Results in Applied Mathematics 16 (2022) 100344

Contents lists available at ScienceDirect

Results in Applied Mathematics

journal homepage: www.elsevier.com/locate/results-in-applied-mathematics

Comparative study of FeedForward and Radial Basis Function Neural Networks for solving an Environmental Boundary Value Problem

I. Famelis [a], A. Donas [a,b,*], G. Galanis [b]

[a] microSENSES Laboratory, Department of Electrical and Electronics Engineering, University of West Attica, Ag. Spyridonos Str., GR-12210 Egaleo, Greece
[b] Mathematical Modeling and Applications Laboratory, Hellenic Naval Academy, Hatzikiriakion, Piraeus 18539, Greece

ARTICLE INFO

Article history:
Received 30 September 2022
Received in revised form 1 November 2022
Accepted 3 November 2022
Available online 16 November 2022

Keywords:
Environmental Boundary Value Problem
FeedForward Neural Network
Radial Basis Function Neural Network
Stiff solver

ABSTRACT

The aim of this paper is to introduce an alternative approach, using Neural Networks, for the approximate solution of a Boundary Value Problem (B.V.P) for second order quadratic Differential Equations, which arises in the numerical prediction of meteorological parameters. We used two different types of neural networks and more specific a FeedForward and a Radial Basis Function Neural Network, based on an approach which first presented by Lagari et al. (1998). Our objective is to examine whether the proposed methodology satisfies the solution system of differential equations of the second order, by studying the residual of the method for different time periods. Furthermore, we present a new, neural network, trial solution based on rational approximation, highlighting the advantages and disadvantages of each architecture. Finally, we study in terms of defect, the obtained results with those given by a previous work introduced by Famelis and Tsitouras 2015, in order to compare the new approach with classical schemes of numerical analysis.

1. Introduction

The significant development of technology increased our computing potential, giving the opportunity to apply high computational cost methodologies efficiently. Among the applications that benefited from that event was the Artificial Neural Networks (ANNs). The ANNs were first introduced by Warren McCulloch and Walter Pitts in 1943, when they developed a computational model for neural networks based on an algorithm called threshold logic and since then various ANNs structures have been considered, providing powerful tools in various applications [1].

In this work we present an application of ANNs for solving differential equations. In particular, we have dealt with an Environmental Boundary Value Problem formed by second order quadratic differential equations, which arises in the numerical prediction of meteorological parameters. Our approach is inspired by works of Lagaris et al. [2], Simos and Famelis [3] and Hou et al. [4], which rely on the function approximation capabilities of the FeedForward Neural Networks. These methods construct a trial solution written in a differentiable, closed analytic form which contains a neural network as the approximation factor where the parameters are adjusted to minimize a cost function.

In addition to the use of the FeedForward structure, the Radial Basis Function Neural Network have been applied, as the initial approach presented a significant drawback related to stiff problems. Our objective is to analyze the behavior

* Correspondence to: Department of Electrical and Electronic Engineering, University of West Attica, microSENSES Laboratory, Ag. Spyridonos Str., GR-12210 Egaleo, Greece.
E-mail address: adonas@uniwa.gr (A. Donas).

https://doi.org/10.1016/j.rinam.2022.100344

of the various neural network structures, by examining whether and in what extent the proposed approach satisfies the solution system of differential equations of the second order, by checking the maximum absolute residual from the different study periods. Furthermore, we introduce a new, trial solution based on rational approximation contrasting the results derived from the implementations of the methodologies. Finally, we compare, in terms of defect (residual), the alternative methodology with a numerical analysis method and in particular with the shooting method [5], aiming to examine the efficiency of the providing tool in comparison to a classical numerical scheme.

The presented work is organized as follows: In Section 2 we analyze the framework of the problem under study, while in Section 3 we describe the different structures of the Artificial Neural Networks. The proposed methodology as well as the various case studies are presented in Section 4, where the obtained results are analyzed and discussed in detail. In Section 5 we introduce a new trial solution, emphasizing its disadvantages and advantages over the initial approach and in Section 6 we compare the alternative methodology with the shooting method, providing the corresponding results from each implementation. Finally, the main conclusions are summarized in Section 7.

2. The problem

In many cases such as climate change, transportations, marine pollution and ship safety, high quality environmental predictions–simulations are needed. Forecasting models for meteorological parameters are usually successful in simulating global environmental conditions, while on the other hand they do not respond relatively well to forecasts of local interest. This is due to the fact that the physical problems are multiparametric and as a result various factors are involved, such as the strong dependence on the initial and lateral conditions, the powerlessness to capture sub-scale phenomena and the parametrization of certain atmospheric or wave procedures [5–7].

A possible solution to address these issues is to enhance the model resolution, but still is not certain if this strategy leads to a substantial improvement of the forecast quality. Even if this is the case, the result would be a significant increase in computational cost. An alternative approach would be to improve the initial conditions based on assimilation systems, using post process algorithms. These tools are used for the enhancement of the final results based on statistical models (Neural networks, MOS methods, Kalman filters). In this procedure, a "cost function" measuring the bias ("the distance") of the model outcome and the environmental data is minimized [5–7]. The conventional approaches in computing distances involved in such cost-functions usually neglect the topological–geometrical properties of the data space under study by adopting least square methods based on classical Euclidean geometry tools. Though, the "distance/cost-function" should be better measured by means of Information Geometry theory [8–10].

Under the above framework the meteorological and the model data can be considered as elements of statistical manifolds. The minimum distance between two elements f_1 and f_2 of such manifolds is defined by the corresponding geodesic ω which is the minimum length curve that connects them [5–7]. Such a curve

$$\omega = (\omega_i) : \mathbb{R} \to S$$

satisfies the following system of 2nd order differential equations:

$$\omega_i''(t) + \sum_{j,k=1}^{n} \Gamma_{jk}^{i}(t)\omega_j'(t)\omega_k'(t) = 0,\ i = 1, 2, \ldots, n$$

under the condition $\omega(0) = f_1, \omega(1) = f_2$.

The two parameter Weibull distributions have been proved a proper choice for fitting wind and wave data [5–7]. These distributions form a 2-dimensional statistical manifold with $\xi = [\alpha, \beta]$, $\Xi = \{[\alpha, \beta]; \alpha \text{ and } \beta > 0\}$ (where α is the shape and β the scale parameter) and

$$p(x) = \frac{\alpha}{\beta}\left(\frac{x}{\beta}\right)^{\alpha-1} e^{-\left(\frac{x}{\beta}\right)^{\alpha}},\ \alpha, \beta > 0$$

When we consider two members of the Weibull statistical manifold $\xi_0 = [\alpha_0, \beta_0]$, $\xi_1 = [\alpha_1, \beta_1]$, the geodesic satisfies the following system of quadratic 2nd order differential equations:

$$\omega_1''(t) + \frac{6\left(\gamma\alpha_0 - \alpha_0 - \frac{\pi^2}{6}\right)}{\pi^2\beta_0}\left(\omega_1'(t)\right)^2 + \frac{12\left(\gamma^2 - 2\gamma + \frac{\pi^2}{6} + 1\right)}{\pi^2\alpha_0}\omega_1'(t)\omega_2'(t)$$
$$- \frac{6(1-\gamma)\beta_0\left(\gamma^2 - 2\gamma + \frac{\pi^2}{6} + 1\right)}{\pi^2 a^3}\left(\omega_2'(t)\right)^2 = 0$$
$$\omega_2''(t) - \frac{\alpha_0^3}{\pi^2\beta_0^2}\left(\omega_1'(t)\right)^2 + \frac{12\alpha_0(1-\gamma)}{\pi^2\beta_0}\omega_1'(t)\,\omega_2'(t)$$
$$- \frac{6\left(\gamma^2 - 2\gamma + \frac{\pi 2}{6} + 1\right)}{\pi^2\alpha_0}\left(\omega_2'(t)\right)^2 = 0$$

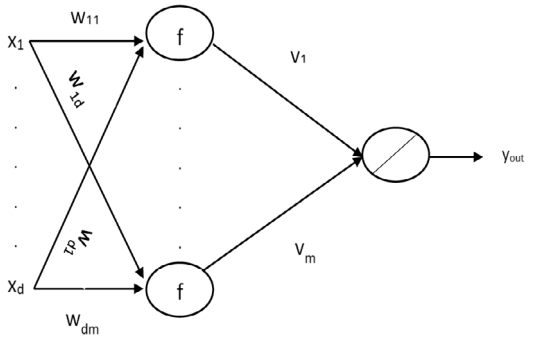

Fig. 1. FeedForward Neural Network.

under the conditions $\omega(0) = \begin{bmatrix} \alpha_0 \\ \beta_0 \end{bmatrix}, \omega(1) = \begin{bmatrix} \alpha_1 \\ \beta_1 \end{bmatrix}$, where $\omega(t) = \begin{bmatrix} \omega_1(t) \\ \omega_2(t) \end{bmatrix}$ and is $\gamma \approx 0.57721566490153286060$ (the Euler gamma). The above is a second order Boundary Value Problem of a form $\tilde{\omega}'' = F\left(\tilde{\omega}, \tilde{\omega}'\right)$, where $\tilde{\omega} = \begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix}$ are defined on the interval [0, 1] and can be written in the following format:

$$\begin{aligned} &\omega_1^{(2)} + \alpha_{11}(\omega_1^{(1)})^2 + \alpha_{12}\omega_1^{(1)}\omega_2^{(1)} + \alpha_{22}(\omega_2^{(1)})^2 = 0 \\ &\omega_2^{(2)} + b_{11}(\omega_1^{(1)})^2 + b_{12}\omega_1^{(1)}\omega_2^{(1)} + b_{22}(\omega_2^{(1)})^2 = 0 \end{aligned} \tag{*}$$

In previous works [5–7] the system (*) has been addressed using numerical analysis tools, such as the classical Newton's and Quasi-Newton methods [7], the Mono-Implicit Runge–Kutta (MIRK) schemes [6] and the Quadric Shooting method [5]. The results of these studies showed that using high order methods may have high computational cost but they provide more robust results e.g., less sensitive solutions to the choice of the initial conditions [5].

In this study, our intention is to attend the solution of the same system of differential equations by taking advantage of the ability of neural networks to approximate any function [2–4,11], as described in Section 4.

3. The different types of ANN

In this section, we introduce the different types of Artificial Neural Networks (ANNs) which have been applied. More specific, we carried out two case studies with dissimilar trial solutions where each one contains a FeedForward Neural Network (FFNN) and a RadialBasisFunction Neural Network (RBF NN) respectively.

3.1. FeedForward Neural Network

The FFNN structure is one the simplest and most common types of artificial neural networks and is used in a variety of applications. It is a multi-layer network where each neuron in the previous layer connects to all neurons in the next layer. In particular, the first level is the input layer where the data enter the network, while the last layer, or layers, is the output level. In between these two there can be many hidden layers with different number of neurons each, which enables the creation of neural networks with multiple structures.

Regardless of the net's structure, the input data are processed in only one direction from the first layer to the last. This process is called front propagated. Additionally, it should be noted that both the number of neurons and the hidden levels affects the complexity of the network and therefore the desired result. Furthermore, an important factor in functioning of neural networks are the activation functions, as they determine whether and to what extent each neuron will be activated, which affects their contribution to the final outcome. There are several types of transfer functions, such as the sigmoid and the hyperbolic tangent function or the ReLu activation function, the use of which depends on the application. More information about the FFNNs can be found in the work of Martin T. Hagan [1].

In this work a three-level FFNN have been applied with one hidden layer, the sigmoid function $f(x) = \frac{1}{1+e^{-x}}$, as activation function and one linear output level (Fig. 1). The choice of this architecture was not random but arose through a comparative study. In particular, we applied multilayer FFNNs and the obtained results was identical to those of the three-level FFNN, with the difference that the computational cost intensified significantly and therefore the increase in complexity was not beneficial.

Moving forward the output of the one hidden layer is:

$$H = f\left(W^T x + b\right) \tag{1}$$

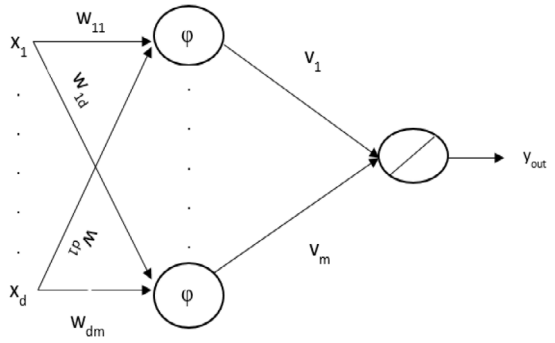

Fig. 2. RBF Neural Network.

where, $f(\bullet)$ is the transfer function and the parameter W is a nxm weight matrix, which connects the input to the hidden layer. Furthermore, the variable x expresses the $nx1$ dimensional column vector of input elements, while the parameter b is a $mx1$ dimensional column vector of bias. Note here that, the quantities n and m represent the number of the input elements and the number of neurons of the hidden level respectively.

The variable H is inserted in to the output layer and multiplied by the $mx1$ dimensional column vector V, which connects the neurons of the hidden layer to the output level. Finally, in the output we added the offset (bias) parameter $\hat{b}$. The mathematical formulation of the above expression is:

$$y_{out} = V^T * H + \hat{b} \tag{2}$$

where, V^T is the transpose matrix of V and the variable y_{out} express the predicted output of the FFNN. Replacing the (1) in (2) follows that:

$$y_{out} = V^T * f\left(W^T * x + b\right) + \hat{b} \tag{3}$$

3.2. Radial Basis Function Neural Network

The first appearance of Radial Basis Function Neural Networks was in 1988 [12] and introduced by Broomhead and Lowe. It is a standard three-layer neural network, with the first input layer consisting of d input neurons, one hidden layer residing of m radial basis functions, as activation functions and a third linear output level (Fig. 2). The transfer function for a RBF NN implementation is not unique, but on the contrary can be one of the Gaussian, the Multiquadric, or another and has the following form [4,12,13]:

$$\varphi_j(x_i) = \varphi(\|w_j - x_i\| b_j) \tag{4}$$

$$for\ i = 1, \ldots, d\ and\ j = 1, \ldots, m$$

where, the quantity $\|w_j - x_i\|$ express the distance between the x input data and the w centers, while the b is the bias parameter.

The main characteristic of Radial Basis functions is that their response decreases or increases monotonically when moving away from a centroid. In particular, the Gaussian Radial Basis function monotonically decreases, while the Multiquadric function increases monotonically [13]. Moreover, another essential asset of the Gaussian is that it is local, which means that the output is close to zero if you move very far from the center point [1], while on the other hand the Multiquadric function is global [13]. Here the selected radial basis activation function was the Gaussian, $\varphi(z) = e^{-z^2}$, as it is commonly used in RBF NN applications [13] and also because it was observed, after tuning, that fits best with our study.

Based on the above the output of the RBF NN has the following form [12,13]:

$$y_{out} = \sum_{j=1}^{m} v_j \varphi_j(x_i) + \tilde{b} \tag{5}$$

where, the quantity $\tilde{b}$ is a 1-dimensional offset factor and the variable v determine the effect that each neuron, of the hidden-layer, will have on the output. Replacing the formula (4) into (5) we have the final form of the RBF's neural network output:

$$y_{out} = \sum_{j=1}^{m} v_j \varphi(\|w_j - x_i\| b_j) + \tilde{b} \tag{6}$$

A crucial factor in the modeling of RBF NNs is the way parameters are selected, as it can affect both the final outcome and the computational cost. In this study, we considered that the centers w, which must belong to the same definition field as the input data, are stable and were randomly selected, while on the contrary the parameters v, b and $\tilde{b}$ will be tuning during the training of RBF NN, in order to fit the data well [1].

4. The proposed methodology

In order to solve the 2-dimensional system of differential equations (*) we consider the following suitably differentiable parameterized trial solution [2–4,11]:

$$Y_{\mathrm{tr(i)}}(x, p_i) = \frac{b\omega_1(a) - a\omega_2(b)}{b-a} + \frac{\omega_2(b) - \omega_1(a)}{b-a}x + (x-a)(x-b)N_i(x, p_i) \qquad (**)$$

for $i = 1, 2$.

Here the quantity $N(x, p)$ is a single neural network output with p parameters, while the variables α, b are the limits of the definition field (here $\alpha = 0$ and b = 1) and the $\omega_1(a)$, $\omega_2(b)$ are the boundary conditions. Note at this point that, $Y_{\mathrm{tr}}(x)$ is twice differentiable and assures that the boundary conditions are satisfied. Moreover, the boundary conditions are not the same for the various time periods and can be found in the work of Famelis and Tsitouras [5].

Our goal is to minimize a 2- dimensional cost function, in order to examine whether the proposed methodology satisfies the solution system of differential equations of the second order, of the form [4]:

$$Cost\left[\vec{p}\right] = \sum_{k=1}^{2}\sum_{i=1}^{2}\left[\frac{d^2 Y_{tr(k)}}{dx^2}(x_i, p_i) - f_k\left(x_i, \Psi_{\mathrm{tr(1)}}(x_i, p_i), \Psi_{tr(2)}(x_i, p_i), \Psi'_{t_1}(x_i, p_i), \Psi'_{t_2}(x_i, p_i)\right)\right]^2.$$

As a minimization technique the Levenberg–Marquardt method was selected. The Levenberg–Marquardt algorithm combines the steepest descent method and the Gauss–Newton algorithm, inheriting the advantages of both, as it incorporates the speed of the Gauss–Newton algorithm and the stability of the steepest descent method [14]. Its main idea is that it performs a combined training process: around areas with intense complexity, the method switches to the steepest descent algorithm, until the local curvature is proper to make a quadratic approximation and then it approximately becomes the Gauss–Newton algorithm, which can speed up the convergence significant [14].

Moreover, the number of neurons in the case studies was not fixed, as six combinations were selected and tested during the training process. Then, for each number of neurons we keep the absolute maximum residual of the best solution for the training points, as well as the optimal parameters of it. Note here that, the concept of best solution concerns the extent to which it satisfied the solution system of differential equations, based on the trial solution which is used. When the training process is over the parameters of the best fitting are used, in order to examine the accuracy of the model in a denser dataset within the interval [0,1]. The results derived from the above implementation are presented in the following paragraphs.

4.1. Case study 1

The first attempt to solve the system (*) was made using the FeedForward Neural Network. In particular, two different neural networks, with one hidden layer and two different linear output levels, which contribute jointly to the final outcome were used for each of the solutions ω_1 and ω_2, as shown in the following illustration (see Fig. 3).

Where the outputs y_1 and y_2 replaced in (**) and are essentially the term $N(x, p)$, while the indication f is the activation function.

4.1.1. The results from the FFNN implementation

In this paragraph we present the results obtained from the implementation of FeedForward Neural Network for the various study periods (column 1). Table 1 presents the absolute maximum residual of the model in the training points (column 2) and the corresponding residual for the evaluation points (Column 3). Moreover, Fig. 4 presents the number of the decimal digits of the residual, showing its variability between the training and the evaluation process.

The data in the above table shows that, in the training process (column 2) the model performs quite well, as the maximum residual is 2.44E−09 and was observed in November. However, when we calculate the corresponding residual in the validation dataset (column 3) we see a huge difference, which is sometimes significant, such as the period of August or June. This deviation observed in this particular case, but also in others (Fig. 4), is due to the nature of the problem which can be described as stiff, as the solution ω_1 increases while at the same time the solution ω_2 decreases as shown below (Fig. 5).

Stiffness definition is difficult to formulate. In general, the stiffness phenomenon occurs when a numerical method, in order to provide a reliable solution with the desirable accuracy, continuously reduces the step size. A typical example of stiff problems are those that have some solution elements that change very fast whereas other very slow [15].

This led us to conclude that the model made on the basis of the FeedForward Neural Network is unable to cope with stiff problems and additionally performs overfitting, which means that it memorizes the training data instead of generalizing

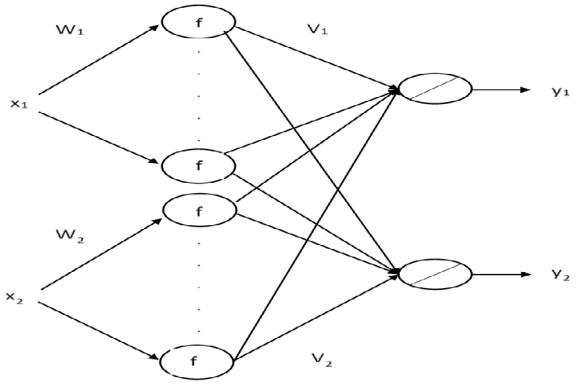

Fig. 3. Two FeedForward Neural Networks.

Table 1
Residual of the FFNN implementation between training and evaluation process.

	FFNN training	FFNN evaluation
Jan no c.	4.22E−15	1.79E−04
Jan wi c.	4.64E−15	6.33E−05
Feb no c.	1.36E−15	2.57E−05
Feb wi c.	6.99E−15	6.52E−05
Mar no c.	5.33E−15	4.85E−04
Mar wi c.	3.66E−15	4.20E−04
Apr no c.	3.44E−14	9.50E−03
Apr wi c.	3.33E−15	4.49E−05
May no c.	8.88E−11	2.95E−05
May wi c.	1.57E−14	3.54E−06
Jun no c.	4.62E−14	1.53E+01
Jun wi c.	1.42E−14	1.10E+00
Jul no c.	4.44E−15	1.11E−02
Jul wi c.	9.33E−15	1.96E−03
Aug no c.	5.68E−13	3.14E+01
Aug wi c.	9.95E−14	1.44E+01
Sep no c.	1.60E−14	9.16E−02
Sep wi c.	6.99E−15	1.61E−02
Oct no c.	9.49E−15	6.00E−03
Oct wi c.	2.33E−15	5.53E−05
Nov no c.	2.51E−15	1.12E−03
Nov wi c.	2.44E−09	4.11E−04
Dec no c.	1.17E−14	1.85E−03
Dec wi c.	2.60E−14	1.24E−04

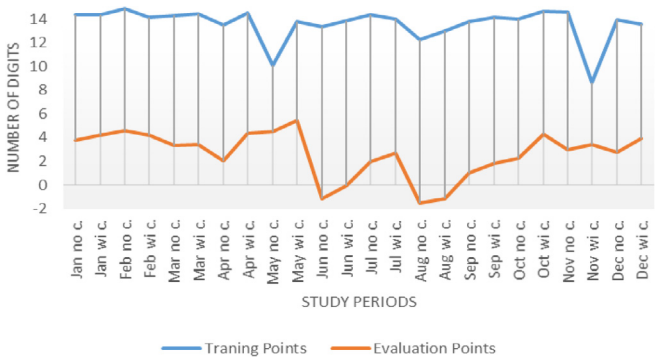

Fig. 4. The variability of residual between training and evaluation points. Case Study 1.

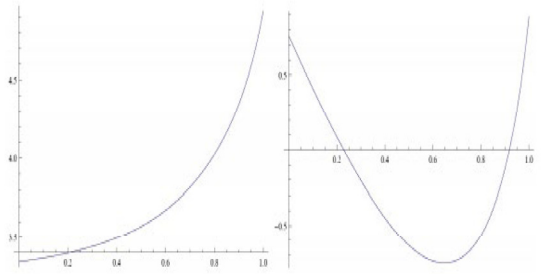

Fig. 5. Solutions of ω_1 and ω_2 for August with current [5].

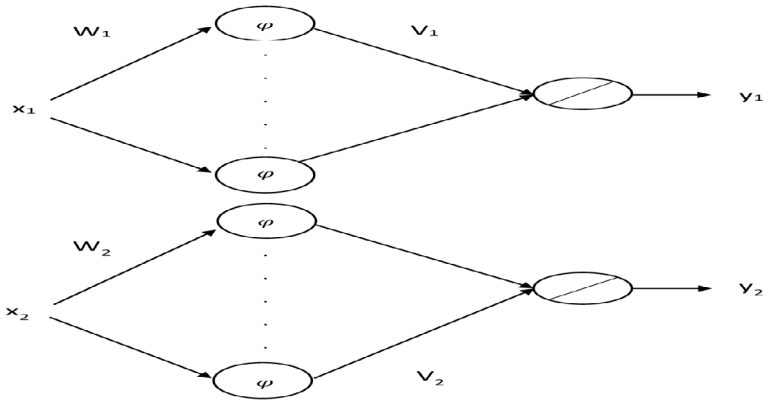

Fig. 6. Two RBF Neural Networks.

over the problem [16,17]. Therefore, we proceeded to study other neural networks structures in order to address this issue.

4.2. Case study 2

The second attempt to solve (*) was made using the Radial Basis Function Neural Networks, where the centers w are fixed and selected randomly from the interval [0,1]. The parameters b, $\widetilde{b}$ and V are unknown and will be adjusted during the training process. Unlike the previous case study (Case study 1), where the two neural networks contribute jointly, here the networks are distinct without any mutual benefaction to the final output as shown in the following Figure (Fig. 6).

Here the outputs y_1 and y_2 are the term $N(x, p)$, which replaced in formula (**) and giving the final form of the RBF NN trial solution. The indication φ represent the Gaussian activation function.

4.2.1. The results from the RBF NN implementation

This section presents the results obtained from the of Radial Basis Neural Network implementation, where Table 2 presents the absolute maximum residual of the model in the training points (column 2) and the absolute maximum residual for the evaluation points (Column 3). Furthermore, the Figure (Fig. 7) below presents once again the number of the decimal digits of the residual and the column 1 of Table 2 the different study periods.

The above data shows that the RBF approach may give worse results at the training process, compared to FFNN usage, but does not exhibit the same volatility in terms of its behavior in calculating the maximum absolute residual at the evaluation points and thus overfitting did not occur. This absence of strong fluctuation, as shows in Fig. 7 reveals the stability of this structure, which is observed even in cases of stiff problems. In particular, for the study period of August with current (Aug wi c.), but not only, we see this important difference between the methods, as the absolute maximum residual derived from the FFNN and RBF NN implementations for the training set was 9.95e−14 and 2.67E−04, while the corresponding results for the verification points was 1.44E+01 and 1.04E−04 respectively.

5. Alternative trial solution

After the implementation of the initial trial solution (**), we introduce here a new one based on rational approximation, presenting the results we have extracted from its application to the problem under study. More specifically, the new trial

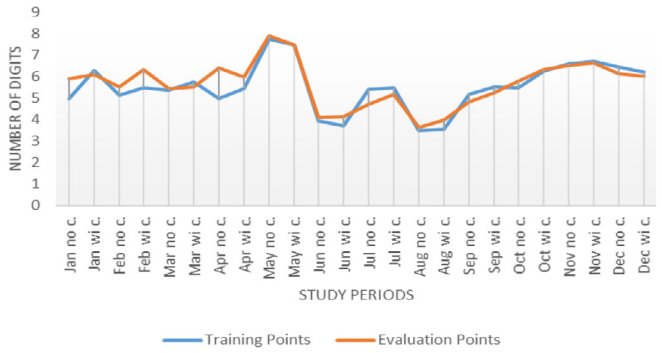

Fig. 7. The variability of residual between training and evaluation points. Case Study 2.

Table 2
Residual of the RBF NN implementation between training and evaluation process.

	RBF NN training	RBF NN evaluation
Jan no c.	1.02E−05	1.20E−06
Jan wi c.	5.25E−07	7.71E−07
Feb no c.	7.48E−06	2.87E−06
Feb wi c.	3.28E−06	4.71E−07
Mar no c.	4.11E−06	3.66E−06
Mar wi c.	1.72E−06	3.12E−06
Apr no c.	1.06E−05	3.98E−07
Apr wi c.	3.51E−06	1.05E−06
May no c.	1.82E−08	1.26E−08
May wi c.	3.16E−08	3.26E−08
Jun no c.	1.12E−04	8.06E−05
Jun wi c.	1.87E−04	7.42E−05
Jul no c.	4.02E−06	1.86E−05
Jul wi c.	3.40E−06	6.78E−06
Aug no c.	3.31E−04	2.35E−04
Aug wi c.	2.67E−04	1.04E−04
Sep no c.	6.75E−06	1.50E−05
Sep wi c.	2.96E−06	5.75E−06
Oct no c.	3.26E−06	1.62E−06
Oct wi c.	5.46E−07	4.77E−07
Nov no c.	2.52E−07	2.99E−07
Nov wi c.	1.94E−07	2.22E−07
Dec no c.	3.53E−07	7.39E−07
Dec wi c.	6.14E−07	9.70E−07

solution has the following format:

$$Y_{\text{newtr(i)}}\left(x, p_{i,1}, p_{i,2}\right) = \frac{b\omega_1(a) - a\omega_2(b)}{b-a} + \frac{\omega_2(b) - \omega_1(a)}{b-a}x + (x-a)(x-b)\frac{N_{i,1}\left(x, p_{i,1}\right)}{1 + N_{i,2}\left(x, p_{i,2}\right)}$$

for $i = 1, 2$.

The terms $N_1(x, p_1)$ and $N_2(x, p_2)$ represent the outputs of the artificial neural networks with one input unit for x and weights p.

In comparison to the initial FFNN approach, which used two distinct neural networks, the alternative one uses four. This increases the computing cost as the complexity of the system intensified, but significantly improve the obtained results as shown below. On the other hand, the results of its implementation using the RBF NN were worse than the corresponding results presented in Section 4 (Case Study 2) and thus they are not included in this study.

Comparing the above results to those presented in Table 1, we observe that the new trial solution has largely dealt with overfitting in most cases, as the absolute maximum residual of the training process (Fig. 8) was almost identical with the corresponding of the evaluation process. However, in this case too, the implementation of the FFNN has not been able to address satisfactory problems which present a stiff behavior, such as the cases of August and Julie.

6. Comparative results of the different methods

In this section we present the comparative results between the proposed methodology and that of the Famelis and Tsitouras [5], which dealt with the same B.V.P, using tools of numerical analysis. More specifically, the shooting method

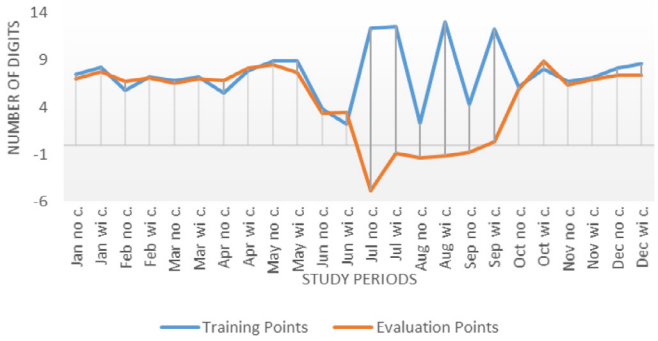

Fig. 8. The variability of residual between training and evaluation points. New Trial solution.

Table 3
Residual of the FFNN implementation between training and evaluation process. New trial solution.

	FFNN 10 points	FFNN 100 points
Jan no c.	3.06E−08	9.27E−08
Jan wi c.	5.73E−09	1.65E−08
Feb no c.	1.51E−06	2.00E−07
Feb wi c.	6.53E−08	7.60E−08
Mar no c.	1.55E−07	3.15E−07
Mar wi c.	6.70E−08	9.48E−08
Apr no c.	2.94E−06	1.50E−07
Apr wi c.	1.34E−08	7.69E−09
May no c.	1.22E−09	3.17E−09
May wi c.	1.29E−09	2.32E−08
Jun no c.	1.49E−04	4.43E−04
Jun wi c.	6.13E−03	3.64E−04
Jul no c.	4.76E−13	6.18E+04
Jul wi c.	2.93E−13	6.95E+00
Aug no c.	5.28E−03	2.50E+01
Aug wi c.	9.37E−14	1.56E+01
Sep no c.	5.73E−05	6.28E+00
Sep wi c.	5.19E−13	4.43E−01
Oct no c.	6.61E−07	1.21E−06
Oct wi c.	8.30E−09	1.30E−09
Nov no c.	1.71E−07	4.04E−07
Nov wi c.	7.22E−08	1.30E−07
Dec no c.	8.02E−09	4.41E−08
Dec wi c.	2.65E−09	4.22E−08

was implemented, which applies Runge–Kutta integrators specially made for quadric stiff ordinary differential equations [5]. In particular, the classical Singly Diagonally Implicit Runge–Kutta SDIRK 4(3) and a quadric SDIRK 5(3) methods have been applied. More information about those methodologies can be found in [5].

The following table (Table 4) shows the aggregated comparative results between the different methods. Note here that the indications $O(1)$, $O(0.1)$, $O(0.01)$ and «*» express the approximation of the unknown initial values [5] and the failure of the method for the different study periods respectively. Moreover, from the neural network based method we use only the results derived from the evaluation process (see Table 3).

Examining the above data it seems that, the classical methods of numerical analysis and especially SDIRK 5(3) approach produces better results in general, compared to the proposed method based on neural networks. However, in the study periods of June and August the RBF NN methodology presents a noticeable improvement, as the absolute maximum residual decreased from 6.0E−3 to 8.06E−5 in Jun no.c, from 9.30E−5 to 7.42E−5 in Jun wi.c and from 9.50E−2 to 2.35E−4 in Aug no.c. This enhancement is particularly important as the specific study periods concern stiff problems, which demonstrates that in some cases the use of neural networks and in particular the RBF NNs structure can overcome an eminently stiff solver, providing a reliable and efficient tool.

7. Conclusions

In this study we have studied the numerical solution of an Environmental Boundary Value Problem for a second order quadratic system of Differential Equations, using a trial solution that contains a neural network. More specifically,

Table 4
Defect/Residual for the 24 study periods [5].

	Defect/Residual for the 24 study periods								
	SDIRK43			SDIRK53			FFNN	RBF NN	FFNN new trial solution
	Initial conditions estimation								
	O(1)	O(0.1)	O(0.01)	O(1)	O(0.1)	O(0.01)	100 evaluation points	100 evaluation points	100 evaluation points
Jan no c.	2.30E−07	2.30E−07	2.30E−07	4.10E−09	4.10E−09	4.10E−09	1.79E−04	1.20E−06	9.27E−08
Jan wi c.	6.10E−08	6.10E−08	6.10E−08	8.20E−10	8.20E−10	8.20E−10	6.33E−05	7.71E−07	1.65E−08
Feb no c.	2.30E−08	2.30E−08	2.30E−08	2.40E−10	2.40E−10	2.40E−10	2.57E−05	2.87E−06	2.00E−07
Feb wi c.	1.60E−08	1.60E−08	1.60E−08	1.50E−10	1.50E−10	1.50E−10	6.52E−05	4.71E−07	7.60E−08
Mar no c.	1.80E−07	4.50E−07	4.50E−07	9.00E−09	9.00E−09	9.00E−09	4.85E−04	3.66E−06	3.15E−07
Mar wi c.	1.80E−07	4.50E−07	1.70E−07	1.50E−10	2.90E−09	2.90E−09	4.20E−04	3.12E−06	9.48E−08
Apr no c.	1.10E−07	1.10E−07	1.10E−07	2.90E−09	1.80E−09	1.80E−09	9.50E−03	3.98E−07	1.50E−07
Apr wi c.	7.30E−09	7.30E−09	7.30E−09	7.30E−11	7.30E−11	7.00E−11	4.49E−05	1.05E−06	7.69E−09
May no c.	1.20E−11	1.30E−11	1.20E−11	1.30E−11	1.10E−11	1.10E−11	2.95E−05	1.26E−08	3.17E−09
May wi c.	1.30E−11	1.20E−11	1.20E−11	1.20E−11	1.20E−11	1.20E−11	3.54E−06	3.26E−08	2.32E−08
Jun no c.	*	*	*	*	2.50E−03	6.40E−03	1.53E+01	8.06E−05	4.43E−04
Jun wi c.	*	*	*	*	9.30E−05	9.30E−05	1.10E+00	7.42E−05	3.64E−04
Jul no c.	*	9.00E−06	9.00E−06	*	3.50E−07	3.50E−07	1.11E−02	1.86E−05	6.18E+04
Jul wi c.	*	1.90E−06	1.90E−06	*	5.10E−08	5.10E−08	1.96E−03	6.78E−06	6.95E+00
Aug no c.	*	*	*	*	*	9.50E−02	3.14E+01	2.35E−04	2.50E+01
Aug wi c.	*	*	*	*	2.10E−03	2.10E−03	1.44E+01	1.04E−04	1.56E+01
Sep no c.	*	7.20E−05	7.20E−06	*	3.30E−06	3.30E−06	9.16E−02	1.50E−05	6.28E+00
Sep wi c.	*	7.60E−06	7.60E−06	*	2.80E−07	2.80E−07	1.61E−02	5.75E−06	4.43E−01
Oct no c.	*	7.60E−06	1.80E−06	*	5.60E−08	5.60E−08	6.00E−03	1.62E−06	1.21E−06
Oct wi c.	2.70E−08	2.70E−08	2.70E−08	3.50E−10	3.50E−10	3.60E−10	5.53E−05	4.77E−07	1.30E−09
Nov no c.	5.10E−10	5.10E−10	5.10E−10	1.30E−11	1.30E−11	1.80E−11	1.12E−03	2.99E−07	4.04E−07
Nov wi c.	3.10E−10	3.10E−10	3.10E−10	1.30E−11	1.20E−11	1.20E−11	4.11E−04	2.22E−07	1.30E−07
Dec no c.	2.70E−08	2.70E−08	2.70E−08	3.00E−10	3.00E−10	3.00E−10	1.85E−03	7.39E−07	4.41E−08
Dec wi c.	1.50E−08	1.50E−08	1.50E−08	1.40E−10	1.40E−10	1.40E−10	1.24E−04	9.70E−07	4.22E−08

we carried out two case studies with different forms of neural networks and in particular the FeedForward and the RadialBasisFunction Neural Networks.

Our goal was to minimize the cost function and thus reduce the residual (distance), so that the proposed methodology meets the solution system of differential equations of the second order. The implementation of the FeedForward Neural Network gave better results at the training process, compared to the corresponding evaluation procedure, as was shown for instance in the case of August. This significant discrepancy during this period (Fig. 5), shows that FFNN form is over trained (overfitting) and therefore our model based on this structure is unable to predict a trend in challenging problems. This phenomenon that occurred was due to the nature of the problem, at those study periods, which was stiff.

This conclusion led us to examine other types of neural networks and in particular the RBF NN. The results obtained from this application was better in terms of overfitting, as this issue was fully addressed (Fig. 7). Furthermore, this consistent behavior was extended to all study periods, which makes this approach a nifty tool to deal with stiff problems also. However, the RBF structure did not present the same satisfactorily results, as the FFNN did, regarding to the training process (Table 2).

Moving forward, we introduced an alternative trial solution based on rational approximation. The results derived from this application significantly improved FFNN's behavior as the overfitting was effectively addressed in the largest number of cases. Yet, in this case too, the FeedForward Neural Network does not exhibit the same smooth performance as the RBF NN did in the various study periods, as in the period of August.

Finally, in the last section we compare the results obtained from the proposed methodology with those derived from a previous work, introduced by Famelis and Tsitouras [5], in order to compare the alternative approach with classical schemes of numerical analysis. A noticeable improvement (Table 4) was observed only from the implementation of RBF NN, during the study periods of the Jun no.c, Jun wi.c Aug no.c and Aug wi.c, as the maximum reduction in residual (defect) was in the range of 10^{-2}. The specific study periods are concern stiff problems and therefore the use of the neural network based method and especially the RBF NN structure shows that, the proposed method can be used as an effective stiff solver which is able, under certain conditions, to compete with classical methodologies.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work has been funded by the Unit of Economic and Administrative Support (E.L.K.E) of University of West Attica.

References

[1] Hagan M, Demuth H, Beale M, De Jesús O. Neural network design-2nd Edition. 2014.
[2] Lagaris I, Likas A, Member, IEEE, Fotiadis D. Artificial neural networks for solving ordinary and partial differential equations. Appl Mech Rev 1998;9:987–1000.
[3] Simos TE, Famelis I. A neural network training algorithm for singular perturbation boundary value problems. Neural Comput Appl 2021. in press.
[4] Hou C, Simos TE, Famelis I. Neural network solution of pantograph type differential equations. Math Methods Appl Sci 2020;43:3369–74.
[5] Famelis I, Tsitouras Ch. Quadric shooting solution for an environmental parameter prediction model. Far East J Appl Math 2015;91:81–98.
[6] Famelis I. Runge–Kutta solutions for an environmental parameter prediction boundary value problem. J Coupled Syst Multiscale Dyn Exp 2014;2:62–9.
[7] Famelis I, Galanis G, Ehrhardt M, Triantafyllou D. Classical and Quasi-Newton methods for a meteorological parameters prediction boundary value problem. Appl Math Inf Sci 2014;8:2683–93.
[8] Amari S. Differential–Geometrical methods in statistics. Berlin: Springer-Verlag; 1985.
[9] Amari S, Nagaoka H. Methods of information geometry. Oxford: Oxford University Press; 2000.
[10] Galanis G, Famelis I, Kalogeri Ch, Kallos G. Numerical and geometric optimization techniques for environmental prediction systems. In: SIAM conference on mathematical and computational issues in the geosciences Padova, Italy. 2013.
[11] Lagari P, Tsoukalas L, Safarkhani S, Lagaris I. Systematic construction of neural forms for solving partial differential equations inside rectangular domains, subject to initial, boundary and interface conditions. Int J Artif Intell Tools 2020;29:12.
[12] Zainuddin Z, Pauline O. Function approximation using artificial neural networks. Int J Syst Appl Eng Dev 2007;4:173–8.
[13] Orr M. Introduction to radial basis function networks. Edinburgh: University of Edinburgh; 1996.
[14] Wilamowski BM, Irwin JD. Intelligent systems. Boca Raton: CRC Press; 2018.
[15] Famelis I, Kaloutsa V. Parameterized neural network training for the solution of a class of stiff initial value systems. Neural Comput Appl 2021;33:3363–70.
[16] Jabbar HK, Khan DRRZ. Methods to avoid over-fitting and under-fitting in supervised machine learning (Comparative Study). In: Conference on computer science, communication and instrumentation devices. Kochi, India; 2014.
[17] Gavrilov A, Jordache A, Vasdani M, Deng J. Preventing model overfitting and underfitting in convolutional neural networks. Int J Softw Sci Comput Intell 2018;10:19–28.

Ioannis Th. Famelis is a Professor at the Department of Electrical and Electronics Engineering of the School of Mechanics, University of West Attica, Athens, Greece. He is a member of the microSENSES research laboratory. His research interests include Computational Mathematics, Numerical Analysis of Differential Equations, Symbolic Computations and Applications of Neural Networks and Computational Intelligence methods in Numerical Analysis.

Donas Athanasios is a Ph.D. student in the Department of Electrical and Electronic Engineering of the University of Western Attica. He is a member of the microSENSES Laboratory of the Electrical and Electronic Engineering Department and cooperates with the Mathematical Modeling and Applications Laboratory of the Hellenic Naval Academy. His research interests include Numerical Analysis of Differential Equations, Applications of Neural Networks and Optimization Methods for Numerical Models.

Dr. George Galanis is a Professor and Head of the Section of Mathematics of the Hellenic Naval Academy. He is also a Visiting Professor of the Naval Ocean Analysis and Prediction Laboratory, Naval Postgraduate School, Monterey, USA. Dr. Galanis leads the Mathematical Modeling and Applications Laboratory of the Hellenic Naval Academy and his research interests include Mathematical Modeling, Information Geometry and Optimization Methods for Numerical Models.