Al-Qadisiyah Journal of Pure Science

ISSN(Printed): 1997-2490 ISSN(Online): 2411-3514

DOI: /10.29350/jops.

http://qu.edu.iq/journalsc/i

Contrive a feed-forward fuzzy neural-network for solving Fuzzy Singularly Perturbed Volterra integral Equations

Ghassan salih salman
Khalid Mindeel Mohammed

Department of Mathematics, College of Education,

AL-Qadisiyah University, AL-Qadisiyah, Iraq

.

Qayrawan2007@gmail.com

Abstract

The research of singular Volterra integral equations has a huge curiosity in order to obtain a more accurate and fastest way to find its solutions. We grant in our paper a design be affiliated to fast feed forward neural network so that we could offer brand-***new*** method for solving one dimensions fuzzy singular perturbed integral equations. Applying a single multilayer which has one hidden layer with five units and unique linear output unit. ***it*** must ***be*** noted that is to every unit the sigmoid activation is hyperbolic tangent function and Levenberg – Marquardt training algorithm. In order to confirm the efficacy as well as the accurateness to the exhibited approach we viewed a matching tables clarify the numerical experiments results with the results of our new method.

Key words:, Fuzzy singular Volterra integral equations ; fuzzy sets , feed forward neural network ; hyperbolic tangent function.

1. Introduction

The improving and assessing numerical methods in order to solve one-dimensional integro-differential equation had many years of hard work. In our paper the one-dimensional singular perturbed fuzzy Volterra integral equations is contemplated. where:

$$\varepsilon\mu(x) = F(\mu, x, \varepsilon) + \int_0^x \Phi(x, t)\mu(t)dt \ , x \in [a, b]$$

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

Here, ε gives to the problem its rise of singularly perturbed nature since it is a tiny parameter, the kernel Φ and the data function F are two known smooth functions. Beneath suitable status for F and Φ, one dimension problems of this kind had been studied thoroughly last decade in order to find it's solutions.

It is known that is when we try to solve problems that is relevant to the real world there will be an arising of imprecisions and misgivings. Recently, another ways of calculating designated manufactured insights that through diverse strategies is occurred, in a way that it is easy to apply as well as giving a solid solution. Artificial Neural Networks (ANN) is the strategy that we shall review currently.

By the inspiration of the human brain functioning joined along with intelligence create a mixture of several components in order to processing artificial neurons corresponds to each other so it could be functioned consistently.

So we could solve various problems that are related to our classification aspects by the constructive of any identified given ANN.

Three kinds of computational neurons exists depends on its network **position which named:**
(input**)**, (output) and (hidden).[3-6]

We constituted this paperwork by the following order: the ensuing part defines the artificial neural network, in the part that flowed we illustrate neural network construction, in section 4 we depict the method, after that we illustrate it in section 5, at section 6 in order to find efficiency and accuracy of the method we have demonstrate a report of our numerical. in the end of our paper we put the conclusions of our paperwork.

2 . Artificial Neural Network

mainly, there exist a lot of extraordinary ways to describe the reality of artificial neural network, one of its definitions is the one that has the definition that pursuit in a detailed clarification to understand the neural-network or neural computing.

bearing that in mind, Haykin [7] brought a suggest a descript to show the great correspond organizing in aligned of straight forward ingredients of the artificial neural systems. Where it had a hierarchic organization, Which aim as in the organic apprehensive framework to associate with the **real world objects**.

The training or might called sometimes the learning network, in order to understand it, consider that the artificial is identical to the neuron which are recognizable as the computational neurons or nodes. Where it could be classed gradually via its layers as the same of them, selfsame with that in the nervous systems of biological human. So that

this modification is what we called learning network or training.

The symbolization of the following is ANN by [8]

a-Architecture: is the blueprint of neurons associations.

b- algorithm training: is the procedure that aim to determine the value of the weight that lay on links.
c- Activations functions.

3. **Essential** notions of fuzzy sets:

At this segment, we will demonstrate **different Essential** notions which they are relevant to the fuzzy theory .

Definition 1 [9]: If Υ is a set of substances, then if $\mathcal{A}$ is a fuzzy set in Υ in which it is a set of ordered pairs:

$$\beta = \left\{ \left(\Upsilon, \xi_\beta(y) \right) : y \in \Upsilon, 0 \le \xi_\beta(y) \le 1 \right\}$$

$\xi_\beta(y)$ is referred to be the grade of membership or membership function of y in β (also identified to be the compatibility degree or truth degree) that frameΥ to the space of membership $\xi_\beta(y)$ (β is not fuzzy when $\xi_\beta(y)$ consist just the two points 0 and 1) and $\xi_\beta(y)$ is identical to a non fuzzy set's characteristic function).

Definition 2 [10] : the crisp set of all $y \in \Upsilon$ which is known by the support of a fuzzy set β, and denoted by S(β). Along with $\xi_\beta(y) > 0$

Definition 3 [11] : From the fuzzy set β at least to the degree λ , the (crisp) set of elements which is called the λ-level set: $\beta_\lambda = \{ y \in \Upsilon : \xi_\beta(y) \ge \lambda \}$

$\grave{\beta_\lambda} = \{ y \in \Upsilon : \xi_\beta(y) > \lambda \}$ is called "strong λ-level set" or "strong λ-cut."

Definition 4 [11] : β is convex , where β is a fuzzy set if:

$$\beta(\delta y_1 + (1-\varepsilon) y_2) \ge min\{\xi_\beta(y_1), \xi_\beta(y_2)\}, y_1, y_2 \in \Upsilon, \delta \in [0,1]$$

On the other hand, if all the λ -level sets of a fuzzy set are convex then this fuzzy set is convex

Definition 5 [12]: If r is a fuzzy number, then it well be determined completely by an ordered pair of functions ($r^l(\lambda)$,$r^u(\lambda)$), $0 \le \lambda \le 1$, that appease the next points:

1) $r^l(\lambda)$ is a bounded left continuous non-decreasing function on [0,1].
2) $r^u(\lambda)$ is a bounded left continuous function which not increasing. [0,1].
3) $r^l(\lambda) \le r^u(\lambda)$, $0 \le \lambda \le 1$. The crispy number q is only signified by: $r^l(\lambda) = r^u(\lambda) = q$, $0 \le \lambda \le 1$. Where E^1 denotes the set of all the fuzzy numbers.

Remark (**1**),[12]: For subjective $r = (r^l , \lambda^u)$, $v = (v^l, v^u)$ and $C \in$ R, then addition as well as multiplication by C is defined by :

1) $(\delta + v)^l(\lambda) = \delta^l(\lambda) + v^l(\lambda)$
2) $(\delta + v)^u(\lambda) = \delta^u(\lambda) + v^u(\lambda)$
3) $(C\delta)^l(\lambda) = C\delta^l(\lambda)$, $(C\delta)^u(\lambda) = C\delta^u(\lambda)$, if $C \ge 0$
4) $(C\delta)^l(\lambda) = C\delta^u(\lambda)$, $(C\delta)^u(\lambda) = C\delta^l(\lambda)$, if $C < 0$.For all $\lambda \in [0,1]$.

Remark (**2**),[13]:The distance function in the middle of random two fuzzy numbers $r = (r^l , r^u)$, and $v = (v^l, v^u)$ is given as:

$$D(r, v) = \left[\int_0^1 \left(r^l(\lambda) - v^l(\lambda) \right)^2 d\tau + \int_0^1 r^u(\lambda) - v^u(\lambda)\, d\lambda \right]^{\frac{1}{2}}$$

Definition 6,[12] : Let ψ: $R \longrightarrow E^1$ be a function called (fuzzy function). In addition , a (fuzzy function) is every function defined from $\beta_1 \subseteq E^1$ into $\beta_2 \subseteq E^1$.

Definition 7,[12]: The function ψ: R ⟶ E^1 which is clearly fuzzy, is said to be continuous when: as regards a random $y_1 \in$ R as well as $\in > 0$ then should be in existence $\alpha > 0$ in which: $|y - y_1| < \alpha \Rightarrow D(\psi(u), \psi(u_1)) < \in$, where the distance function D is betwixt of a pair of fuzzy numbers.

Definition 8,[12]: Let I denotes the real interval. The τ-level set of the fuzzy function Ψ: I → E_1 can be denoted by: $[\Psi(u)]^\tau = [\Psi_1^\tau(u), \Psi_2^\tau(u)], x \in I, \tau \in [0,1]$

From the fuzzy function Ψ(x) let the Seikkala derivative Ψ′(u) defined by: $[\Psi'(u)]^\tau = [(\Psi_1^\tau)'(u), (\Psi_2^\tau)'(u)]$ $x \in$ I, $\tau \in [0,1]$

4. A Fuzzy Neural Network Structure

Put in mind that the triplet layers which are feed forward neural network as well as fuzzy these layers have n input units, m hidden units and s output units. The input vector is any real number. Also the fuzzy numbers are the target vectors, the biases as well as connection weights. The amount of neurons that each layer has, that is (n × m × s) , refers to the feed forward neural network dimension, where n is the amount of neurons in an input layer, m is the amount of hidden layers and s is the amount of output layer.

The model construction explains the way that feed forward neural network alters the n number of inputs $(x_1, x_2, \ldots x_i, \ldots x_n)$ into the an s amount of fuzzy outputs:

$([O_{t1}]_\Upsilon, [O_{t2}]_\Upsilon, [O_{t3}]_\Upsilon, \ldots, [O_{tk}]_\Upsilon, \ldots, [O_{ts}]_\Upsilon)$ including every one of the m hidden fuzzy neurons $([h_{d1}]_\Upsilon, [h_{d2}]_\Upsilon, [h_{d3}]_\Upsilon, \ldots, [h_{dj}]_\Upsilon, \ldots, [h_{dm}]_\Upsilon)$ being $[q_j]_\Upsilon$ and $[z_k]_\Upsilon$ fuzzy biases the fuzzy neurons $[h_{dj}]_\Upsilon$, $[O_t]_\Upsilon$ respectively, $[w_{ji}]_\Upsilon$ be all the weights where they are fuzzy and connects fuzzy neuron $[h_{dj}]_\Upsilon$ to the crisp neuron x_i, side to side with the fuzzy weight $[v_{kj}]_\Upsilon$ that connects fuzzy neuron $[O_t]_\Upsilon$ to the fuzzy neuron $[h_{dj}]_\Upsilon$.

<u>Input unit</u>

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

Hidden unit:

$[h_{dj}]_\Upsilon = [\,[h_{dj}]_\Upsilon^l, [h_{dj}]_\Upsilon^u] = \Upsilon([N_{tj}]_\Upsilon) = [\,\Upsilon[N_{tj}]_\Upsilon^l, \Upsilon[[N_{tj}]_\Upsilon^u]$

$[h_{dj}]_\Upsilon = \Upsilon([N_{tj}]_\Upsilon)$, j = 1,2,3,…, m

Where

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

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

<u>Output unit :</u>

$[O_t]_\Upsilon = [[O_t]_\Upsilon^l, [O_t]_\Upsilon^u] = \Upsilon([N_{tjk}]_\Upsilon)$, k = 1,2,3,…, s

Where

$[N_{tj}]_\Upsilon^l = (\sum_{j\epsilon a}[v_{kj}]_\Upsilon^l [h_{dj}]_\Upsilon^l + \sum_{j\epsilon b}[v_{kj}]_\Upsilon^l [h_{dj}]_\Upsilon^u) + [z_k]_\Upsilon^l$

$[N_{tj}]_\Upsilon^u = (\sum_{j\epsilon c}[v_{kj}]_\Upsilon^u [h_{dj}]_\Upsilon^u + \sum_{j\epsilon d}[v_{kj}]_\Upsilon^u [h_{dj}]_\Upsilon^l) + [z_k]_\Upsilon^u$

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

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

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

5. The method illustration

Consider the one dimension Fuzzy Singularly Perturbed Volterra integral Equations for ODEs

$$\varepsilon\mu(x) = F(x,\varepsilon) + \int_0^x \Phi(x,q)\mu(q)dq \ , x \in [a,b]$$

where ε is perturbation paremter ($0 < \varepsilon << 1$)

With the fuzzy (BC) :

In the case of Dirichlet (BC) : $\mu(a) = [A]_\Upsilon$, $\mu(b) = [B]_\Upsilon$

Where $[A]_\Upsilon$ *and* $[B]_\Upsilon$ are fuzzy number in E^1 with $\Upsilon -$ level sets

$[A]_\Upsilon = [[A]^l_\Upsilon, [A]^u_\Upsilon]$ and $[B]_\Upsilon = [[B]^l_\Upsilon, [B]^u_\Upsilon]$

, Φ refers to the kernel function over the oblongata dominion $x, q \in [a,b]$, μ is a fuzzy function and $F(x)$ is any specific given fuzzy function, such an equation **might** possess only one fuzzy solution .

Say that $\mu(x) = [[\mu^l(x,\Upsilon)], [\mu^u(x,\Upsilon)]]$ is the first order fuzzy solution of the Fuzzy Singularly Perturbed Volterra integral Equations , have the equivalent system:

$$\varepsilon[\mu(x)]^l_\Upsilon = [F(x,\varepsilon)]^l_\Upsilon + \int_0^x [\Phi(x,q)\mu(q)]^l_\Upsilon \, dq \text{ , Where}$$

$$[\mu]^l_\Upsilon(a) = [A]^l_\Upsilon, [\mu]^u_\Upsilon(a) = [A]^u_\Upsilon$$

$$\varepsilon[\mu(x)]^u_\Upsilon = [F(x,\varepsilon)]^u_\Upsilon + \int_0^x [\Phi(x,q)\mu(q)]^u_\Upsilon dq \text{ , Where}$$

$$[\mu]^l_\Upsilon(b) = [B]^l_\Upsilon, [\mu]^u_\Upsilon(b) = [B]^u_\Upsilon$$

For $\Upsilon \in [0,1]$. Suppose $\Phi(x,q)$ on [0,1] is continuous, and changes its sing in finite points for fix q.

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

$$[\mu_t(x,p)]_\Upsilon = \frac{b[A]_\Upsilon - a[B]_\Upsilon}{b-a} + \frac{[B]_\Upsilon - [A]_\Upsilon}{b-a} x + (x - a)(x - b)\,[O_t(x,p,\varepsilon)]_\Upsilon$$

where $O_t(x, p, \varepsilon)$ is output of the feed forward FFNN with one input for x and parameter p .

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

$$E(p,\varepsilon) = \sum_{i=1}^{g} (E^l_{i\Upsilon}(p) + E^u_{i\Upsilon}(p))$$

Where

$$\mathbb{E}^l_{i\Upsilon}(p,\varepsilon) = \left[[\mu_t(x_i,p)]^l_\Upsilon - \frac{1}{\varepsilon}\left[\sum_{x_i \in D} F(\mu_t(x_i), x_i, \varepsilon) + \int_0^{x_i} \Phi(x_i,q,\varepsilon)\mu_t(q)dq\right]^l_\Upsilon\right]^2$$

$$\mathbb{E}^u_{i\Upsilon}(p,\varepsilon) = \left[[\mu_t(x_i,p)]^u_\Upsilon - \frac{1}{\varepsilon}\left[\sum_{x_i \in D} F(\mu_t(x_i), x_i, \varepsilon) + \int_0^{x_i} \Phi(x_i,q,\varepsilon)\mu_t(q)dq\right]^u_\Upsilon\right]^2$$

where $\{x_i\}_{i=1}^{g}$ are discrete points [a,b], $E^l_{i\Upsilon}$ and $E^u_{i\Upsilon}$ are squared errors for the lower and upper limits of the $\Upsilon -$ level sets, respectively.

To drive the minimized error function for we get:

$$[\mu_t(x,p)]^l_Y = \frac{b[A]^l_Y - a[B]^l_Y}{b-a} + \frac{[B]^l_Y - [A]^l_Y}{b-a}\, x + (x-a)(x-b)[O_t(x,p,\varepsilon)]^l_Y$$

$$[\mu_t(x,p)]^u_Y = \frac{b[A]^u_Y - a[B]^u_Y}{b-a} + \frac{[B]^u_Y - [A]^u_Y}{b-a}\, x + (x-a)(x-b)[O_t(x,p,\varepsilon)]^u_Y$$

Then:

$$E^l_{iY}(p,\varepsilon) = \frac{b[A]^l_Y - a[B]^l_Y}{b-a} + \frac{[B]^l_Y - [A]^l_Y}{b-a}\, x + (x-a)(x-b)[O_t(x,p,\varepsilon)]^l_Y - \left[\frac{1}{\varepsilon}\left[\sum_{x_i\in D} F(\mu_t(x_i),x_i,\varepsilon) + \int_0^{x_i}\Phi(x_i,q,\varepsilon)\left(\frac{b[A]^l_Y - a[B]^l_Y}{b-a} + \frac{[B]^l_Y - [A]^l_Y}{b-a} q + (q-a)(q-b)[O_t(x,p,\varepsilon)]^l_Y\right)dq\right]^l_Y\right]^2$$

$$E^u_{iY}(p,\varepsilon) = \frac{b[A]^u_Y - a[B]^u_Y}{b-a} + \frac{[A]^u_Y - [B]^u_Y}{b-a}\, x + (x-a)(x-b)[O_t(x,p,\varepsilon)]^u_Y - \left[\frac{1}{\varepsilon}\left[\sum_{x_i\in D}(\mu_t(x_i),x_i,\varepsilon) + \int_0^{x_i}\Phi(x_i,q,\varepsilon)\left(\frac{b[A]^u_Y - a[B]^u_Y}{b-a} + \frac{[A]^u_Y - [B]^u_Y}{b-a}\, q + (q-a)(q-b)[O_t(x,p,\varepsilon)]^u_Y\right)dq\right]^u_Y\right]^2$$

6. Numerical results

Now we will give in this segment, a graphics that represent the operating as well as qualities of our present design from a programs that we built in MATLAB 7.11. so the segment below is to depict the numerical result and give model problem its solution. in the present model we took advantage a multi-layer feed forward neural network that have only one hidden layer as well as 5 hidden neurons and just one unique linear output unit. Where for every test problem, it's analytic exact solution $\mu(x_i)$ is previously noted. Eventually by calculating the mean square error (MSE) we approved the gained solutions accuracy.

Example.1 :

Consider the fuzzy singular Volterra Abel's integral equation of the second kind

$$\varepsilon\mu(x,\Upsilon) = F(x,\Upsilon) - \int_0^x \frac{\mu(q,\Upsilon)}{\sqrt{x-q}}\,dq \quad ,$$

Where $F(x,\Upsilon) = \left(\frac{\Upsilon}{15}\left(15x^2 + 16x^{\frac{5}{2}}\right), \frac{1}{15}\,(2-\Upsilon)x^2 + (32-16\Upsilon)x^{\frac{5}{2}}\right)$

with BCs: $[\mu(0)]_\Upsilon = [0,0]_\Upsilon$, $[\mu(1)]_\Upsilon = [\Upsilon, 2-\Upsilon]_\Upsilon$

The exact solution is: $[\mu(x)]^l_\Upsilon = (\Upsilon)x^2$, $[\mu(x)]^u_\Upsilon = (2-\Upsilon)x^2$

Then the fuzzy trial solution for this example is :

$[\mu_t(x)]_Y = [\Upsilon, 2-\Upsilon]x + x(x-1)[O_t(x,p,\varepsilon)]_Y$

where $O_t(x,p,\varepsilon)$ the output of the feed forword .From fuzzy trial solution we have

$[\mu_t(x)]^l_Y = Y\,x + x(x-1)[O_t(x,p,\varepsilon))]^l_Y$

$[\mu_t(x)]^u_Y = (2-\Upsilon)x + x(x-1)[O_t(x,p,\varepsilon))]^u_Y$

Now , the function of error that should minimized is:

$\mathbb{E}(p) = \sum_{i=1}^{g}(\mathbb{E}^l_{iY}(p) + \mathbb{E}^u_{iY}(p))$ where

$$\mathbb{E}^l_{iY}(p,\varepsilon) = \left[[\mu_t(x_i,p)]^l_Y - \frac{1}{\varepsilon}\left[\sum_{x_i\in D} F(x_i,\varepsilon) + \int_0^{x_i}\Phi(x_i,q,\varepsilon)\mu(q)dq\right]^l_Y\right]^2$$

$$\mathbb{E}^u_{iY}(p,\varepsilon) = \left[[\mu_t(x_i,p)]^u_Y - \frac{1}{\varepsilon}\left[\sum_{x_i\in D} F(x_i,\varepsilon) + \int_0^{x_i}\Phi(x_i,q,\varepsilon)\mu_t(q)dq\right]^u_Y\right]^2$$

$$\mathbb{E}_{i\Upsilon}^{l}(p,\varepsilon)=\left[\Upsilon\, x+x(x-1)[O_t(x,p,\varepsilon))]_{\Upsilon}^{l}-\frac{1}{\varepsilon}\left[\sum_{x_i\in D}(\Upsilon x^2)\left(\frac{\Upsilon}{15}\left(15x^2+16x^{\frac{5}{2}}\right)\right)+\int_0^{x_i}\frac{\Upsilon q+q(q-1)[O_t(q,p,\varepsilon)]_{\Upsilon}^{l}}{\sqrt{x-q}}\,dq\right]_{\Upsilon}^{l}\right]^2$$

$$\mathbb{E}_{i\Upsilon}^{u}(p,\varepsilon)=\left[(2-\Upsilon)x+x(x-1)\,[O_t(x,p,\varepsilon))]_{\Upsilon}^{u}-\frac{1}{\varepsilon}\left[\sum_{x_i\in D}(2-\Upsilon)x^2\left(\frac{1}{15}(2-\Upsilon)x^2+(32-16\Upsilon)x^{\frac{5}{2}}\right)+\int_0^{x_i}\frac{(2-\Upsilon)q+q(q-1)[O_t(q,p,\varepsilon))]_{\Upsilon}^{u}}{\sqrt{x-q}}\,dq\right]_{\Upsilon}^{u}\right]^2$$

Now $\mathbb{E}=\sum_{i=1}^{11}(\mathbb{E}_{i\Upsilon}^{l}+\mathbb{E}_{i\Upsilon}^{u})$ is the error function that should minimized for a state that it became plain to estimate.

The figures and tables below shows the approximated solution as well as comparing it with an analytic solution for a group about points of training to the offered network, as well as tables of error showing the precision and acceleration of convergence of our offered method.

Table(1) the analytic and FFNN solution of the example (1) where ε=10^{-5} , $\Upsilon=0.7$

input	"Analytic solution"		"Solution of FFNN $\mu_t(x)$"	
x	$[\mu_a(x)]_{\Upsilon}^{l}$	$[\mu_a(x)]_{\Upsilon}^{u}$	$[\mu_t(x)]_{\Upsilon}^{l}$	$[\mu_t(x)]_{\Upsilon}^{u}$
0.0	0.000000000000	0.980000000000	0.000000000000	0.980000003280
0.1	0.049000000000	0.931000000000	0.049000000535	0.931000008860
0.2	0.098000000000	0.882000000000	0.098000000911	0.882000000000
0.3	0.147000000000	0.833000000000	0.147000000000	0.833000004380
0.4	0.196000000000	0.784000000000	0.196000003520	0.784000000810
0.5	0.245000000000	0.735000000000	0.245000003360	0.735000000004
0.6	0.294000000000	0.686000000000	0.294000000002	0.686000000004
0.7	0.343000000000	0.637000000000	0.343000000028	0.637000000006
0.8	0.392000000000	0.588000000000	0.392000043775	0.588000000300
0.9	0.441000000000	0.539000000000	0.441000000034	0.539000000400
1	0.490000000000	0.490000000000	0.490000000000	0.490000000016

Table(2) accuracy of solutions of the example(1) where ε=10^{-5}, $\Upsilon=0.7$

The error $[\mathbb{E}(x)]_{\Upsilon}=\lvert[\mu_a(x)]_{\Upsilon}-\mu_t(x)]_{\Upsilon}\rvert$	
$[\text{error}]_{\Upsilon}^{l}$	$[\text{error}]_{\Upsilon}^{u}$
0	3.28000016036611E-09
5.35000002632735E-10	8.86000017796817E-09
9.10800004860768E-10	1.11022302462516E-16
2.77555756156289E-17	4.38000014035822E-09
3.52000001369035E-09	8.10000067019701E-10
3.36000002820747E-09	3.80007136868699E-12
2.30004904011594E-12	3.90021348550817E-12
2.76000888810302E-11	5.80013814754921E-12
4.37750000137349E-08	3.00000135844414E-10
3.4420077899E-11	3.99800081929413E-10
0	1.60000901416879E-11
MSE= 1.94078815538898E-15	MSE= 2.72269883812399E-17

Table(3) the analytic and FFNN solution of the example(1) where ε=10^{-5}, $x = 0.5$

input	"Analytic solution"		"Solution of FFNN $\mu_t(x)$"	
Υ	$[\mu_a(x)]^l_\Upsilon$	$[\mu_a(x)]^u_\Upsilon$	$[\mu_t(x)]^l_\Upsilon$	$[\mu_t(x)]^u_\Upsilon$
0.0	0.000000000000	0.500000000000	0.000000000000	0.500000000000
0.2	0.050000000000	0.450000000000	0.050000000010	0.450000000054
0.4	0.100000000000	0.400000000000	0.100000000043	0.400000000003
0.6	0.150000000000	0.350000000000	0.150000000000	0.350000000003
0.8	0.200000000000	0.300000000000	0.200000000000	0.300000000642
1	0.250000000000	0.250000000000	0.250000000001	0.250000000004

Table(4) accuracy of solutions of the example(1) where ε=10^{-5}, $x = 0.5$

The error $[\mathbb{E}(x)]_\Upsilon = \lvert[\mu_a(x)]_\Upsilon - \mu_t(x)]_\Upsilon\rvert$	
$[\text{error}]^l_\Upsilon$	$[\text{error}]^u_\Upsilon$
0	0
9.79999414951749E-12	5.43000089336942E-11
4.29999924556057E-11	3.19999582387709E-12
0	3.0000446571421E-12
3.99985600196828E-13	6.42000008710397E-10
1.1999845561661E-12	4.30000479667569E-12
MSE= 1.94663918792813E-21	MSE= 4.15150232436818E-19

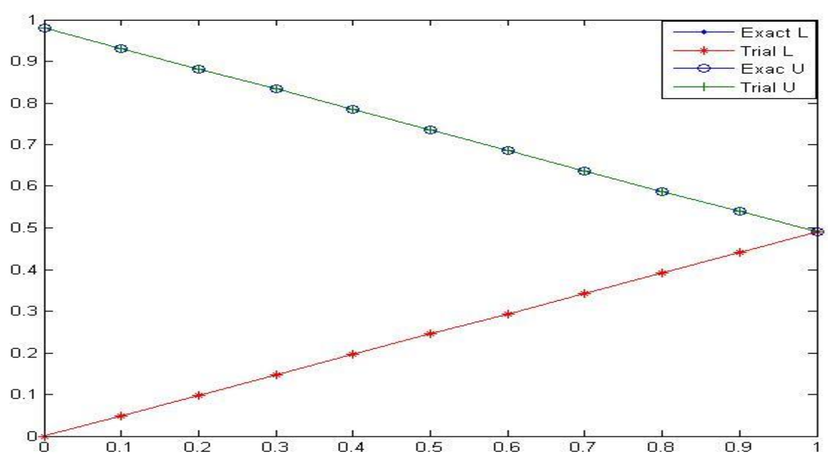

Figure (1) analytic and neural solution of example (1) with ε =10^{-5}

Example.2 : Consider the following fuzzy singular Volterra integral equation

$$\varepsilon\mu(x,\gamma) = \left(\left(x+\frac{4}{3}x^{\frac{3}{2}}\right)(4+\gamma), \left(x+\frac{4}{3}x^{\frac{5}{2}}\right)(6-\gamma)\right) - \int_0^x \left(\frac{[\mu(x)]^l_\Upsilon, [\mu(x)]^u_\Upsilon)}{\sqrt{x-q}}\right) dq$$

with I.C : $[\mu(0)]_\gamma = [0,0]_\gamma$

The exact solution is

$$[\mu(x)]^l_\Upsilon = (4+\gamma)x \ , \qquad [\mu(x)]^u_\Upsilon = (6-\gamma)x$$

Then the fuzzy trial solution for this example is :

$[\mu_t(x)]_\Upsilon = \ x\,[O_t(x, \text{p}, \varepsilon)]_\Upsilon$

where $O_t(x, \mathrm{p}, \varepsilon)$ the feed forword output. From fuzzy trial solution we get:
E(p) = $\sum_{i=1}^{g}([E(p)]_{i\Upsilon}^{l}, [E(p)]_{i\Upsilon}^{u})$

$$\mathbb{E}_{i\Upsilon}^{l}(p,\varepsilon) = \left[[\mu_t(x_i,\mathrm{p})]_{\Upsilon}^{l} - \frac{1}{\varepsilon}\left[\sum_{x_i\in D} F(\mu_t(x_i), x_i, \varepsilon) + \int_0^{x_i} \Phi(x_i, q, \varepsilon)\mu_t(q)dq\right]_{\Upsilon}^{l}\right]^2$$

$$\mathbb{E}_{i\Upsilon}^{u}(p,\varepsilon) = \left[[\mu_t(x_i,\mathrm{p})]_{\Upsilon}^{u} - \frac{1}{\varepsilon}\left[\sum_{x_i\in D} F(\mu_t(x_i), x_i, \varepsilon) + \int_0^{x_i} \Phi(x_i, q, \varepsilon)\mu_t(q)dq\right]_{\Upsilon}^{u}\right]^2$$

Then we get:

$$[E(p)]_{i\Upsilon}^{l} = \left[x\,[O_t(x,\mathrm{p},\varepsilon) - \tfrac{1}{\varepsilon}\left[\textstyle\sum_{x_i,y_i,\in D}\left(x + \tfrac{4}{3}x^{\frac{3}{2}}\right)(4+\gamma)x + \int_0^{x_i}(\tfrac{1}{\sqrt{x-q}})(\,\mathrm{q}\,[O_t(\mathrm{q},\mathrm{p},\varepsilon)]_{\Upsilon})dq\right]_{\Upsilon}^{l}\right]^2$$

$$[E(p)]_{i\Upsilon}^{u} = \left[x\,[O_t(x,\mathrm{p},\varepsilon) - \tfrac{1}{\varepsilon}\left[\textstyle\sum_{x_i,y_i,\in D}\left(x + \tfrac{4}{3}x^{\frac{5}{2}}\right)(6-\gamma)x + \int_0^{x_i}(\tfrac{1}{\sqrt{x-q}})(\,\mathrm{q}\,[O_t(\mathrm{q},\mathrm{p},\varepsilon)]_{\Upsilon})dq\right]_{\Upsilon}^{u}\right]^2$$

Then the function of error that should minimized for now is simply evaluated by:
$\mathbb{E} = \sum_{i=1}^{11}(\mathbb{E}_{i\Upsilon}^{l} + \mathbb{E}_{i\Upsilon}^{u})$

The figures and tables below shows the approximated solution as well as comparing it with an analytic solution for a group about points of training to the offered network, as well as tables of error showing the precision and acceleration of convergence of our offered method.

Table(5) the analytic and FFNN solution of the example (2) where **ε=10^{-9} ,** $\gamma = 0.7$

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	2.800000000000	4.200000000000	2.800000000000	4.200000000000
0.1	2.870000000000	4.130000000000	2.870000000443	4.130000000365
0.2	2.940000000000	4.060000000000	2.940000000334	4.060000000078
0.3	3.010000000000	3.990000000000	3.010000000001	3.990000044298
0.4	3.080000000000	3.920000000000	3.080000000002	3.920007726540
0.5	3.150000000000	3.850000000000	3.150000000000	3.850000000006
0.6	3.220000000000	3.780000000000	3.220000000870	3.780000000000
0.7	3.290000000000	3.710000000000	3.290000000653	3.710000000663
0.8	3.360000000000	3.640000000000	3.360000000019	3.640000003966
0.9	3.430000000000	3.570000000000	3.430000000554	3.570000006692
1	3.500000000000	3.500000000000	3.500000000000	3.500000000000

Table(6) accuracy of solutions of the example (2) where **ε=10^{-9}**, $\gamma = 0.7$

The error $[\mathbb{E}(x)]_\gamma = \left|[\mu_a(x)]_\gamma - \mu_t(x)]_\gamma\right|$	
$[\mathbf{error}]_\gamma^{\mathbf{l}}$	$[\mathbf{error}]_\gamma^{\mathbf{u}}$
0	0
4.4300030310751E-10	3.65000474289445E-10
3.33999938817442E-10	7.76507746991228E-11
1.00008890058234E-12	4.42980003789728E-08
1.99973371195483E-12	0.0000077265400006965
0	5.50048895320288E-12
8.70000071984123E-10	0
6.52999876393778E-10	6.63000321310392E-10
1.88000726097926E-11	3.96600041696615E-09
5.53999957020324E-10	6.6923004915509E-09
0	0
MSE= 1.61998015976986E-18	**MSE**= 5.97014436689866E-11

Table(7) the analytic and FFNN solution of the example (2) where **ε=10^{-9}** , $x = 0.5$

input	**"Analytic solution"**		**"Solution of FFNN $\mu_t(x)$"**	
ε	$[\mu_a(x)]_\gamma^{\mathbf{l}}$	$[\mu_a(x)]_\gamma^{\mathbf{u}}$	$[\mu_t(x)]_\gamma^{\mathbf{l}}$	$[\mu_t(x)]_\gamma^{\mathbf{u}}$
0	2.000000000000	3.000000000000	2.000000000000	3.000000000000
0.2	2.100000000000	2.900000000000	2.100000005400	2.900000000043
0.4	2.200000000000	2.800000000000	2.200000000110	2.800000000011
0.6	2.300000000000	2.700000000000	2.300000000000	2.700000000065
0.8	2.400000000000	2.600000000000	2.400000000000	2.600000000033
1	2.500000000000	2.500000000000	2.500000000060	2.500000004430

Table(8) accuracy of solutions of the example (2) where **ε=10^{-9}** , $x = 0.5$

The error $[\mathbb{E}(x)]_\gamma = \left|[\mu_a(x)]_\gamma - \mu_t(x)]_\gamma\right|$	
$[\mathbf{error}]_\gamma^{\mathbf{u}}$	$[\mathbf{error}]_\gamma^{\mathbf{l}}$
0	0
4.30002700113619E-11	5.40000000270879E-09
1.10000897279861E-11	1.10000009101441E-10
6.50000053781241E-11	0
3.29998250947483E-11	3.20188320301895E-13
4.42999992245063E-09	6.00000049644223E-11
MSE= 1.96321833272631E-17	**MSE**= 2.91757001343736E-17

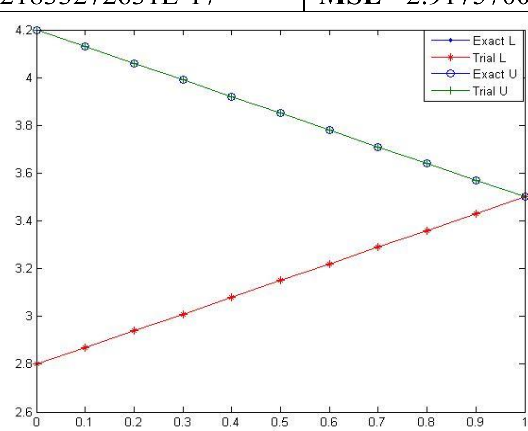

Figure (2) analytic and neural solution of example (2), with **ε=10^{-9}**

References

[1] G.S. Salman, K.M.M. Al-Abrahemee, Designing Feed Forward Fully Fuzzy Neural Network to Solve Fuzzy Singular Perturbed Volterra Integro-Differential Equations , Journal of Interdisciplinary Mathematics(2024).

[2] Badeye, S.R., Woldaregay, M.M. & Dinka, T.G., Solving singularly perturbed fredholm integro-differential equation using exact finite difference method, Badeye et al. BMC Research Notes,(2023).https://doi.org/10.1186/s13104-023-06488-8

[3] Al-Abrahemee, K.M.M.,Efficient Algorithm for Solving Fuzzy Singularly Perturbed Volterra Integro-Differential Equation,*Iraqi Journal of Science*., 64(11), pp. 5851–5865,2023. https://orcid.org/0000-0002-4705-6349.

[4] K.M.M. Al-Abrahemee, Modification of high performance training algorithm for solving singular perturbation partial differential equations with cubic convergence, Journal of Interdisciplinary Mathematics, 24(7), 2035-2047, (2021) https://doi.org/10.1080/09720502.2021.2001136.

[5] Rusul Kareem, K.M.M. Al-Abrahemee, Modification artificial neural networks for solving singular perturbation problems, Journal of Interdisciplinary Mathematics, (2022), https://doi.org/10.1080/09720502.2022.2072063.

[6] R.F. Kadam, K.M.M. Al-Abrahemee, On convergence of the Levenberg-Marquardt method underlocal error bound, problems, Journal of Interdisciplinary Mathematics, 25(5), 1495-1508, (2022), https://doi.org/10.1080/09720502.2022.2079228.

[7] Al-Abrahemee, K.M., Jabber, A.K., High Performance Training Algorithm with Strong Local Convergence Properties for Solving Some Fractional Integral Equation, AIP Conference Proceedings, Volume 2414, Issue 1,2023. https://doi.org/10.1063/5.0115034.

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

[9] R. M. Hristev , " The ANN Book ", Edition 1, (1998), https://www.academia.edu/44105267/Hritsev_The_ANN_Book

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

[11] R.F. Kadam, K.M.M. Al-Abrahemee. Neuro-fuzzy system for solving fuzzy singular perturbation problems. Journal of Interdisciplinary Mathematics. 25(5), 1509-1524, 2022. https://doi.org/10.1080/09720502.2022.2079229

[12] Al-Abrahemee, K.M.M.. A novel hybridized neuro-fuzzy model for solving fuzzy singular perturbation problems with initial conditions,*Journal of Interdisciplinary Mathematics*, 26(6), pp. 1287–1301,2023, https://doi.org/10.47974/JIM-1627

[13] D. and Kruse, R. Nauck, "Neuro-Fuzzy Classification with NEFCLASS," in *Operations Research Proceedings*, Berlin, pp. pp. 294-299,(1996). ISBN : 978-3-540-60806-6.

[14] Otadi M Mosleh M., "Simulation and Evaluation of Fuzzy Differential Equations by Fuzzy Neural Network," *Applied Soft Computing* , vol. 12, pp. 2817-2827, (2012). https://doi.org/10.1016/j.asoc.2012.03.041