.750000000000	0.000000000000	0.750000000000
0.1	0.025001250000	0.749625000000	0.025001244300	0.749623766400
0.2	0.050040000000	0.747000000000	0.050906654900	0.747653288530
0.3	0.075303750000	0.739875000000	0.075303743000	0.739873488520
0.4	0.101280000000	0.726000000000	0.101230000000	0.726723776400
0.5	0.128906250000	0.703125000000	0.128906303000	0.703123665930
0.6	0.159720000000	0.669000000000	0.159727500000	0.669400436000
0.7	0.196008750000	0.621375000000	0.196008320000	0.621372664554
0.8	0.240960000000	0.558000000000	0.240970080000	0.558500003800
0.9	0.298811250000	0.476625000000	0.298811290000	0.476623299763
1	0.375000000000	0.375000000000	0.375000000000	0.375000233500

Table 4
Accuracy of solutions of the example where $\varepsilon=10^{-5}$, $x = 0.1$

The error $[\mathbb{E}(x)]_\gamma = \left\lvert [\mu_a(x)]_\gamma - \mu_t(x)]_\gamma \right\rvert$		The error in HPM	
$[\mathbf{error}]_\gamma^{\mathbf{l}}$	$[\mathbf{error}]_\gamma^{\mathbf{u}}$	$[\mathbf{error}]_\gamma^{\mathbf{l}}$	$[\mathbf{error}]_\gamma^{\mathbf{u}}$
0	0	0	7.3571E-05
5.70000000324478E-09	1.23360000003903E-06	2.5125E-06	7.5333E-05
0.0008666549	0.000653288530000018	5.0288E-06	7.5070E-05
6.99999999631551E-09	1.51147999993828E-06	7.5676E-06	7.4354E-05
0.0000500000000000084	0.000723776399999987	1.0178E-05	7.2959E-05
5.29999999998587E-08	1.33407000002528E-06	1.2954E-05	7.0660E-05
7.49999999999362E-06	0.000400435999999948	1.6051E-05	6.7231E-05
4.29999999967956E-07	0.0000023354457000524	1.9698E-05	6.2444E-05
0.0000100799999999679	0.000500003800000037	2.4215E-05	5.6076E-05
3.99999999789458E-08	1.70023699996857E-06	3.0029E-05	4.7899E-05
0	2.33500000001996E-07	3.7686E-05	3.7686E-05
MSE= 7.53748761454963E-07	MSE= 1.36100357302887E-06	MSE= 3.55841E-05	MSE= 4.34963E-05

References

[1] S. R. Badeye, M. M. Woldaregay, and T. G. Dinka, "Solving singularly perturbed Fredholm integro-differential equation using exact finite difference method," *BMC Research Notes* (2023). [Online]. Available: https://doi.org/10.1186/s13104-023-06488-8.

[2] K. M. M. Al-Abrahemee, "Efficient Algorithm for Solving Fuzzy Singularly Perturbed Volterra Integro-Differential Equation," *Iraqi Journal of Science*, vol. 64, no. 11, pp. 5851-5865 (2023). [Online]. Available: https://orcid.org/0000-0002-4705-6349.

[3] K. M. M. Al-Abrahemee, "Modification of high performance training algorithm for solving singular perturbation partial differential equations with cubic convergence," *J. Interdiscip. Math.*, vol. 24, no. 7, pp. 2035-2047 (2021). [Online]. Available: https://doi.org/10.1080/09720502.2021.2001136.

[4] R. Kareem and M. Al-Abrahemee, "Modification artificial neural networks for solving singular perturbation problems," *J. Interdiscip. Math.*, vol. 25, no. 5, pp. 1535-1549 (2022). [Online]. Available: https://doi.org/10.1080/09720502.2022.2072063.

[5] R. F. Kadam and K. M. M. Al-Abrahemee, "On convergence of the Levenberg-Marquardt method under local error bound problems," *J.*

Interdiscip. Math., vol. 25, no. 5, pp. 1495-1508 (2022). [Online]. Available: https://doi.org/10.1080/09720502.2022.2079228.

[6] K. M. M. Al-Abrahemee, "High performance training algorithm with strong local convergence properties for solving some fractional integral equation," *AIP Conf. Proc.,* vol. 2611, p. 040038 (2023). [Online]. Available: https://doi.org/10.1063/5.0115034.

[7] S. Haykin, Neural networks: A comprehensive foundation (1993). [Online]. Available: https://dl.acm.org/doi/10.5555/541500.

[8] R. M. Hristev, The ANN Book, 1st ed. (1998). [Online]. Available: https://www.academia.edu/44105267/Hritsev_The_ANN_Book.

[9] L. Hooshangian, "Nonlinear Fuzzy Volterra Integro-differential Equation of N-th Order: Analytic Solution and Existence and Uniqueness of Solution," *Int. J. Industrial Mathematics*, vol. 11, no. 1 (2019). [Online]. Available: https://www.researchgate.net/publication/329894133.

[10] M. A. Saeed, H.K. Alkhayyat, and S.M. Kadham, "Enhance MRI images of lung cancer using hybrid transform," *J. Discrete Math. Sci. Cryptogr.,* vol. 26, no. 7, pp. 1897–1902 (Jan. 2023), [Online]. Available https://doi.org/10.47974/jdmsc-1682.

[11] K. M. M. Al-Abrahemee, "A novel hybridized neuro-fuzzy model for solving fuzzy singular perturbation problems with initial conditions," *J. Interdiscip. Math.,* vol. 26, no. 6, pp. 1287-1301 (2023). [Online]. Available: https://doi.org/10.47974/JIM-1627.

[12] D. Nauck and R. Kruse, "Neuro-Fuzzy Classification with NEFCLASS," *Operations Research Proceedings*, pp. 294-299 (1996). [Online]. Available: https://doi.org/10.1007/978-3-642-80117-4_5

[13] M. Mosleh and M. Otadi, "Simulation and evaluation of fuzzy differential equations by fuzzy neural network," *Appl. Soft Comput.,* vol. 12, no. 9, pp. 2817-2827 (2012). [Online]. Available: https://doi.org/10.1016/j.asoc.2012.03.041.

[14] S. M. Kadham, "Acute interstitial pneumonia image enhancement using fuzzy partial transforms," *Applied Geomatics,* vol. 16, no. 1, pp. 35–39 (May 2023), doi: https://doi.org/10.1007/s12518-023-00509-8.

[15] S. M. Kadham and A. N. Alkiffai, "Generalization of Fuzzy SHA-Transform with Medical Application," *Mathematical Modelling of Engineering Problems,* vol. 9, no. 5, pp. 1378–1384 (Dec. 2022), doi: https://doi.org/10.18280/mmep.090528.

Received June, 2024