ASME Early Career Technical Journal
2010 ASME Early Career Technical Conference, ASME ECTC
October 1 – 2, Atlanta, Georgia USA

SOLVING COUPLED SYSTEMS OF DIFFERENTIAL EQUATIONS USING THE LENGTH FACTOR ARTIFICIAL NEURAL NETWORK METHOD

Kevin Stanley McFall
The Pennsylvania State University
Lehigh Valley Campus
Center Valley, PA, USA

ABSTRACT

The length factor artificial neural network method for solving differential equations has previously been shown to successfully solve boundary value problems involving partial differential equations. This manuscript extends the method to solve coupled systems of partial differential equations, including accurate approximation of local Nusselt numbers in boundary layers and solving the Navier-Stokes equations for the entry length problem. With strengths including an explicit and continuous approximate solution, elimination of meshing concerns and simple implementation for nonlinear differential equations, this method is emerging as a viable alternative to traditional numerical techniques such as the finite element method.

INTRODUCTION

Many problems in science and engineering involve differential equations (DEs) sufficiently complicated to require numerical techniques for approximating their solutions. Traditionally, the finite difference [1], finite element [2], and boundary element [3] methods have been employed to numerically solve DEs. Although powerful and widespread, these tools do have drawbacks including problematic discretization of the problem domain and complications in solving nonlinear DEs. Artificial neural networks (ANNs) have emerged as an alternative method for numerical solution of DEs [4-12]. Methods using ANNs generally avoid the drawbacks of traditional numerical techniques. For example, domain discretization often involves simple square grids where no special treatment is necessary for nonlinear DEs.

ANN methods begin with a trial approximate solution (TAS) continuous over the problem domain whose value is influenced by a number of ANN parameters initialized to random values. Those parameters are then optimized to most closely approximate a solution to the given DE equation. While most ANN methods follow this general strategy, they vary greatly in implementation. The length factor ANN method featured here was developed to be simple in order to improve accessibility to those less familiar with ANNs. The hallmark of this method is that the boundary conditions (BCs) associated with the DE are automatically satisfied during all stages of training the ANN, including during initialization of network parameters with random values. Such an approach removes the BC constraint, allowing a simpler and more straightforward optimization stage compared with other ANN methods.

The length factor method for solving DEs has already been shown to successfully solve partial differential equations (PDEs) in two and three dimensions [8]. The main contribution of this manuscript is to expand the method to solve coupled systems of PDEs including the two-dimensional steady Navier-Stokes equations. The method is first demonstrated with a toy problem on an irregularly shaped domain, it produces an approximation for the local Nusselt number in the Blasius boundary layer, and it models the entrance region of flow between two infinite flat plates.

FORM OF THE TRIAL APPROXIMATE SOLUTION

Consider the two-dimensional, second-order differential equation defined by

$$f\left(\mathbf{x},\psi,\frac{\partial\psi}{\partial x},\frac{\partial\psi}{\partial y},\frac{\partial^2\psi}{\partial x^2},\frac{\partial^2\psi}{\partial y^2},\frac{\partial^2\psi}{\partial x\partial y}\right)=0 \qquad (1)$$

over the domain $\Omega\subset\mathbb{R}^2$ with exact analytical solution $\psi(\mathbf{x})$ subject to the Dirichlet BCs

$$\psi=g(\mathbf{x})\ \ \forall\mathbf{x}\in\partial\Omega \qquad (2)$$

for domain boundary $\partial\Omega$ given the position vector

$$\mathbf{x}=\begin{bmatrix}x\\y\end{bmatrix}\in\mathbb{R}^2 \qquad (3)$$

For the length factor method, the TAS to the DE represented by Equation (1) is of the form

$$\psi_t(\mathbf{x})=A_D(\mathbf{x})+L(\mathbf{x})N(\mathbf{x},\boldsymbol{\theta}) \qquad (4)$$

where the continuous function $A_D(\mathbf{x})$ is developed to satisfy the BCs such that

$$A_D = g(\mathbf{x}) \ \forall \mathbf{x} \in \partial\Omega \tag{5}$$

A_D may take any value inside the domain. The length factor L quantifies a measure of distance from the domain boundary and satisfies

$$L = 0 \ \forall \mathbf{x} \in \partial\Omega \text{ and } L \neq 0 \ \forall \mathbf{x} \in (\Omega - \partial\Omega) \tag{6}$$

The ANN output $N(\mathbf{x},\boldsymbol{\theta})$ depends of the spatial vector $\mathbf{x}$ as well as network parameters $\boldsymbol{\theta}$. Defined in such a manner, the TAS $\psi_t(\mathbf{x})$ automatically satisfies the BCs in Equation (2) regardless of network parameter values. Conversely, the ANN output controls the value of the TAS inside the domain and is adjusted by minimizing the function

$$G(\mathbf{x},\boldsymbol{\theta}) = f\left(\mathbf{x}, \psi_t, \frac{\partial\psi_t}{\partial x}, \frac{\partial\psi_t}{\partial y}, \frac{\partial^2\psi_t}{\partial x^2}, \frac{\partial^2\psi_t}{\partial y^2}, \frac{\partial^2\psi_t}{\partial x \partial y}\right) = 0 \tag{7}$$

by iteratively updating network parameters $\boldsymbol{\theta}$ with a gradient descent scheme. The exact functional values for A_D, L, and N are developed in the following sections.

GENERATING A_D AND L

The only constraints on the function A_D are that it must be continuous and return specific values on the domain boundary. Some problems lead to straightforward determination of an acceptable A_D function, such as

$$A_D = \sin\theta = \sin\left(\tan^{-1}\frac{y}{x}\right) = \frac{y}{\sqrt{x^2+y^2}} \tag{8}$$

for BCs requiring zero value on the x-axis and unity value on the y-axis. For domain shapes and BCs without obvious A_D functions, an appropriate function can be interpolated using thin plate splines (TPSs) [13] which are inspired by equilibrium of a plate exposed to forces at various locations called control points. The TPS for A_D is defined as

$$A_D(\mathbf{x}) = \sum_{i=1}^{n} F_i r_i^2 \ln r_i^2 + F_{n+1} + F_{n+2}x + F_{n+3}y \tag{9}$$

where

$$r_i = \sqrt{(x-\tilde{x}_i)^2 + (y-\tilde{y}_i)^2 + d^2} \tag{10}$$

A number of control points

$$\tilde{\mathbf{x}}_i = \begin{bmatrix} \tilde{x}_i \\ \tilde{y}_i \end{bmatrix} \in \partial\Omega \text{ for } 1 \le i \le n \tag{11}$$

are chosen for the TPS where the value of A_D is to be correctly specified. The $1 \le i \le n+3$ TPS parameters F_i are determined by solving a linear system where the first n equations

$$A_D(\tilde{\mathbf{x}}_i) = g(\tilde{\mathbf{x}}_i) \text{ for } 1 \le i \le n \tag{12}$$

ensure the desired values for A_D, and the final three

$$\sum_{i=1}^{n} F_i = 0, \ \sum_{i=1}^{n} F_i x = 0, \text{ and } \sum_{i=1}^{n} F_i y = 0 \tag{13}$$

ensure force and moment balance. The parameter d is chosen as the small value 0.01 so that each force F_i represents approximately a point force with localized effect at each of the control points. Use of TPSs allows for straightforward creation of an appropriate A_D function for arbitrary domain shapes with Dirichlet BCs.

Like A_D, the length factor L can be determined ad hoc for a given problem. For example,

$$L = xy \tag{14}$$

is appropriate for a domain boundary including the x and y axes. Length factors for complicated domain shapes can be constructed using a two-dimensional TPS to map the boundary control points onto a circle and computing the distance of a point inside the domain from that circle [8].

ARTIFICIAL NEURAL NETWORK DEFINITION

ANNs are inspired by the brain where neurons send electric pulses to neighboring output neurons if the neuron in question receives sufficiently many pulses from the input neurons connected to it. Multilayer perceptron (MLP) networks as illustrated in Figure 1 are one of the most common architectures of ANNs. MLPs are characterized by H hidden nodes where the output of the j^{th} hidden node

$$\sigma_j = \frac{1}{1+e^{-\left(w_{j,0}+\sum_{i=1}^{I} w_{j,i}x_i\right)}} \text{ for } 1 \le j \le H \tag{15}$$

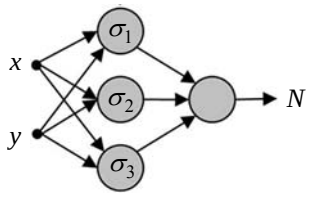

Figure 1: Illustration of a multilayer perceptron structure with two inputs (x and y), three hidden nodes, and a single output (N).

depends on the weighted sum of the I inputs to the ANN where $w_{j,i}$ is the weight for input x_i for node j, and $w_{j,0}$ is the j^{th} node's bias. The logistic sigmoid function in Equation (15) is chosen to approximate the binary decision of sending or not sending an electric pulse with a continuous function of range (0,1). Hidden nodes are so named since they are directly connected to neither input nor output. The final output of the ANN

$$N(\mathbf{x},\boldsymbol{\theta}) = u_0 + \sum_{j=1}^{H} u_j \sigma_j = u_0 + \sum_{j=1}^{H} \frac{u_j}{1+e^{-\left(w_{j,0}+\sum_{i=1}^{I} w_{j,i}x_i\right)}} \tag{16}$$

is defined as a weighted sum of the hidden node values. The output N is an explicit function depending on the position vector $\mathbf{x}$ and parameter vector

$$\boldsymbol{\theta} = \begin{bmatrix} w_{1,0} & \cdots & w_{H,I} & u_0 & \cdots & u_H \end{bmatrix} \tag{17}$$

which includes the weights and biases for the hidden and output nodes, with a total of $(I+2)H+1$ network parameters. All problems solved here have $I=2$ inputs (x and y) and $H=30$ hidden nodes. MLP networks have been shown to be universal approximators [14], ensuring that a $\boldsymbol{\theta}$ exists so that N can approximate any function of $\mathbf{x}$ with an arbitrarily small error given sufficiently many hidden nodes.

TRAINING THE ARTIFICIAL NEURAL NETWORK

One of the most common algorithms for optimizing, or training, the ANN is the gradient descent Levenberg-Marquardt technique [15]. Executing this technique involves selecting a discrete set of points S in the domain at which to evaluate the function in Equation (7). All problems solved here use a 40×40 square grid over the domain to define S. The sum-squared error is defined as

$$E(\boldsymbol{\theta}) = \sum_{i=1}^{\|S\|} G(\mathbf{x}_i,\boldsymbol{\theta})^2 = \mathbf{G}^{\mathrm{T}}\mathbf{G} \tag{18}$$

where

$$\mathbf{G}(\boldsymbol{\theta}) = \begin{bmatrix} G(\mathbf{x}_1,\boldsymbol{\theta}) & \cdots & G(\mathbf{x}_{\|S\|},\boldsymbol{\theta}) \end{bmatrix}^{\mathrm{T}} \tag{19}$$

given that $\mathbf{x}_i \in S$ for $1 \le i \le \|S\|$ and $S \subset \Omega$.

The error E can be expanded as the Taylor series

$$E(\boldsymbol{\theta}_0 + d\boldsymbol{\theta}) = E(\boldsymbol{\theta}_0) + \mathbf{g}^{\mathrm{T}} d\boldsymbol{\theta} + \tfrac{1}{2} d\boldsymbol{\theta}^{\mathrm{T}} \mathbf{H} d\boldsymbol{\theta} + \text{H.O.T.} \tag{20}$$

around arbitrary point $\boldsymbol{\theta}_0$ where $\mathbf{g}$ is the error gradient and $\mathbf{H}$ is the Hessian matrix. The optimal change $d\boldsymbol{\theta}$ is determined by differentiating Equation (20) with respect to $\boldsymbol{\theta}$, setting the result to zero, and solving for

$$d\boldsymbol{\theta} = -\mathbf{H}^{-1}\mathbf{g} = -\left(2\mathbf{J}^{\mathrm{T}}\mathbf{J} + 2\frac{\partial \mathbf{J}^{\mathrm{T}}}{\partial \boldsymbol{\theta}}\mathbf{G}\right)^{-1}\mathbf{g} \tag{21}$$

where the Hessian is replaced with an identity involving the Jacobian matrix

$$\mathbf{J}(\boldsymbol{\theta}) = \begin{bmatrix} \dfrac{\partial G(\mathbf{x}_1,\boldsymbol{\theta})^{\mathrm{T}}}{\partial \boldsymbol{\theta}} & \cdots & \dfrac{\partial G(\mathbf{x}_{\|S\|},\boldsymbol{\theta})^{\mathrm{T}}}{\partial \boldsymbol{\theta}} \end{bmatrix}^{\mathrm{T}} \tag{22}$$

In order to avoid computing mixed second derivatives, Equation (21) is approximated as

$$d\boldsymbol{\theta} \approx -\left(2\mathbf{J}^{\mathrm{T}}\mathbf{J} + \lambda\mathbf{I}\right)^{-1}\mathbf{g} \tag{23}$$

where λ is a small number chosen to be 0.01 for all calculations here. ANN parameters are updated iteratively according to the scheme

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \eta\left(2\mathbf{J}^{\mathrm{T}}\mathbf{J} + \lambda\mathbf{I}\right)^{-1}\mathbf{g} \tag{24}$$

where the parameter η is adjusted every iteration by a line search algorithm [16] where the largest value is chosen such that $\Delta\boldsymbol{\theta} = \boldsymbol{\theta}_{i+1} - \boldsymbol{\theta}_i$ is still a descent direction. This scheme converges rapidly to the (possibly locally) minimal error value once a region in the parameter space is found with sufficiently low error.

Practically implementing this learning algorithm involves computing the sensitivities of G to each ANN parameter (i.e. the weights and biases $w_{j,i}$ and u_j) evaluated at the points selected in S. If the DE in question were, for example, the Laplace equation

$$G = \frac{\partial^2 \psi_t}{\partial x^2} + \frac{\partial^2 \psi_t}{\partial y^2} \tag{25}$$

then the partial derivatives

$$\frac{\partial^3 \psi_t}{\partial x^2 \partial w_{j,i}}, \frac{\partial^3 \psi_t}{\partial y^2 \partial w_{j,i}}, \frac{\partial^3 \psi_t}{\partial x^2 \partial u_j}, \text{ and } \frac{\partial^3 \psi_t}{\partial y^2 \partial u_j} \tag{26}$$

would be required; they are straightforward to obtain from the TAS in Equation (4) and the explicit expression for ANN output in Equation (16).

EXPANDING TRAINING TO COUPLED SYSTEMS

The learning algorithm described in the previous section is appropriately developed to solve a boundary

value problem whose DE is defined by f in Equation (1) and whose Dirichlet BCs are satisfied by appropriately choosing A_D and L for the TAS in Equation (4). Expanding to solve systems of coupled DEs involves solving for the $1 \le i \le m$ variables $\psi^{(i)}$ represented approximately by m DEs

$$G_k = f_k\left(\mathbf{x}, \psi_t^{(1)}, \cdots, \frac{\partial^2 \psi_t^{(1)}}{\partial x \partial y}, \cdots, \psi_t^{(m)}, \cdots \frac{\partial^2 \psi_t^{(m)}}{\partial x \partial y}\right) \quad (27)$$

with $1 \le k \le m$. In general, each DE f_k depends on the position vector as well as the value and partial derivatives of all the $1 \le i \le m$ variables $\psi^{(i)}$. The sum-squared error

$$E(\boldsymbol{\theta}) = \sum_{k=1}^{m} \mathbf{G}_k^{\mathrm{T}} \mathbf{G}_k \quad (28)$$

is developed by adding the error from approximating each DE. The Jacobian for each DE

$$\mathbf{J}_k(\boldsymbol{\theta}) = \left[\frac{\partial G_k(\mathbf{x}_1, \boldsymbol{\theta})^{\mathrm{T}}}{\partial \boldsymbol{\theta}} \quad \cdots \quad \frac{\partial G_k(\mathbf{x}_{\|S\|}, \boldsymbol{\theta})^{\mathrm{T}}}{\partial \boldsymbol{\theta}} \right]^{\mathrm{T}} \quad (29)$$

can be combined to correctly approximate the Hessian as

$$\mathbf{H} \approx 2\sum_{k=1}^{m} \mathbf{J}_k^{\mathrm{T}} \mathbf{J}_k + \lambda \mathbf{I} \quad (30)$$

The learning rule for coupled systems of DEs is then

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \eta \left(2\sum_{k=1}^{m} \mathbf{J}_k^{\mathrm{T}} \mathbf{J}_k + \lambda \mathbf{I} \right)^{-1} \mathbf{g} \quad (31)$$

which involves simply summing the contributions for each DE before inverting the approximate Hessian.

DEMONSTRATION WITH A TOY PROBLEM

In order to demonstrate the technique, consider the functions

$$\begin{aligned} \psi^{(1)} &= -\frac{B_1}{A_1} e^{A_1 x} + \frac{B_2}{A_2^2} e^{A_2 y} \text{ and} \\ \psi^{(2)} &= B_1 e^{A_1 x} - \frac{B_2}{A_2} e^{A_2 y} \end{aligned} \quad (32)$$

which satisfy the coupled system

$$\begin{aligned} &\frac{\partial^2 \psi^{(1)}}{\partial x^2} + \frac{\partial^2 \psi^{(1)}}{\partial y^2} + \frac{\partial \psi^{(2)}}{\partial x} + \frac{\partial \psi^{(2)}}{\partial y} = 0 \text{ and} \\ &\frac{\partial \psi^{(1)}}{\partial x} + \frac{\partial \psi^{(2)}}{\partial y} + B_1 e^{A_1 x} + B_2 e^{A_2 y} = 0 \end{aligned} \quad (33)$$

The BCs for this problem are created by evaluating the analytical solutions along the cardioid boundary

$$r = \tfrac{3}{2}\left(\tfrac{3}{2} + \cos\theta\right) \quad (34)$$

using constants

$$A_1 = 0.85, A_2 = 0.2, B_1 = -0.3, \text{ and } B_2 = 0.3 \quad (35)$$

The TPS generated length factor appears in Figure 2 where markers indicate the chosen control points. The functions A_D for both unknowns are developed according to Equation (9) using the same control points in Figure 2 set to values according to Equation (32). The resulting TASs for $\psi^{(1)}$ and $\psi^{(2)}$ were optimized and compared with the analytical solutions to produce the percentage errors appearing in Figure 3. Approximations for both variables were excellent, with the largest local error at less than three thousands of a percent.

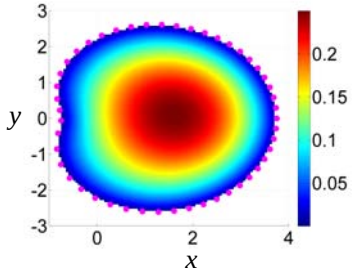

Figure 2: Length factor for the cardioid domain generated using the marked control points.

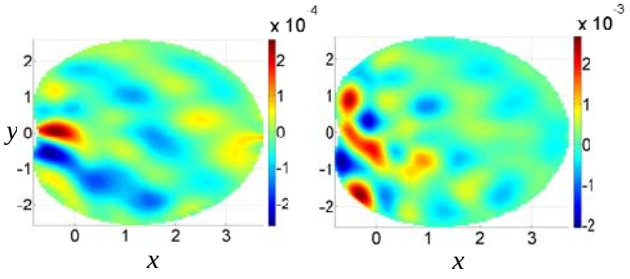

Figure 3: Percentage errors in the variables (a) $\psi^{(1)}$ and (b) $\psi^{(2)}$ as approximated by the ANN.

MODELLING THE BLASIUS BOUNDARY LAYER

The Blasius boundary layer involving laminar flow over a semi-infinite plate of length L held parallel to a constant and uniform flow is subject to the continuity equation

$$\frac{\partial u^*}{\partial x} + \frac{\partial v^*}{\partial y} = 0 \tag{36}$$

momentum balance equation

$$u^* \frac{\partial u^*}{\partial x^*} + v^* \frac{\partial u^*}{\partial y^*} = \frac{1}{\mathrm{Re}} \frac{\partial^2 u^*}{\partial y^{*2}} \tag{37}$$

and energy balance equation

$$u^* \frac{\partial T^*}{\partial x^*} + v^* \frac{\partial T^*}{\partial y^*} = \frac{1}{\mathrm{Re}\,\mathrm{Pr}} \frac{\partial^2 T^*}{\partial y^{*2}} \tag{38}$$

The governing equations are normalized to limit the magnitude of values necessary for the ANN to represent. The three unknowns are the dimensionless x- and y-components of velocity u^* and v^*, as well as the dimensionless temperature T^*. BCs on these three variables are

$$u^* = \begin{cases} y^* = 0: & 0 \\ y^* \to \infty: & 1, \; v^*\left(y^* = 0\right) = 0, \\ x^* = 0: & 1 \end{cases} \quad T^* = \begin{cases} x^* = 0: & 0 \\ y^* = 0: & 1 \end{cases} \tag{39}$$

The TASs

$$\begin{aligned} u^* &\approx \psi_t^{(1)} = \frac{y^*}{\sqrt{x^{*2} + y^{*2}}} + x^* y^* N_1 \\ v^* &\approx \psi_t^{(2)} = y^* N_2 \text{ and} \\ T^* &\approx \psi_t^{(3)} = \frac{x^*}{\sqrt{x^{*2} + y^{*2}}} + x^* y^* N_3 \end{aligned} \tag{40}$$

satisfy all the BCs except for the requirement that u^* approach unity as y^* approaches infinity. This requirement is intentionally omitted to illustrate that this ANN method can successfully solve DEs even with incomplete boundary information.

The local dimensionless heat transfer coefficient is represented by the local Nusselt number

$$\mathrm{Nu}_x = x^* \left. \frac{\partial T^*}{\partial y^*} \right|_{x^*, y^* = 0} \tag{41}$$

which is straightforward to compute from the TAS in Equation (40) and the ANN output in Equation (16). A generally accepted correlation for Nu_x is

$$\mathrm{Nu}_x = 0.332\,\mathrm{Re}_x^{\frac{1}{2}}\,\mathrm{Pr}^{\frac{1}{3}} \tag{42}$$

whose error can be as high as 25% depending on flow conditions [17].

The coupled system in Equations (36), (37), and (38) is traditionally solved by recognizing self-similarity and defining

$$\eta = y^* \sqrt{\frac{L u_\infty}{\nu x^*}} \tag{43}$$

with the free stream velocity u_∞ and kinematic viscosity ν. The coupled system reduces to the ordinary DE

$$f''' + f f'' = 0 \tag{44}$$

where $f(\eta)$ is a function η alone, and dimensionless temperature is expressed as

$$T^*(\eta) = 1 - \frac{\int_0^\eta \left(f''\right)^{\mathrm{Pr}}\, d\eta}{\int_0^\infty \left(f''\right)^{\mathrm{Pr}}\, d\eta} \tag{45}$$

Equation (44) can be approximated with an iterative technique [18] and Nu_x in Equation (41) rewritten

$$\mathrm{Nu}_x = x^* \left. \frac{\partial T^*}{\partial \eta^*} \right|_{x^*, y^* = 0} \left. \frac{\partial \eta}{\partial y} = \frac{\partial T^*}{\partial \eta^*} \right|_{x^*, y^* = 0} \mathrm{Re}^{\frac{1}{2}} \tag{46}$$

using the chain rule.

The TASs in Equation (40) are optimized with ANNs for a 10 cm long plate, free stream velocity of 1 cm/s, and property data for air at room temperature, i.e. Re=62.9 and Pr=0.707. TASs for u^*, v^*, and T^* appear in Figures 4, 5, and 6 respectively. Figure 7 compares the resulting local Nusselt number in Equation (41) with the accepted

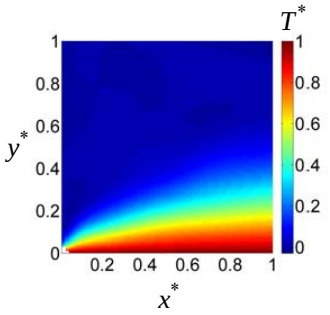

Figure 4: Dimensionless temperature T^* in the Blasius boundary layer as approximated by the ANN.

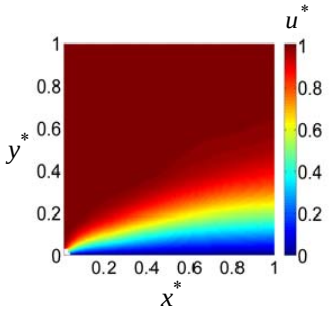

Figure 5: Dimensionless horizontal speed u^* in the Blasius boundary layer as approximated by the ANN.

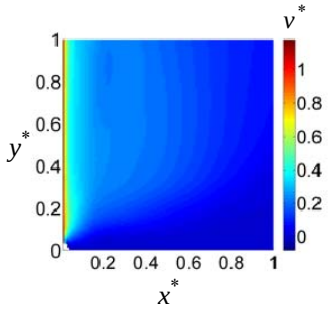

Figure 6: Dimensionless vertical speed v^* in the Blasius boundary layer as approximated by the ANN.

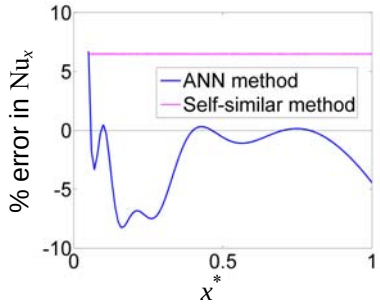

Figure 7: Percentage error in local Nusselt number for the ANN method and the self-similar method.

correlation in Equation (42) as a percent error, including error in the self-similar approximation using Equation (46) as a benchmark. Error in the ANN method is smaller over most of the plate's length than the self-similar method, and easily within the 25% error margin. Results are worst near the leading edge of the plate due to the discontinuous BCs in both u^* and T^* at the origin. The ANN has difficulty in this region as it is updated based on derivatives required to approach infinity at the origin. Indeed the point at the origin in the 40×40 training grid must be removed in order for the method to operate at all. Also note that

$$\lim_{x^*\to 0} v^* = \infty \tag{47}$$

in the analytical solution is impossible for the ANN to model correctly. Despite discontinuities, an analytical solution with infinite values, and incompletely specified BCs for u^*, the local Nusselt number predicted by the ANN is in general more accurate than the traditional self-similar technique for solving this problem.

MODELLING ENTRANCE FLOW

The normalized Navier-Stokes equations for steady two-dimensional flow ignoring body forces

$$\begin{aligned}\frac{\partial p^*}{\partial x^*}+\left(u^*\frac{\partial u^*}{\partial x^*}+v^*\frac{\partial u^*}{\partial y^*}\right)-\frac{1}{\mathrm{Re}}\left(\frac{\partial^2 u^*}{\partial x^{*2}}+\frac{\partial^2 u^*}{\partial y^{*2}}\right)=0 \text{ and}\\ \frac{\partial p^*}{\partial y^*}+\left(u^*\frac{\partial v^*}{\partial x^*}+v^*\frac{\partial v^*}{\partial y^*}\right)-\frac{1}{\mathrm{Re}}\left(\frac{\partial^2 v^*}{\partial x^{*2}}+\frac{\partial^2 v^*}{\partial y^{*2}}\right)=0\end{aligned} \tag{48}$$

along with the continuity equation

$$\frac{\partial u^*}{\partial x^*}+\frac{\partial v^*}{\partial y^*}=0 \tag{49}$$

govern the entry length problem for flow between two parallel plates with a steady and uniform horizontal inlet velocity. The BCs on dimensionless x and y velocity components and dimensionless pressure

$$u^*=\begin{cases} x^*=0: & 1\\ y^*=\pm\frac{1}{2}: & 0\end{cases}, v^*\left(y^*=\pm\tfrac{1}{2}\right)=0,\ p^*\left(x^*=0\right)=0 \tag{50}$$

are satisfied with the TASs

$$\begin{aligned} u^* &\approx \psi_t^{(1)}=\frac{\frac{1}{4}-y^{*2}}{\sqrt{x^{*2}+\left(\frac{1}{4}-y^{*2}\right)^2}}+x^*\left(\tfrac{1}{4}-y^{*2}\right)N_1\\ v^* &\approx \psi_t^{(2)}=\left(\tfrac{1}{4}-y^{*2}\right)N_2 \text{ and}\\ p^* &\approx \psi_t^{(3)}=x^*N_3\end{aligned} \tag{51}$$

The approximations of u^*, v^*, and p^* for laminar flow with Re=15 appear in Figures 8, 9, and 10 respectively. The ANNs producing these approximations were trained using a 120×40 grid reflecting the domain's 3:1 aspect ratio. Again, the analytical solution at the entry corners will have infinite derivatives and so those two points are removed from S.

Fully-developed flow can be predicted to occur at a dimensionless entrance length [19]

$$l_e^*=0.06\,\mathrm{Re} \tag{52}$$

with a corresponding velocity profile of

$$u^* = \tfrac{1}{2}\mathrm{Re}\frac{\partial p^*}{\partial x^*}\left(y^{*2} - \tfrac{1}{4}\right) \tag{53}$$

The fully-developed region is characterized by a velocity uniform in the flow direction. The velocity gradient $\partial u^*/\partial x^*$ for the u^* in Figure 8 averaged over the flow cross-section appears in Figure 11. The gradient drops sharply just after the predicted entrance length of l_e^*=0.9 to remain below 10^{-2}, representing changes in speed of less than a 1% of the free stream velocity over a distance equal to the hydraulic diameter.

The volumetric flow rate

$$\int u^* dA^* = w^* \int_{-\frac{1}{4}}^{\frac{1}{4}} u^* dy^* \tag{54}$$

for a cross-section with dimensionless width w^* is equal to ½w^* at the inlet with u^*=1. Numerically approximating Equation (54) for u^* in Figure 8 in the fully-developed region results in 0.500w^* for the flow rate. Combining Equations (53) and (54) require that the dimensionless pressure gradient $\partial p^*/\partial x^*$ in the fully-developed region be 48/Re, or 3.2 in the case explored here. The pressure in Figure 10 has a gradient of 3.19 in the fully-developed region. Equation (53) and the fully-developed velocity profile approximated by the ANN are essentially coincident, as illustrated in Figure 12.

CONCLUSION

The length factor artificial neural network (ANN) method has been expanded to solve coupled systems of partial differential equations (DEs). This method uses approximate solutions to the DEs which depend on the output of an ANN but exactly satisfy Dirichlet boundary conditions (BCs) independent of the ANN. The approximate solution is then optimized by updating the ANN via gradient descent.

The main strength of this method is that the same solution method is followed regardless of the form or complexity of the DEs involved, including nonlinear equations like the Navier-Stokes equations. Every problem requires developing a length factor L dependent on the shape of the problem domain, and a function A_D based on the value of the solution on the boundary. Most irregularly-shaped domains preclude obvious ad hoc functions for L and A_D, but their generation can be automated in a straight-forward process using thin plate splines. Other benefits of the method include an explicit, differentiable approximate solution, the absence of meshing concerns, and success despite omission of some boundary conditions (BCs).

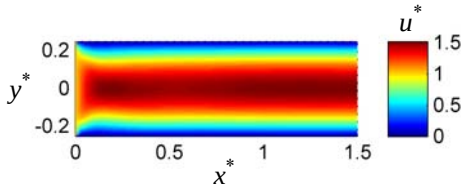

Figure 8: Dimensionless horizontal speed u^* for the entrance flow problem as approximated by the ANN.

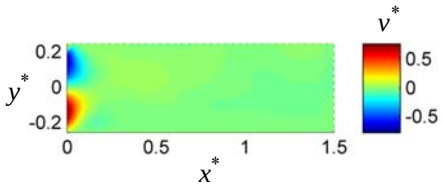

Figure 9: Dimensionless vertical speed v^* for the entrance flow problem as approximated by the ANN.

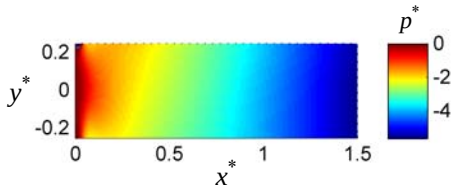

Figure 10: Dimensionless pressure p^* for the entrance flow problem as approximated by the ANN.

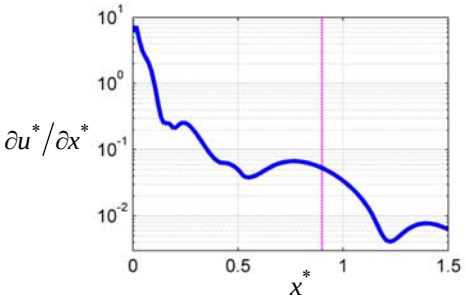

Figure 11: Velocity gradient $\partial u^*/\partial x^*$ averaged over the flow cross-section as approximated by the ANN, with the location for fully-developed flow as predicted by Equation (52) indicated with the vertical line.

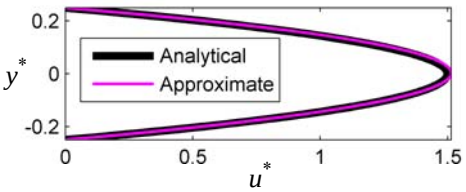

Figure 12: Dimensionless velocity profile in the fully-developed region as determined analytically and approximated by the ANN.

This manuscript offers the first time the length factor method has been shown to solve coupled systems of DEs. Also for the first time, the method has been shown to accurately solve reasonably complicated problems of interest in science and engineering, including finding the local Nusselt number in the Blasius boundary layer and solving the Navier-Stokes equations for laminar flow between parallel plates. Accurate solutions for these problems are obtainable despite the fact that the analytical solutions involve discontinuities impossible for the continuous trial approximate solution (TAS) to model exactly. Optimization of the TAS is minimally affected by such discontinuities since they arise from BCs which are automatically satisfied independent of the ANN due to the unique design of the length factor method. Since the value of the TAS is correctly specified near the discontinuity, the DEs need not be satisfied accurately there. However, points in the domain too close to discontinuities should not be used for training since optimization will concentrate on regions with infinite derivatives at the expense of the remainder of the domain. While using a square training grid does avoid complicated meshing schemes, problems with discontinuous solutions do require modest attention to remove training points with infinite derivatives.

Additionally, the method offers a continuous TAS so that quantities such as temperature, velocity, and pressure gradients are available analytically everywhere in the domain rather than requiring numerical approximation. Also of interest is the ability of the method to accurately solve problems with incomplete boundary conditions.

While not as recognized as traditional numerical techniques such as the finite element method (FEM), ANN methods, and the length factor method in particular, offer alternatives eliminating some vexing issues with the FEM. The length factor method shows promise as a tool for solving difficult problems in science and engineering.

REFERENCES

[1] G. Smith, *Numerical Solution of Partial Differential Equations: Finite Difference Methods*, Oxford, UK: Clarendon, 1978.

[2] S. Rao, *The Finite Element Method in Engineering*, Boston: Butterworth-Heinemann, 1999.

[3] C. Brebbia, J. Telles, and L. Wrobel, *Boundary Element Techniques: Theory and Applications in Engineering*, Berlin: Springer-Verlag, 1984.

[4] L. Aarts, P. Van Der Veer, and van der Veer, "Neural Network Method for Solving Partial Differential Equations," *Neural Processing Letters*, vol. 14, 2001, pp. 261-271.

[5] S. He, K. Reif, and R. Unbehauen, "Multilayer Neural Networks for Solving a Class of Partial Differential Equations," *Neural Networks*, vol. 13, 2000, pp. 385-396.

[6] I. Lagaris, A. Likas, and D. Papageorgiou, "Neural-network Methods for Boundary Value Problems with Irregular Boundaries," *Neural Networks, IEEE Transactions on*, vol. 11, 2000, pp. 1041-1049.

[7] N. Mai-Duy and T. Tran-Cong, "An Efficient Indirect RBFN-based Method for Numerical Solution of PDEs," *Numerical Methods for Partial Differential Equations*, vol. 21, 2005, pp. 770-790.

[8] K. McFall and J. Mahan, "Artificial Neural Network Method for Solution of Boundary Value Problems With Exact Satisfaction of Arbitrary Boundary Conditions," *IEEE Transactions on Neural Networks*, vol. 20, 2009, pp. 1221-1233.

[9] A. Meade and A. Fernandez, "The Numerical Solution of Linear Ordinary Differential Equations by Feedforward Neural Networks," *Mathematical and Computer Modelling*, vol. 19, 1994, pp. 1-25.

[10] D. Parisi, M. Mariani, and M. Laborde, "Solving Differential Equations with Unsupervised Neural Networks," *Chemical Engineering and Processing*, vol. 42, 2003, pp. 715-721.

[11] I.G. Tsoulos, D. Gavrilis, and E. Glavas, "Solving Differential Equations with Constructed Neural Networks," *Neurocomputing*, vol. 72, 2009, pp. 2385-2391.

[12] R. Yentis and M. Zaghloul, "VLSI Implementation of Locally Connected Neural Network for Solving Partial Differential Equations," *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on*, vol. 43, 1996, pp. 687-690.

[13] L. Zagorchev and A. Goshtasby, "A Comparative Study of Transformation Functions for Nonrigid Image Registration," *Image Processing, IEEE Transactions on*, vol. 15, 2006, pp. 529-538.

[14] K. Hornik, M. Stinchcombe, and H. White, "Multilayer Feedforward Networks are Universal Approximators," *Neural Networks*, vol. 2, 1989, pp. 359-366.

[15] J. Jang, C. Sun, and E. Mitzutani, *Neuro-fuzzy and Soft Computing*, Englewood Cliffs, NJ: Prentice-Hall, 1997.

[16] M. Albaali and R. Fletcher, "An Efficient Line Search for Nonlinear Least-squares," *Journal of Optimization Theory and Applications*, vol. 48, 1986, pp. 359-377.

[17] F.P. Incropera and D.P. DeWitt, *Fundamentals of Heat and Mass Transfer*, John Wiley, 2007.

[18] A. Mills, *Heat and Mass Transfer*, CRC Press, 1994.

[19] B.R. Munson, D.F. Young, and T.H. Okiishi, *Fundamentals of Fluid Mechanics*, Wiley, 2006.