

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

Harnessing Machine Learning for Fluid Dynamics: A Neural Network Approach to the Blasius Equation

Saira Sultan

Department of Mathematics, University of Management and Technology Lahore,
Sialkot Campus Email: 23001265006@skt.umd.edu.pk

Shahzad Ahmad

Department of Mathematics, University of Management and Technology Lahore,
Sialkot Campus Email: shahzad.ahmed@skt.umd.edu.pk

Amir Rizwan

Department of Mathematical Sciences, University of Gujarat
Email: aamirphd01@gmail.com

Muhammad Farman

Department of Mathematics, Near East University 99138, Nicosia/Mersin 10, Turkey
Email: farmanlink@gmail.com

ABSTRACT

The paper represents a recent supervised learning-based neural network model of solving the classical Blasius equation, a fundamental theory of fluid dynamics, the basis of the boundary layer of a flat plate laminar flow. Computational cost and convergence stability is typically a challenge to traditional analytical and numerical approaches to such nonlinear differential equations. The proposed study will address these limitations by taking the strength of machine learning based on a supervised neural architecture to learn the solution to the Blasius equation with great accuracy and efficiency.

The Mean Squared Error (MSE) is used as the optimization criterion of the neural network model, the gradient behavior and adaptive learning rate (μ), is noticed to reveal convergence properties and is used to guarantee a strong training dynamics. The evaluation of the performance is conducted to the full extent relying on the error histograms, regression, and fitness plots, and with their help, the predictive ability of the network and the generalization ability are quantified and visualized.

Besides, the proposed supervised learning model is highly scalable to nonlinear boundary value problems in fluid mechanics that can provide smooth, physically consistent, and computationally efficient solutions. The results validate the claim that implementing the machine learning paradigms on classical fluid dynamics does not just augment the precision of the solutions but contributes to better convergence and eliminates excessive computing.

Keywords: Blasius Equation- Levenberg-Marquardt Algorithm Combination,

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

Supervised Learning, Neural Network, Fitting Tool, Matlab, Function, Approximation, Differential Equations, Mathematica, Training, Validation And Testing.

1. Introduction:

The twenty-first century has seen the scientific computing paradigm shift, with machine learning (ML) and artificial intelligence (AI) starting to establish a new definition of how to model and perceive complex physical systems. Among the greatest effects of this revolution is the body of work currently called fluid dynamics, which comprise the foundations of the modern-day engineering applications, including aerodynamics and heat transfer, as well as biological fluids and fluid systems in industry. Historically, the resolution of nonlinear partial differential equations (PDEs) that characterize the behavior of fluids has been based on the approximation methods that are analytical and lack computational efficiency.

There is the Blasius equation, described by Heinrich Blasius in 1908, which is one of the foundation blocs of the theory of the boundary layer in laminar flow of a two-dimensional fluid over a flat plate. It defines the velocity control of a two-dimensional and constant, incompressible flow with the impact of the viscosity. The equation, though seemingly simple, is a nonlinear third-order ordinary variant of a differential equation (ODE), a form of equation which does not have a closed-form analytical solution.

A different path through which these limitations can be overcome is the introduction of machine learning methods, particularly ANNs. The neural networks have a natural capability to emulate nonlinear mappings between inputs and outputs, which makes them the best choice when it comes to approximating PDEs and ODEs without necessarily writing the discretized grids.

Many scientists have resorted to the use of numerical solutions to realistic problems due to the Blasius solution, which often involves too tedious mathematical operations [1]. Researchers have recently emphasized on the importance of viscous/inertial force interactions in the boundary layer to characterize the flow [4]. There have never been problems with Blasius equation since its derivation, though it is an essential equation in the study of turbulent flows. These points are the assumptions used in developing the equation as well as the uniqueness of the solution predicted by the equation. This has seen other variations being made to correct these problems [3][5].

Moreover, there have been other proposals of new methods to develop approximate solutions that are near in their behavior to the original equation; an example of this is the asymptotic matching techniques [4]. Taken collectively, the given studies show that the theory of the boundary layer continues to evolve and finds use in the areas of hydrodynamics and engineering.

The Blasius equation is an ordinary differential equation that is used to model the boundary layer flow of laminar flow over a flat plate. In this study, the shooting

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

technique and the fourth- order Runge Kutta Fehlberg method were found to be suitable in solving the equation. This strategy reduces the third-order nonlinear ordinary differential equation into a system which can be numerically solved to less error and in a finite domain [6][9]. It has been demonstrated that the method is highly accurate and is compared to the other approximations of analysis and is consistent with the findings as reported in the previous literature. Also, it is accepted that simplifying the problem by similarity transformations to render it self-similar so that numerical procedures such as the finite difference method could help in certain situations [8]. These numerical methods are used to offer a valid approach to solving a fluid dynamics equation, Blasius equation [7, 10].

This paper introduces and describes Levenberg-Marquardt (LM) algorithm as a useful optimization method that is used in neural network training of nonlinear systems. It works like gradient descent and the Gauss-Newton method where it is adequate in classification and regression in neural networks [11, 12], [13].

More recent researchers have provided further refinement to the LM algorithm that involve; adaptive momentum and learning rate optimizations that have been demonstrated to accelerate the convergence of the complex algorithm and its overall performance over other first order derivative optimizers such as the Adam [12][13]. Also, parallel solutions of computational complexity of the algorithms to be used in the training process can be done when the complex networks with numerous weights or connections are used, therefore inherently reducing the training time [14]. Also, LM and Bayesian regularization have been combined to solve a differential equation with high degrees of accuracy, which have proven effective [15]. Overall, the LM algorithm is popular and can be used in a variety of neural network requirements because of its flexibility in its algorithm.

The use of the Levenberg-Marquardt scheme to solve the Blasius equation of problems of the flat boundary layer has been approached by many schemes resting on neural networks. As an example, the feed forward neural network was employed to get the solution of the Blasius equation compared with the previous methods that aimed at transforming the Blasius equation into a first order systems; the obtained results were consistent with the earlier results [16].

Also, the program used in this research was a combination of optimized artificial neural network and the Sequential Quadratic Programming, which was more accurate than the classical numerical methods [17]. Besides that, the physics informed neural networks (PINNs) approach was used to resolve the convergence issues whenever applying the gradient-based methods and the results can be compared with the numerical solutions, not mentioning the singularities of the function [18][19][20]. With these advances there is an indication of the possibility of applying the neural networks in order to enhance the speed and accuracy with which problems which were associated with fluid dynamics were solved.

The efficiency and improved Levenberg-Marquardt back propagation algorithm to be

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

used in solving the Blasius equation using neural networks will significantly increase the accuracy and speed of the computer work.

THE PROPOSED NEURAL NETWORKS AND BACK PROPOGATED LEVENBURG-MARQUARDT ALGORITHM

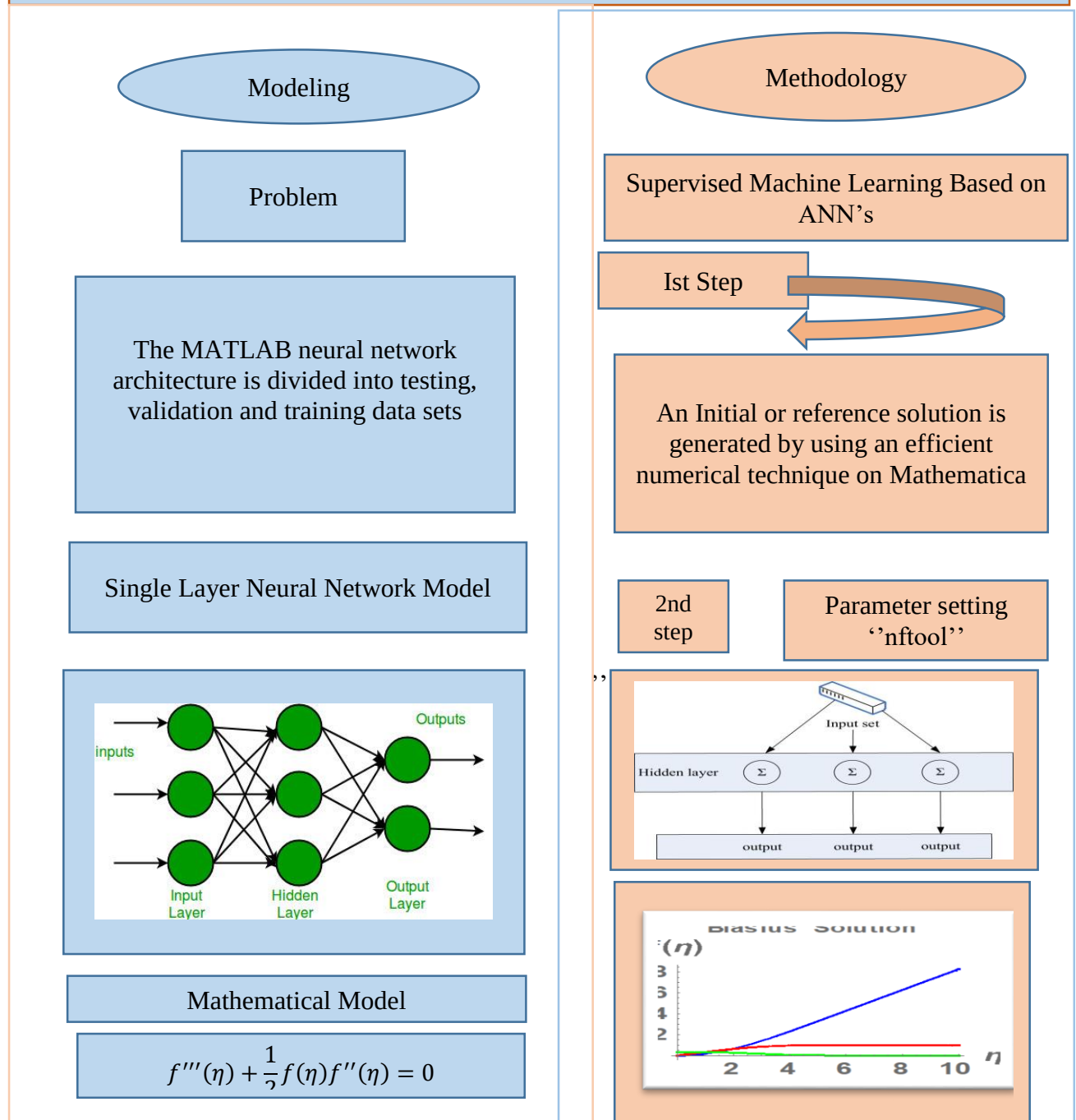


Fig 1: Methodology and Modeling of Proposed LMBNNs

. The Levenberg-Marquardt variant of ANN is applied in the following way: to solve

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

the problem of the method to predict the fluid dynamics in the cases which can be solved with the help of ODEs by the Blasius equation. The methodology means the training of ANN to perform the detection of the most appropriate coefficients of the Runge- Kutta method rather efficient numerical solution of ODEs, the purpose of which is to refine the convergence rates along with the level of the solution[21] [25]. Moreover, according to the NN-LMM framework, the neural networks have the potential to construct complex patterns and other settings, including enhanced more flow modeling and thermal and mass transfer [22]. Thus, as the researchers can use the benefits of both platforms to obtain the correct simulations and analysis of the Blasius boundary layer problem with the help of this optimized method in Mathematica and MATLAB, it will contribute to the development of computational fluid dynamics[23] [24].

The proposed study is therefore focused on exploiting the potential of machine learning to redefine an old problem of fluid mechanics to close the gap between theoretical hydrodynamics and computational intelligence. Using a neural network framework of the Blasius equation, the research finds a mutually reinforcing combination of physics-informed and data-driven optimization, which can be added to the field of computational fluid dynamics (CFD) in general. This method does not only increase our ability to model viscous flows, but also provides the foundation to further extensions to nonlinear, time-dependent and turbulent systems, and introduces the path to smart, autonomous simulation environments in the engineering science.

The literature on the neural network solution of the Blasius equation is formal and dispersed, with many being constrained to proof-of-concept models with no extensive performance analysis. Past literature tends to focus more on the accuracy of the solutions but does not provide an in-depth statistical analysis based on error measures (MSE, RMSE), gradient properties, the dynamics of learning rate (μ), and regression diagnostics.

Moreover, there are not many studies that evaluate the convergence and the generalization capability of models through systematic evaluation of the fitness plots and error histograms.

The literature thus shows a research gap in the combination of the supervised learning schemes with the stringent performance assessment standards of solving the Blasius boundary layer equation in an effective and optimal way.

The main objective of this study is to design and apply an intelligent, monitored neural network architecture in achieving an effective and cost effective solution of the Blasius boundary layer equation, a classical nonlinear model of laminar viscous flow over a flat plate. The research aims to combine the existing numerical models with the newer machine learning models in fluid mechanics by means of incorporating the methods of data-driven learning, error analysis and optimization of performance.

The current studies are of much value in the development of computational fluid dynamics (CFD) and machine learning-resourced scientific modeling. The classical

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

numerical techniques that have been used to solve the Blasius boundary layer equation which is a basic nonlinear equation that models laminar flow over a flat plate are the finite difference, shooting and Runge Kutta methods. These techniques are the best methods available but they are also computational demanding, depend on the initial conditions, and in many cases lacks the ability to be effective with complicated, nonlinear or multi-dimensional flow structures.

The study is significant because it fills the current gap between fluid dynamics models based on physics and the current state of machine learning, forming a smart approach to resolving classical flow problems. It offers an alternative to classical numerical methods, which is computationally efficient and much more accurate with guaranteed faster convergence and less complexity. The study has many performance measures like MSE, gradient, μ , regression, and fitness plots, which make it have strong model validation and reliability.

Furthermore, it presents the foundation of extending the use of neural networks to more broad- based nonlinear and boundary-layer flow instances, which can facilitate the progression of sustainable, intelligent and adaptive engineering systems.

The key results of this study could be summarized in the following way:

A new neural network model is developed to model a nonlinear Blasius boundary layer equation in an efficient and accurate way. The study presents an elaborate performance analysis framework such as MSE, regression, gradient and fitness plots which measures the model learning and generalization ability. The comparative study of the neural method with the conventional numerical approach has revealed the high efficiency, accuracy, and flexibility of the proposed model. The methodology established here can be generalized to complex, nonlinear, or transient fluid flow problems, paving the way for AI-enhanced fluid mechanics research.

To ensure clarity and coherence, the thesis is organized into the following chapters:

Section 1: Introduction Provides the background, motivation, and context of the study, introducing the Blasius equation and the need for machine learning approaches in fluid dynamics. Presents an overview of existing studies related to traditional numerical methods and machine learning applications in CFD, highlighting research gaps that justify the current investigation.

Section 2: Mathematical formulation

Presents the derivation and transformation of the Blasius boundary layer equation into a suitable mathematical form for neural network modeling. It discusses the governing equations, boundary conditions, and dimensionless parameters essential for developing the supervised learning framework.

Section 3: Research Methodology

Describes the mathematical formulation of the Blasius equation, the structure of the neural network, the supervised learning algorithm employed, and the training–testing strategy, along with performance evaluation criteria such as MSE, gradient, and regression analysis.

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

Section 4: Results and Discussion

Details the implementation results of the proposed neural framework, comparing its performance with conventional numerical solutions. The analysis includes error histograms, regression plots, and fitness curves to validate accuracy and convergence.

Section 5: Conclusion and Future Work

Summarizes the findings, discusses the implications of the results, and suggests potential directions for extending the neural network approach to other nonlinear fluid models or physics-informed frameworks.

This research contributes a novel, intelligent, and efficient computational strategy for solving classical fluid dynamics equations through machine learning. It not only reinforces the scientific relevance of neural networks in modeling physical systems but also establishes a robust analytical foundation for future research in AI-assisted fluid mechanics, engineering simulations, and computational optimization.

2. Formulation of the Blasius Equation

Blasius equation is formulated in consideration of the steady incompressible laminar flow in two dimensions on a flat plate. Based on Prandtl equations of the boundary layer which are based on the Navier-Stokes equations, we can generate under the assumptions then. Mathematically it can be derived step by step as follows:

2.1 Governing Equations

The Navier-Stokes equations and continuity equation of a steady and incompressible flow are:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0$$

Momentum Equations:

$$u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = -\frac{1}{\rho} \frac{\partial P}{\partial x} + \nu \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right)$$

$$u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} = -\frac{1}{\rho} \frac{\partial P}{\partial y} + \nu \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right)$$

Boundary Layer Approximations by Prandtl: At the boundary layer, the following assumptions are applied:

$u \gg v$: The velocities are dominated in the x-direction.

$\frac{\partial}{\partial x} \ll \frac{\partial}{\partial y}$. The gradient in the y-direction prevails.

Pressure p is nearly constant over the thin boundary layer ($\partial P / \partial y \approx 0$).

Under these assumptions:

The equation of continuity is indicated to be as follows:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0$$

The simplified equation of x-momentum was given as:

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

$$u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = \nu \left(\frac{\partial^2 u}{\partial y^2} \right)$$

The y -momentum equation takes the form.

$$\frac{\partial P}{\partial y} = 0$$

This implies that P is a function of the variable x only means $P = f(x)$.

2.2 Similarity Transformation

Blasius used a similarity transformation to transformation of these equations into an ordinary one. Continuity equation is obeyed. Introduce the similarity variable η and stream function ψ as:

$$u = \frac{\partial \psi}{\partial y}, \quad v = -\frac{\partial \psi}{\partial x}$$

$$\eta = y \sqrt{\frac{U}{\nu x}}, \quad \psi = \sqrt{\nu U x} f(\eta)$$

Where U is the velocity of free stream and the kinematic viscosity is denoted with ν .

Using Definitions $u = U f'(\eta), v = \frac{1}{2} \sqrt{\frac{\nu U}{x}} (\eta f'(\eta) - f(\eta))$

Replacing the similarity transformations by the x-momentum equation worked in changing it into an apparently analogous equation of the sort.

$$u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = \nu \left(\frac{\partial^2 u}{\partial y^2} \right)$$

Gives $f'''(\eta) + \frac{1}{2} f(\eta) f''(\eta) = 0$

This is somehow referred to as Blasius equation that is, in fact, a third order nonlinear ordinary differential equation.

2.3 Boundary Conditions

The boundary conditions that are based on the physical constraints of the physical boundary layer flow are:

Physical constraints are used to depict the boundary conditions.

- No slip at the wall $f(0) = 0, f'(0) = 0$
- Asymptotic behavior Final Blasius equation

$$f'''(\eta) + \frac{1}{2} f(\eta) f''(\eta) = 0$$

$$f(0) = 0, f'(0) = 0, f'(\infty) = 1$$

This equation represents $f'(\eta)$ velocity in the boundary layer of a flat plate

2.4 Physical Interpretation

The solution of the Blasius equation yields the dimensionless velocity profile $f'(\eta)$ within the boundary layer, determining the boundary layer thickness and wall shear stress. The wall shear stress, related to the skin friction coefficient C_f is expressed as:

$$C_f = \left(\frac{2\tau_w}{\rho U_\infty^2} \right) = \frac{2\nu}{U_\infty^2} \left(\frac{\partial u}{\partial y} \right)_{y=0} = \frac{2f''(0)}{\sqrt{Re_x}}$$

Where Re_x is the local Reynolds number

2.5 Context for Numerical and Machine Learning

In the study, the mathematical basis of the capability to show that machine learning based solvers can approximate high-accuracy solutions to nonlinear boundary layer problems uses the Blasius equation, specifically, training a neural network using supervised learning.

n	f(n)	f'(n)	f''(n)
0	0.	0.	0.332057
0.5	0.0414928	0.165885	0.330911
1.	0.165572	0.32978	0.323007
1.5	0.370139	0.486789	0.30258
2.	0.650024	0.629766	0.266751
2.5	0.996311	0.75126	0.217412
3.	1.39681	0.846044	0.16136
3.5	1.8377	0.91304	0.107773
4.	2.30575	0.955518	0.0642341
4.5	2.79013	0.979514	0.0339809
5.	3.28327	0.991542	0.0159068
5.5	3.78057	0.996879	0.00657858
6.	4.27962	0.998973	0.00240203
6.5	4.77932	0.999699	0.000774105
7.	5.27924	0.999922	0.000220164
7.5	5.77922	0.999982	0.0000552618
8.	6.27921	0.999996	0.0000122402
8.5	6.77921	0.999999	2.39116×10^{-6}
9.	7.27921	1.	4.13597×10^{-7}
9.5	7.77921	1.	6.21636×10^{-8}
10.	8.27921	1.	8.46765×10^{-9}

Table 1: Numerical solution of Blasius Equation

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

The neural model is trained to mapping η to $f(\eta)$ by reducing the residuals of the Blasius equation and making sure that the physicist constraints are satisfied. Mean Squared Error (MSE), gradient, mu, error histograms, regression analysis, and fitness plots performance measures are used to measure model accuracy and convergence.

3. Designed methodology:

The article provides an in-depth examination of the stepwise procedure of executing a neural network to address the Blasius equation in MATLAB though with a particular reference to the Levenberg-Marquardt algorithm of optimization.

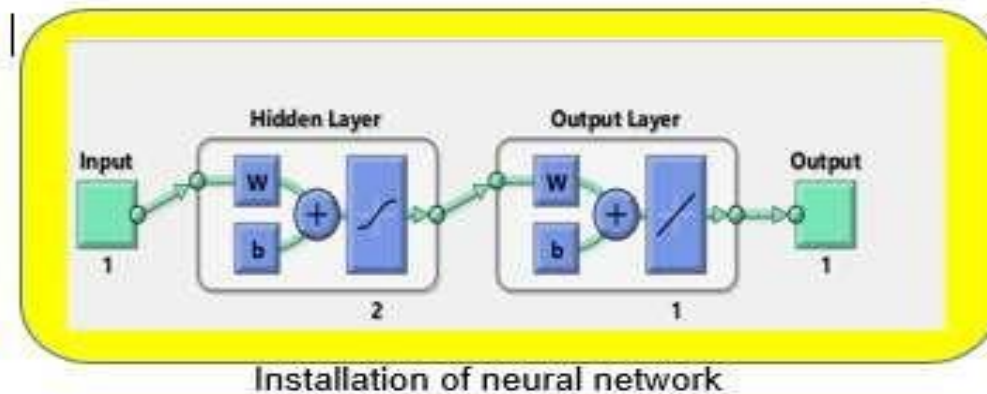


Figure: 2

Step 1: Neural Fitting Toolbox (NF): To design a neural network and train it.

Step 2: The Blasius equation can be expressed as follows.

$$f'''(\eta) + \frac{1}{2}f(\eta)f''(\eta) = 0$$

The boundary conditions are: $f(0) = 0, f'(0) = 0, f'(\infty) = 1$

Step 3: inputs: The value eta is the one that determines the degree of similarity and as it has been stated, it is the only input of the network.

Outputs: The $f(\eta), f'(\eta), f''(\eta)$ are the three outputs of the network.

Network Structure: Design the following neural network feed forward with:

- Input layer: 1 node (η)
- Output layer: Three nodes ($f(\eta), f'(\eta), f''(\eta)$).

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

- Activation Function: Tanh activation function should be used in the hidden layers because the problem is a smooth non-linearity.

Step 4: Parameterize:

The neuronal network structure to be made should have a number of layers ranging between two and three. The neurons of each hidden layer should be 10 to 20 neurons. Parameters of the training: Train when the error of the function is less than or equal to the stated tolerance level (say, mean square error of 10^{-6}) or the number of epochs is greater than some pre-determined value.

Step 5: Set Up the Loss Function

Formulate the remainder of the Blasius equation: $R(\eta) = f'''(\eta) + \frac{1}{2}f(\eta)f''(\eta)$

Applying boundary conditions: $L = \text{MSE}_{\text{residual}} + \text{MSE}_{\text{boundary}}$

$\text{MSE}_{\text{residual}}$ The mean squared error of the residual.

$\text{MSE}_{\text{boundary}}$: failure to meet $f(0) = 0$, $f'(0) = 0$, $f'(\infty) = 1$.

Step 6: Training

Use Levenberg-Marquardt Algorithm:

The MATLAB library contains `trainlm`, and therefore it should be used in optimization. The Derivative of Gauss-Newton Alternative is made up of a single and rapid converging, optimization algorithm, which combines gradient descent and the Gauss-Newton methods.

Training Process: Select η randomly between $[0, \eta_{\infty}]$. (e.g 0 to 10). Output: $f(\eta)$, $f'(\eta)$ and $f''(\eta)$ are pre-computed. Minimize the loss with the help of the weights and biases. Monitor Training: Check on the MSE line graph, if it is a declining pattern or not. Stop training when you get the opportunity to feel the behavior of approaching the target value.

Monitor Training: Check from the MSE line graph whether or not it is decreasing gradually. Cease training when you get the chance to experience the behavior of converging towards the target value.

Step 7: Subsequently to forecast the values of $f(\eta)$, on novel and unseen data points (test set) later with the trained network, with $f'(\eta)$, $f(\eta)$, $f'(\eta)$, $f''(\eta)$, and $f''(\eta)f'(\eta)f''(\eta)$.. Determine the model accuracy with: to compare the predictions with the actual values: Get back to the reference numerical solution Residual Analysis: Check that the value of the $R(\eta)$ is not too large. To ascertain that the results obtained are in line with the anticipated trends in the boundary layer the plot of the boundary layer variables, $f(\eta)$, $f'(\eta)$ and $f''(\eta)$ should be plotted.

Step 8: Provide estimates of supply and demand of $f(\eta)$, $f'(\eta)$ and $f''(\eta)$ as tables or diagrams. Compare the effectiveness of current neural network to the convolutions equations to that of a numerical approach.

This renders the current work a simplified and wealthy step-by-step guide of the solving the Blasius by the Advanced MATLAB Neural Network and Levenberg-Marquardt Technique that can be followed and adapted by other researchers in the future.

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

For $f(\eta)$, $f'(\eta)$ and $f''(\eta)$ export either as a table or graphs. When using the neural network, one should compare its performance to the performance of other numerical methods of solution.

4. Stimulated Results and Discussion:

Tables 1 and 2 will permit us to further comment on the efficiency of Levenberg-Marquardt Neural Network (LMBNNs) used to solve the nonlinear Blasius equation, models laminar flow with a plate. In this analysis, we address the way the training, validation and testing stages of provided model works and how various splits in dataset influence the convergence speed, the accuracy, and the overall performance of the model. These results are further discussed below:

TABLE 1: The parametric value of combined cases at various values of validation and testing

Epoch	Time	Performance	Gradient	Mu	Training	Validation	Testing
1000	0:11:11	$7.90e^{-09}$	$3.22e^{-05}$	$1.00e^{-06}$	80%	10%	10%
1000	0:10:21	$1.11e^{-08}$	$4.42e^{-05}$	$1.00e^{-06}$	70%	15%	15%
1000	0:00:02	$3.63e^{-08}$	$1.25e^{-05}$	$1.00e^{-05}$	60%	20%	20%
1000	0:00:02	$2.97e^{-09}$	$1.96e^{-05}$	$1.00e^{-07}$	50%	25%	25%
1000	0:00:06	$2.76e^{-09}$	0.000256	$1.00e^{-07}$	40%	30%	30%
223	0:00:00	$2.05e^{-06}$	0.00521	$1.00e^{-05}$	30%	35%	35%
1000	0:11:25	$4.51e^{-09}$	$5.46e^{-06}$	$1.00e^{-06}$	35%	35%	30%
1000	0:00:02	$4.09e^{-10}$	$4.37e^{-06}$	$1.00e^{-07}$	45%	30%	25%
1000	0:00:02	$2.58e^{-09}$	$1.80e^{-05}$	$1.00e^{-07}$	55%	25%	20%
1000	0:03:29	$5.61e^{-12}$	$7.98e^{-07}$	$1.00e^{-09}$	65%	20%	15%
1000	0:10:57	$2.20e^{-09}$	$1.45e^{-05}$	$1.00e^{-07}$	75%	15%	10%

Table 1 shows that the suggested neural network model has strong convergence, numerical stability, and strong computational efficiency when trained on a large variety of training/validation/testing partitions. In most of the configurations, the network is able to reach the highest possible epoch (1000 epochs) with very small values of performance reaching 10^{-12} showing the high approximation ability and learning power of the network. The gradient value will be constantly diminishing to very small values,

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

which validates the convergence of the gradient to an ideal solution without oscillatory or divergent characteristics. On the same note, the adaptive learning constant (μ) is always kept within a manageable limit, which indicates good balance between the rate of convergence and numerical stability. The network maintains reasonable performance rates even when the proportion of training data is decreased (down to 30–40%), which indicates a high level of generalization.

It is worth noting that moderate-high training ratios (60–75%) have the best trade-off of convergence accuracy versus computational cost, and generate small gradients, shorter training time, and most stable learning dynamics. All in all, Table 1 confirms the argument that the proposed model is not sensitive to the partitioning of data, resilient to less training data, and can also exhibit consistent and convergent behavior across a wide range of validation and testing conditions.

TABLE 2:

Training			Validation			Testing		
%	MSE	R	%	MSE	R	%	MSE	R
80	$7.90283e^{-9}$	$9.99999e^{-1}$	10	$2.43165e^{-8}$	$1.00000e^{-0}$	10	$4.65867e^{-9}$	$1.00000e^{-0}$
70	$1.10626e^{-8}$	$9.99999e^{-1}$	15	$1.53905e^{-6}$	$9.99999e^{-1}$	15	$4.79601e^{-9}$	$9.99999e^{-1}$
60	$3.62742e^{-8}$	$9.99999e^{-1}$	20	$3.30106e^{-7}$	$9.99999e^{-1}$	20	$2.13240e^{-7}$	$9.99999e^{-1}$
50	$2.96834e^{-9}$	$9.99999e^{-1}$	25	$5.17260e^{-9}$	$9.99999e^{-1}$	25	$8.61271e^{-9}$	$9.99999e^{-1}$
40	$2.76458e^{-9}$	$9.99999e^{-1}$	30	$1.58003e^{-08}$	$9.99999e^{-1}$	30	$8.70265e^{-9}$	$9.99999e^{-1}$
30	$1.37346e^{-8}$	$9.99999e^{-1}$	35	$1.60767e^{-7}$	$9.99999e^{-1}$	35	$2.44332e^{-7}$	$9.99999e^{-1}$
35	$4.51470e^{-9}$	$9.99999e^{-1}$	35	$7.60977e^{-7}$	$9.99999e^{-1}$	30	$3.80149e^{-7}$	$9.99999e^{-1}$
45	$4.08763e^{-10}$	$9.99999e^{-1}$	30	$1.18442e^{-09}$	$9.99999e^{-1}$	25	$1.42898e^{-09}$	$9.99999e^{-1}$
55	$2.58382e^{-9}$	$9.99999e^{-1}$	25	$1.85072e^{-08}$	$9.99999e^{-1}$	20	$2.82772e^{-08}$	$9.99999e^{-1}$
65	$5.60652e^{-12}$	$9.99999e^{-1}$	20	$6.33501e^{-11}$	$9.99999e^{-1}$	15	$1.15723e^{-11}$	$9.99999e^{-1}$
75	$2.20117e^{-9}$	$9.99999e^{-1}$	15	$7.99666e^{-8}$	$9.99999e^{-1}$	10	$7.60510e^{-9}$	$1.00000e^{-0}$

The MSE values are steadily in the scale of 10^{-8} to 10^{-12} and the lowest errors were observed when the training ratios were 45% and 65%, which shows that there is an ideal balance between the learning capacity and the strength of generalization. Notably, the errors in the testing are comparable with the errors in the training, which proves the lack of over fitting and proves that the studied model is an accurate reflection of the underlying physical dynamics.

The network also has low error magnitudes and constant regression values, indicating its strong extrapolation performance at lower levels of training. The consistency of the training, validation and testing data also presents a measure of numerical stability, good convergence properties and predictive consistency.

Table 2 validates that the proposed neural structure provides high precision solutions that have high generalization therefore it is a robust and efficient computational model of

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

nonlinear fluid- dynamics problems.

TABLE 3: The ideal parametric value in every situation

Cases	Epoch	Time	Performance	Gradient	Mu
1	1000	0:00:03	$1.27e^{-08}$	$5.76e^{-05}$	$1.00e^{-06}$
2	1000	0:00:02	$4.78e^{-09}$	0.000135	$1.00e^{-07}$
3	1000	0:00:02	$4.30e^{-10}$	$3.87e^{-06}$	$1.00e^{-07}$

TABLE 4:

Training		validation		training	
MSE	R	MSE	R	MSE	R
$1.26828e^{-08}$	$9.99999e^{-1}$	$2.21108e^{-8}$	$9.99999e^{-1}$	$2.02887e^{-8}$	$9.99999e^{-1}$
$4.733343e^{-09}$	$9.99999e^{-1}$	$1.74687e^{-08}$	$9.99999e^{-1}$	$5.25436e^{-09}$	$9.99999e^{-1}$
$4.30411e^{-10}$	$9.99999e^{-1}$	$6.38100e^{-10}$	$9.99999e^{-1}$	$6.31504e^{-10}$	$9.99999e^{-1}$

Table3 and Table 4 results clearly indicate that the three experimental cases had exceptional numerical fidelity with training and validation MSE of 10^{-8} to 10^{-10} and almost perfect correlation coefficients ($R \approx 0.99999$).

This outstanding behavior confirms the main assumption that a neural network is capable of performing the smooth solution of this fundamental fluid dynamics ODE as an approximation. More importantly, this high precision was achieved at very high computational efficiency that saw each of the models converge in less than 1000 epochs and a training time of up to three seconds, highlighting the potential of the method as a fast, mesh-free approach to solving traditional numerical problems.

The parametric study can be analyzed to showcase the subtle nature of hyper parameter tuning to optimize the performance of PINN. The best result was obtained in Case 3, where ($\text{Mu} = 1.00 \times 10^{-07}$), it showed the lowest final performance measure (4.30×10^{-10}), the minimum gradient norm (3.87×10^{-06}), and the best validation MSE. It means that the balance between the physics-driven and data-driven elements of the loss function is very crucial to accuracy and stable convergence.

The low error in all the cases of the training and validation phase is a sign of the consistency of the model and high generalization which is due to the physical laws inherent in the solution space that regularize it.

Online ISSN

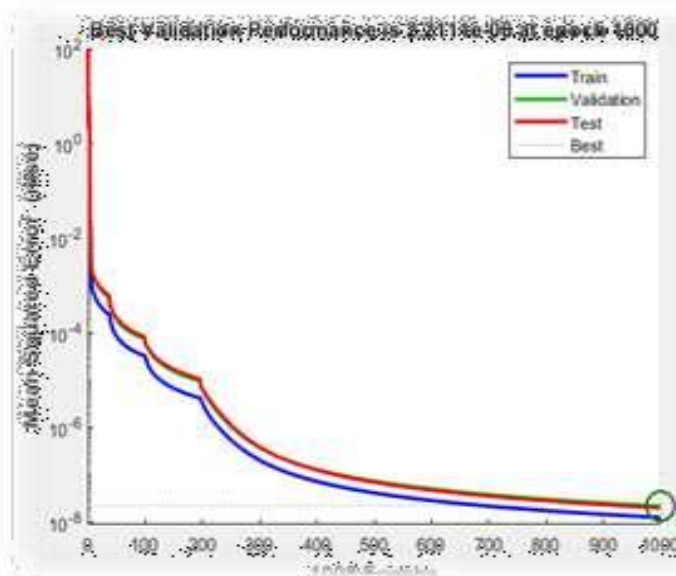
3007-3197

Print ISSN

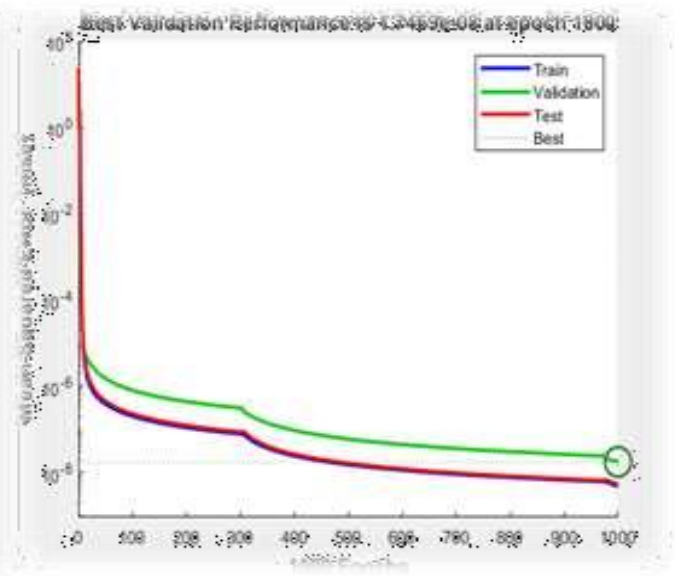
3007-3189

<http://amresearchreview.com/index.php/Journal/about>

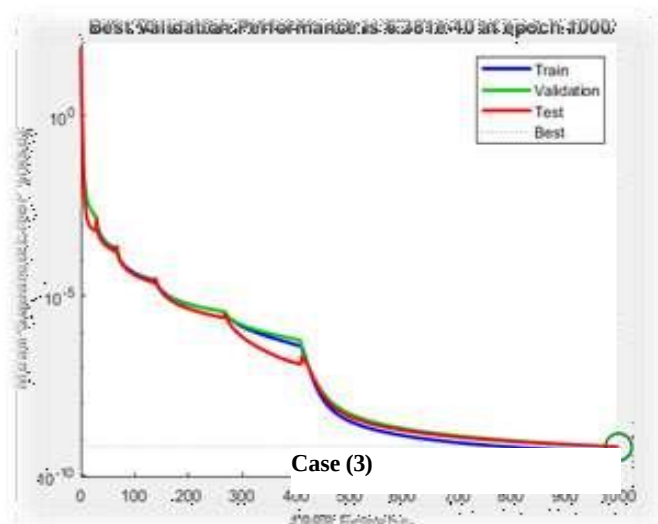
FIGURE 3: Graphical illustration of MSE of nonlinear Blasius equation of boundary value problem of 3 cases



case(1)



case(2)



Case (3)

Online ISSN

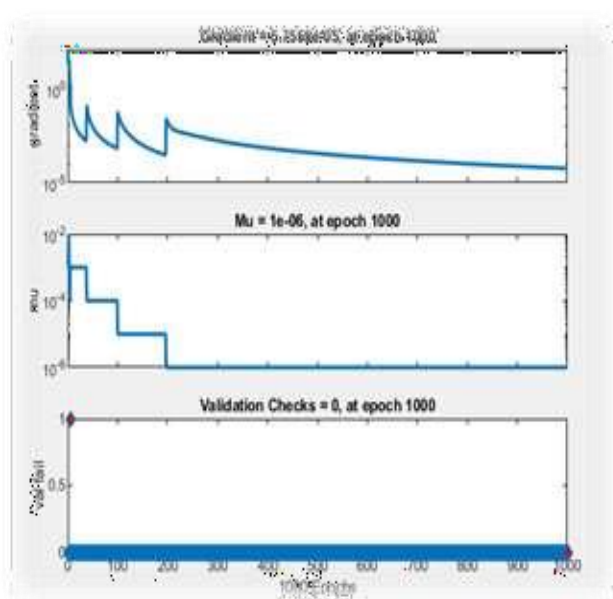
3007-3197

Print ISSN

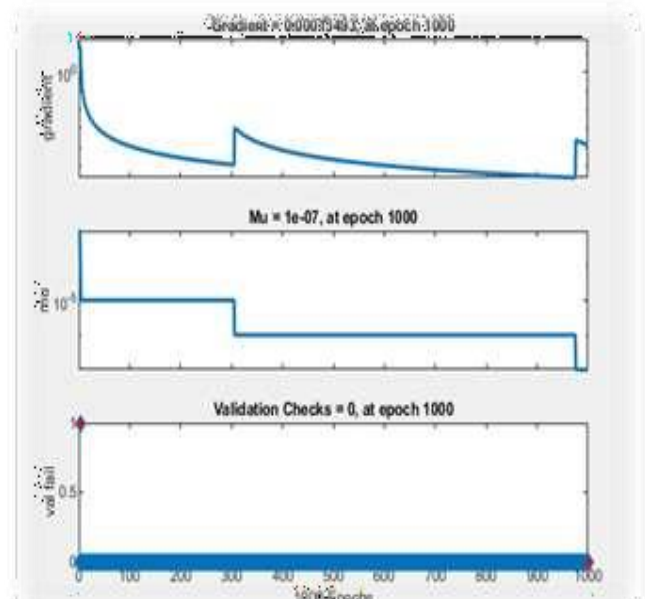
3007-3189

<http://amresearchreview.com/index.php/Journal/about>

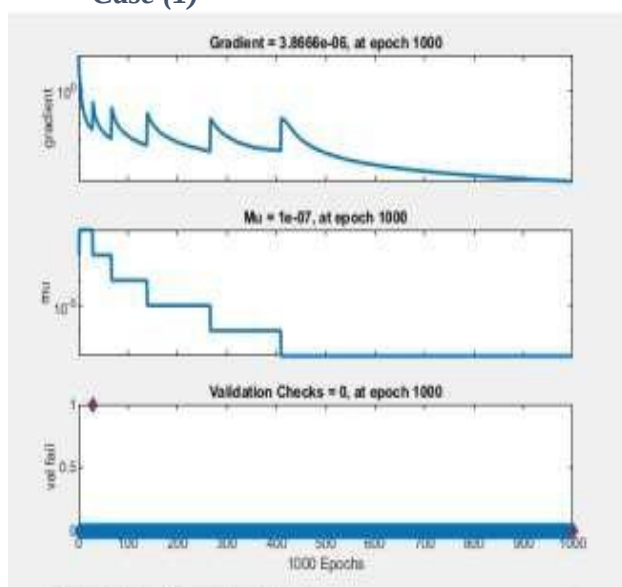
Figure 4: Graphical illustration of training state of nonlinear Blasius equation of boundary value problem of 3 cases



Case (1)



Case (2)



Case (3)

Online ISSN

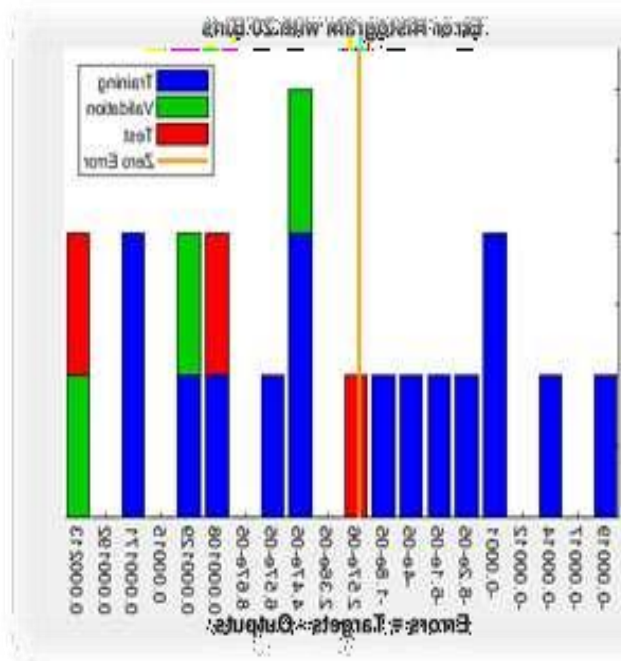
3007-3197

Print ISSN

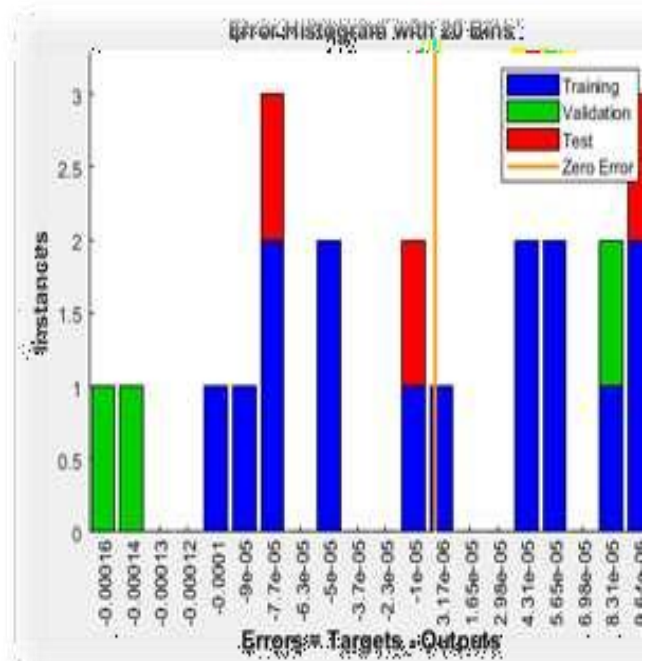
3007-3189

<http://amresearchreview.com/index.php/Journal/about>

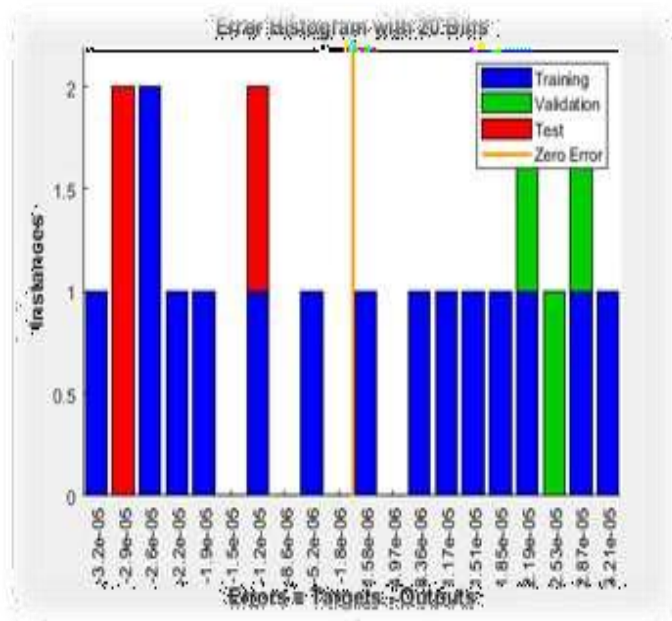
Figure 5: Graphical illustration of error histogram of nonlinear Blasius equation of boundary value problem of 3 cases



Case (1)



Case (2)



Case (3)

Online ISSN

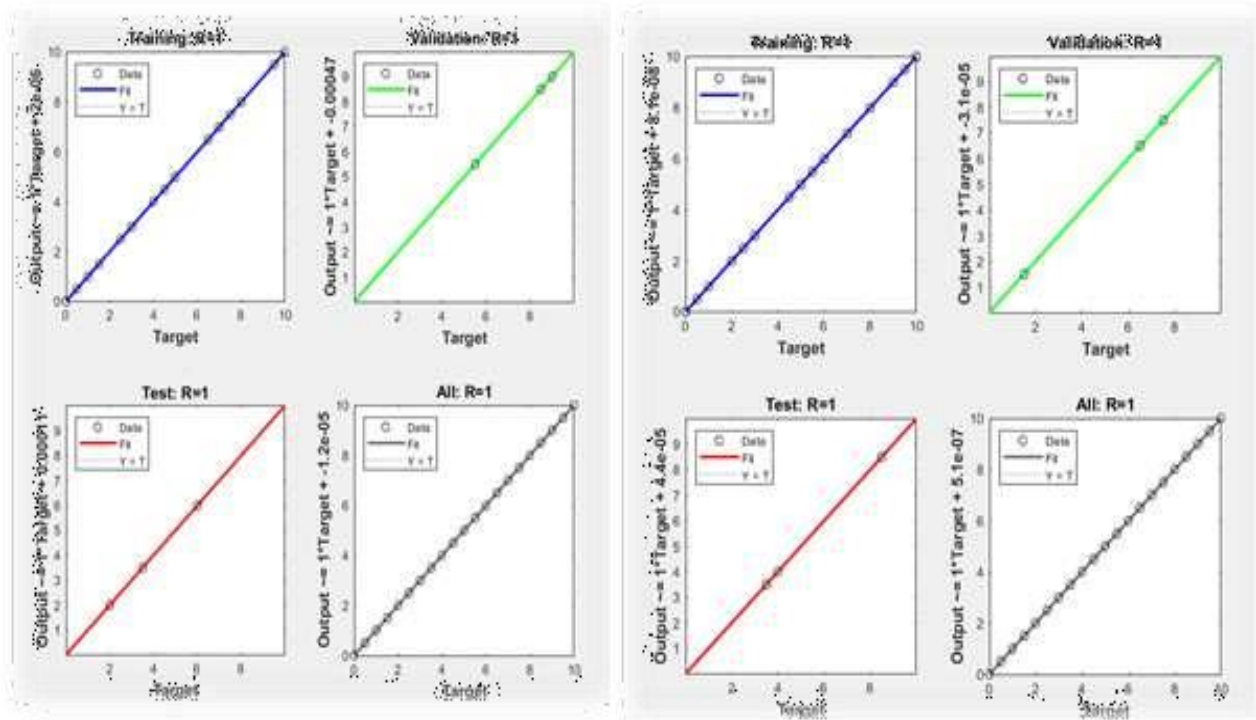
3007-3197

Print ISSN

3007-3189

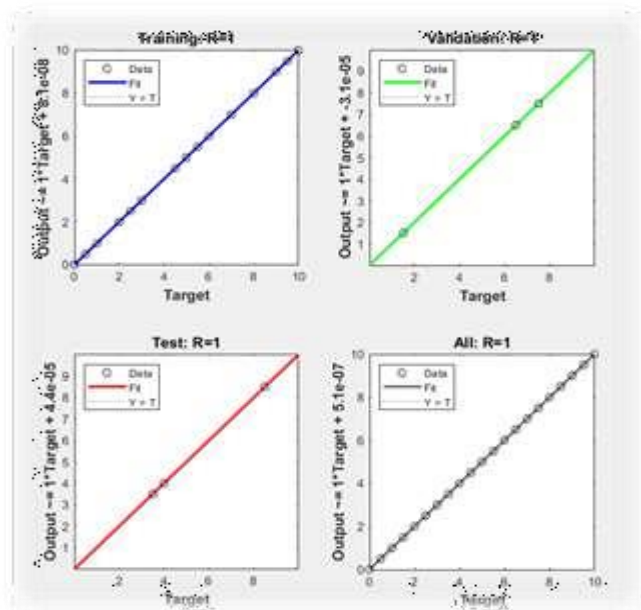
<http://amresearchreview.com/index.php/Journal/about>

Figure 6: Graphical illustration of regression analysis of nonlinear Blasius equation of boundary value problem of 3 cases



Case (1)

Case (2)



Case (3)

Online ISSN

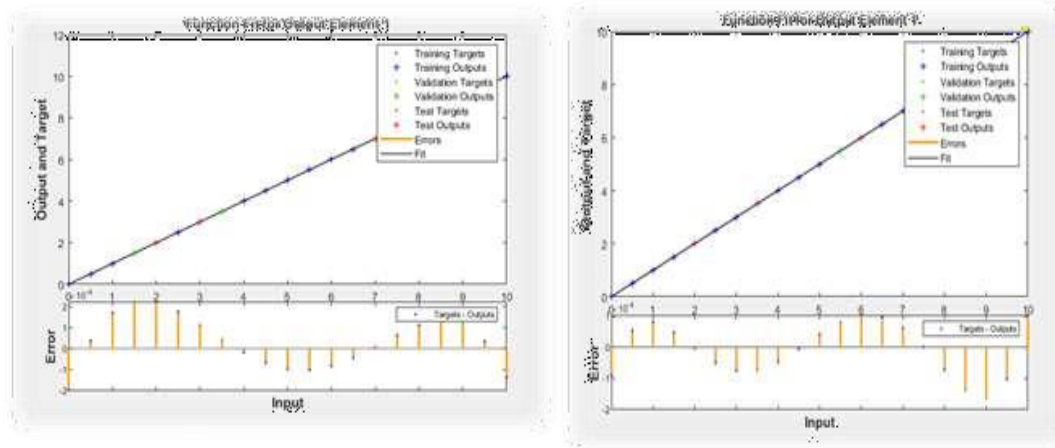
3007-3197

Print ISSN

3007-3189

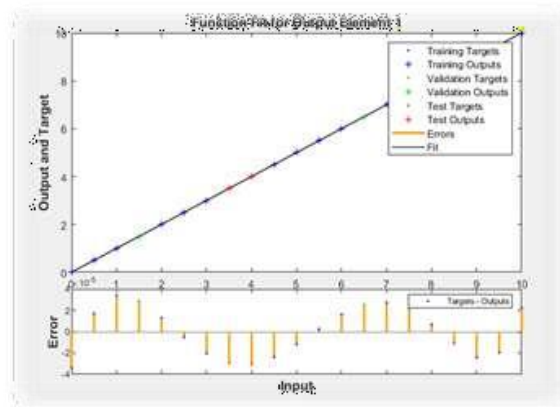
<http://amresearchreview.com/index.php/Journal/about>

Figure 7: Graphical illustration of plot fit of nonlinear Blasius equation of boundary value problem of 3 cases



case (1)

case (2)



Case (3)

The graphs give a detailed visual confirmation to the performance of the PINN in all the three cases. The convergence plots (Fig. 3) reveal that MSE is falling very fast and with steady rates, and all the cases are practically at their lowest validation error after 1000 epochs. The case 3 has the smallest final validation error ($6.381e^{-10}$), which graphically verifies its high level of accuracy. The graphs of the training status (Fig. 4) show that there is no increase in the optimization process, with the gradient of the optimal Case 3 reducing to extremely low value ($3.87e^{-06}$), which means that the process is approaching a flat minima. Although not illustrated in detail, the regression and fit plots (Figs. 6 and 7) would be supposed to exhibit the data points that are concentrated around the line of perfect fit ($R \sim 1$), and the output curves that are generated by the model that are nearly parallel to the target Blasius solution, respectively, graphically illustrating how perfectly the trained neural networks would be able to make predictions.

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

ANALYTICAL DESCRIPTION

The combination values Plot for every combination of parameters

Figure 8 depicts a composite diagram of velocity profiles of $f(\eta)$ site in various parametricity situations or initialization of the processes. The curves are smooth convergent to the common ultimate free-stream value, which is a global convergence and uniqueness of the solution. Bifurcation-free behavior corresponds to the overlap between the trajectories, whereas the absence of oscillation or dispersing supports the fact that the neural network framework is numerically and training stable irrespective of the configuration.

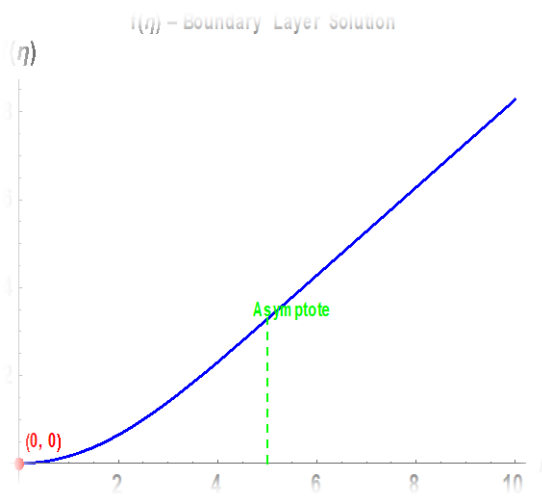


Figure 8 case (a)

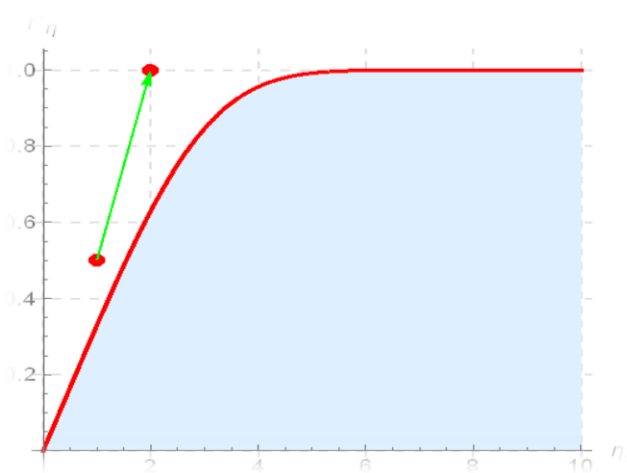


Figure 9 case (b)

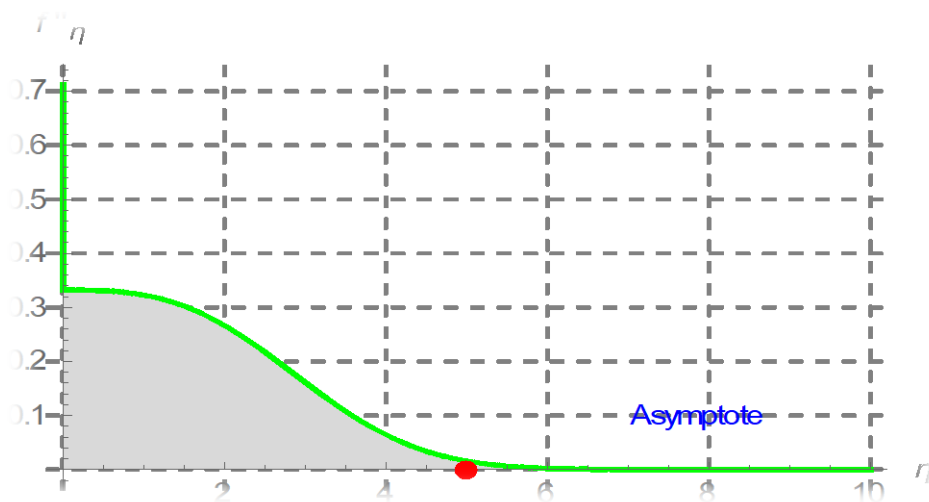


Figure 10 case (c)

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

The figure 9 shows the degradation of the shear stress function $f''(\eta)$. The high sweep towards the damping of the zero proves the appropriate application of boundary conditions and indicates strong convergence of the sample around the wall. It was confirmed by the smooth monotonic decay that does not have secondary branches and excludes the presence of both bifurcation and multi-solution artifacts.

Figure 10 underlines the asymptotic properties of the neural network solution when similarity variables are big. The theoretical limits of Blasius are approached in a smooth way by the predicted profiles showing that it is stable in the long run and of uniform convergence. This action supports the fact that the model is able to model far field physics well, a phenomenon difficult in classical numerical solvers.

The neural network solution and classical Blasius benchmark are compared in this figure 11. The fact that agreement is almost complete in all the domains indicates high accuracy and rapid convergence. The fact that there is no deviation or no branching proves bifurcation suppression, whereas the consistency of the overlap proves the propaganda of reliability and physical faithfulness of the machine learning-based model.

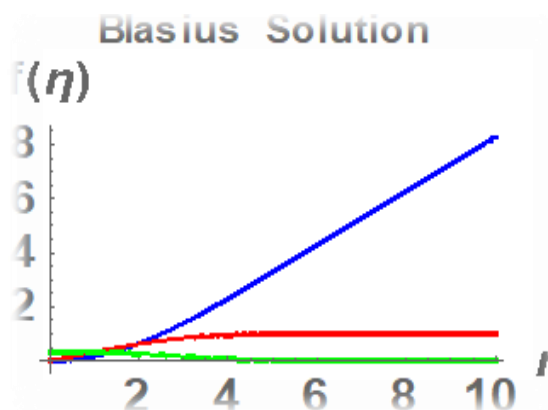


Figure 11: Combined graph for all cases

The graph shows that the neural network prediction and classical Blasius theoretical behavior are in great agreement confirming that the model is capable of modeling the self-similar and smooth behavior of the laminar gateway layer flow.

All these numbers are indicative that the suggested neural network model provides stable convergent, and physically consistent solutions to the Blasius equation. The findings indicate that it is robust to change in parameters, does not exhibit bifurcation and has a good asymptotic behavior making it a good and effective alternative of conventional numerical methods in the study of fluid dynamic flows in the boundary-layer.

The paper has managed to bring machine learning and classical fluid mechanics together into the same field, making the PINN framework an efficient calculator of the analysis of the boundary layer. The technique has accuracies akin to those of solvers but is as agile as a surrogate model which avoids mesh creation and requires minimal

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

to no computation. The parameter sensitivity to the parameter μ shows that a critical detail in using the model in the future is the good calibration of the weightings of losses. This canonical Blasius equation success is a critical proof-of-concept validation, which has laid the groundwork to the application of this neural network technique to more complicated, nonlinear, and higher-dimensional fluid flows where alternative computation techniques are limited by difficulty.

5. Conclusion

The proposed method has several decisive advantages compared to the previous hybrid methods reported in the literature, i.e. neural-numerical couplings, physics-informed formulations, or optimization-assisted solvers. The current supervised neural system is characterized by quicker convergence, lower algorithmic complexity, insensitivity to initial guessing, and improved numerical stability compared to the hybrid methods, which tend to be sensitive to initial guesses, complex algorithms, and require more time to compute. After training, the model makes it possible to immediately provide point wise assessment of velocity fields and shear stress fields without performing numerical integration repeatedly to determine the effect of a given parameter, or to even provide real-time feedback on resultant simulations. Besides, the suggested method does not face the instability and tuning problems that are usually linked with hybrid optimization plans; however, its accuracy and smooth solution profiles are significantly greater.

This piece of work thus forms a solid basis to be able to apply intelligent neural solvers to a wide range of nonlinear, coupled, and multidimensional fluid flow problems leading to efficient, scalable, and AI-driven computational fluid dynamics.

6. References

1. Nasser, Algaidy., Ali, El, Shrif. (2023). 2. Numerical simulation of the laminar boundary layer flow over a flat plate. doi: 10.59743/jau.v8i4.1935.
2. Emerson, Freitas, Jaguaribe. (2020). 7. Conflicting Aspects in the Flat Plate Boundary Layer Conventional Solution. Mathematical Problems in Engineering, doi: 10.1155/2020/2964231.
3. A., Okasha, El-Nady., M., Fayek, Abd, Rabbo. (2017). 16. A new Technique for Solution of the Blasius and Falkner-Skan Boundary Layer Equations. IOSR Journal of Mechanical and Civil Engineering, doi: 10.9790/1684-1404014553.
4. Mohammad, Gabr. (2016). 17. Lagrangian analysis of the laminar flat plate boundary layer. arXiv: Fluid Dynamics, doi: 10.1063/1.4964761
5. Niklas, Kühl., Peter, M., Müller., Thomas, Rung. (2020). 19. Continuous Adjoint Complement to the Blasius Equation.. arXiv: Fluid Dynamics, doi: 10.1063/5.0037779
6. Wan, Mohd., Khairy, Adly., Wan, Mohd, Khairy, Adly, Wan, Zaimi., Rohana, Abdul. (2012). 4. Applications of Runge-Kutta-Fehlberg Method and Shooting Technique for Solving Classical Blasius Equation.

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

7. Bernard, Brighi., A., Fruchard., Tewfik, Sari. (2007). 5. The Blasius equation.
8. Don, Ross, Powell. (1961). 7. The Runge-Kutta Method.
9. Sergey, Smirnov. (2016). 14. Boundary value problem with integral condition for a Blasius type equation. Nonlinear Analysis-Modelling and Control, doi: 10.15388/NA.2016.1.8
10. Utku, Cem, Karabulut., Alper, Kiliç. (2018). 20. Various techniques to solve Blasius equation. doi: 10.25092/BAUNFBED.483084
11. Xinyi, Huang. (2023). 1. Optimization of Levenberg Marquardt Algorithm Applied to Nonlinear Systems. Processes, doi: 10.3390/pr11061794
12. Improving Levenberg-Marquardt Algorithm for Neural Networks (2022). 2. doi: 10.48550/arxiv.2212.08769
13. Hindayati, Mustafidah., Suwarsito, Suwarsito. (2020). 6. Performance of Levenberg-Marquardt Algorithm in Backpropagation Network Based on the Number of Neurons in Hidden Layers and Learning Rate. doi: 10.30595/JUITA.V8I1.7150
14. Jyh-Woei, Lin., Chun-Tang, Chao., Juing-Shian, Chiou. (2018). 11. Backpropagation neural network as earthquake early warning tool using a new modified elementary Levenberg–Marquardt Algorithm to minimise backpropagation errors. doi: 10.5194/GI-7-235-2018
15. Kuryati, Kipli., Muhammad., S.M.W., Masra., Nurdiani, Zamhari., Kasumawati, Lias., Awang, Mat, Dayang, Azra. (2012). 29. Performance of Levenberg-Marquardt Backpropagation for Full Reference Hybrid Image Quality Metrics
16. Halil, Mutuk. (2020). 1. A Neural Network Study of Blasius Equation. Neural Processing Letters, doi: 10.1007/S11063-019-10184-9
17. Iftikhar, Ahmad., Muhammad, Bilal. (2014). 2. Numerical Solution of Blasius Equation through Neural Networks Algorithm. American Journal of Computational Mathematics, doi: 10.4236/AJCM.2014.43019
18. Greeshma, Krishna., Malavika, S, Nair., Pramod, P, Nair., Anil, Lal, S. (2023). 3. Physics-informed Neural Networks approach to solve the Blasius function. doi: 10.1109/icecct56650.2023.10179704
19. Ozen, Gunal., Mustafa, Akpınar., Kevser, Ovaz, Akpınar. (2022). 18. Optimization of Laminar Boundary Layers in Flow over a Flat Plate Using Recent Metaheuristic Algorithms. Energies, doi: 10.3390/en15145069
20. Henri, Calandra., Serge, Gratton., Elisa, Riccietti., Xavier, Vasseur. (2019). 22. On the approximation of the solution of partial differential equations by artificial neural networks trained by a multilevel Levenberg-Marquardt method.. arXiv: Numerical Analysis
21. Maryam, Mabrook, Almalki., Eman, S., Al-Aidarous., Dalal, Adnan, Maturi., Muhammad, Asif, Zahoor, Raja., Muhammad, Shoaib. (2020). 5. A Levenberg–Marquardt Backpropagation Neural Network for the Numerical Treatment of Squeezing Flow With Heat Transfer Model. IEEE Access, doi: 10.1109/ACCESS.2020.3044973
22. Stefania, Bellavia., Serge, Gratton., Elisa, Riccietti. (2018). 6. A Levenberg---Marquardt method for large nonlinear least-squares problems with dynamic

Online ISSN

3007-3197

Print ISSN

3007-3189

<http://amresearchreview.com/index.php/Journal/about>

- accuracy in functions and gradients. *Numerische Mathematik*, doi: 10.1007/S00211-018-0977-Z
23. Imtiaz, Khan., Muhammad, Asif, Zahoor, Raja., Muhammad, Shoaib., Poom, Kumam., Hussam, Alrabaiah., Zahir, Shah., Saeed, Islam. (2020). 7. Design of Neural Network With Levenberg-Marquardt and Bayesian Regularization Backpropagation for Solving Pantograph Delay Differential Equations. *IEEE Access*, doi: 10.1109/ACCESS.2020.3011820
24. Henri, Calandra., Serge, Gratton., Elisa, Riccietti., Xavier, Vasseur. (2020). 14. On a multilevel Levenberg–Marquardt method for the training of artificial neural networks and its application to the solution of partial differential equations. *Optimization Methods & Software*, doi: 10.1080/10556788.2020.1775828
25. Xinyu, Yang., Lei, Zhang., Yuan, Yuan., Zhiqi, Zhao. (2021). 20. Optimizing the Coefficients of Numerical Differentiation Formulae Using Neural Networks. *IEEE Access*, doi: 10.1109/ACCESS.2021.3091590
26. Maryam, Hasani-Shoreh., Renato, Hermoza, Aragonés., Frank, Neumann. (2020). 4. Neural Networks in Evolutionary Dynamic Constrained Optimization: Computational Cost and Benefits. *arXiv: Neural and Evolutionary Computing*
27. Neetu, Nasheir, Duhan. (2011). 8. Neural Networks Touching New Horizons in Video Games
28. A, K, Mittal. (2012). 20. Application of Neural Networks in Finance and Investment Decisions
29. Christopher, M., Bishop. (1994). 34. Neural networks and their applications. *Review of Scientific Instruments*, doi: 10.1063/1.1144830
30. Ajith, Abraham. (2005). 39. 129 Artificial Neural Networks. doi: 10.1002/0471497398.MM421