Elsevier Editorial System(tm) for Computers and Fluids

Manuscript Draft

Manuscript Number:

Title: Non-Newtonian Flow field prediction: A Deep Galerkin Method approach

Article Type: Research paper

Keywords: Deep Galerkin Method, Navier-Stokes, Carreau Fluid, Blood

Corresponding Author: Mr. Amal Roy, B.Tech

Corresponding Author's Institution: School of Engineering, Cochin University of Science and Technology, Cochin, Kerala, India

First Author: Amal Roy, B.Tech

Order of Authors: Amal Roy, B.Tech; Mary Lidiya, B.Tech; Gireesh K Thampi, M.Tech, Ph.D

Abstract: Machine Learning approaches to solve Partial Differential Equations (PDEs) is a ubiquitous topic in contemporary research. The mesh free approach of `training' an arbitrary parameterized function on randomly selected locations in the domain alleviates requirement of high quality meshes and weak formulation of the governing equations. In this work, a novel machine learning algorithm called Deep Galerkin Method (DGM) is employed to solve the generalised Navier-Stokes equations describing the flow of blood through a stenosed artery model. Blood was modelled as a Carreau fluid and the problem was treated as steady in-compressible flow. The DGM network was trained to satisfy the differential operators and the solution was used to predict the axial velocity, wall shear stress and wall pressure. The results are validated against the solution of the same problem using Finite Volume Method(FVM) available from literature. The axial velocity, wall shear stress and wall pressure are seen to be in good agreement with the FVM solution.

Research Data Related to this Submission

Title: Data for: Non-Newtonian Flow field prediction: A Deep Galerkin Method approach
Repository: Mendeley Data
https://data.mendeley.com/datasets/n4nm8jnxsk/draft?a=482b5caa-e05e-4c5a-a89d-faafa54eee15

Non-Newtonian Flow field prediction: A Deep Galerkin Method approach

Amal Roy[a,c], Mary Lidiya[b,c], Gireesh K T[a,c]

[a] *Dept.of Mechanical Engineering, School of Engineering*
[b] *Dept.of Civil Engineering, School of Engineering*
[c] *Cochin University of Science and Technology, India*

Abstract

Machine Learning approaches to solve Partial Differential Equations (PDEs) is a ubiquitous topic in contemporary research. The mesh free approach of 'training' an arbitrary parameterized function on randomly selected locations in the domain alleviates requirement of high quality meshes and weak formulation of the governing equations. In this work, a novel machine learning algorithm called Deep Galerkin Method (DGM) is employed to solve the generalized Navier-Stokes equations describing the flow of blood through a stenosed artery model. Blood was modeled as a Carreau fluid and the problem was treated as steady incompressible flow. The DGM network was trained to satisfy the differential operators and the solution was used to predict the axial velocity, wall shear stress and wall pressure. The results are validated against the solution of the same problem using Finite Volume Method(FVM) available from literature. The axial velocity, wall shear stress and wall pressure are seen to be in good agreement with the FVM solution.

Keywords: Deep Galerkin Method, Navier-Stokes, Carreau Fluid, Blood

1. Introduction

Artificial Neural Networks (ANNs) and its potential applications have become a predominant interest for recent scientific research. This primarily owes to their abstractness and versatility. Their ability to approximate complex mathematical functions enable them to describe complex physical phenomenons also [1, 2, 3, 4, 5, 6, 7]. At the same time, methods to accurately solve the partial differential equations representing such phenomenons are still cumbersome and expensive for complex engineering applications.

Thus the advent of neural networks and machine learning techniques to aid in solving these PDEs seems intuitive and natural.

The seminal work demonstrating the effectiveness of neural network to solve PDEs were done by Lagaris *et al.* [8, 9]. They demonstrated the convergence of neural network solution using a combination of Multi-Layer Perceptrons (MLP) and radial basis function networks. From then on, a long line of researchers have implemented various machine learning methods including MLP, Evolutionary Polynomial Regression (EPR) and Deep Neural Networks to develop simulations. They are employed in a variety of applications as a child model representing complex functions within the solver or as a surrogate to enhance the accuracy of the classical Finite Element or Volume methods [10, 11, 12, 13, 14, 15, 16, 17, 18]. Authors also demonstrated that the deep neural network can be independently used to solve PDEs on complex geometries and the solutions are qualitatively similar to that obtained using conventional methods [19, 20, 21]. Recently, ANNs were successfully employed in Computational Fluid Dynamics(CFD) to solve for the approximate solution of Navier-Stokes equations [22]. Raissi *et al.* [23] demonstrated that neural networks can be a reliable tool for both solving the PDEs as well as for data driven discovery of the PDE systems.

The main advantage of using ANNs to solve PDE arises from the fact that they are universal function approximators and the solution is a continuous function differentiable with respect to any of the input or network variables [24, 25, 26, 27]. This is in stark contrast with conventional numerical methods which produce a discrete approximate solution with limited differentiability [28, 29, 30]. Moreover, the training of neural networks do not require high quality computational grids which significantly reduces the effort for a successful simulation.

That being said, a widely attempted yet unaccomplished problem in fluid dynamics is the simulation of blood rheology. Owing to its biochemical composition, blood exhibits special features [31, 32] which are comparatively difficult to be described by simple mathematical models. Though researchers were able to approximate its rheological behavior using various fluid models, most of them include difficult mathematical formulation. As a matter of fact, recently proposed blood models [33, 34, 35] are increasingly difficult to be solved using traditional simulation techniques. At this juncture, an intuitive, adaptable and stable method for solving these PDEs would be of paramount importance for the swift advancements in hemorheology as well as in the numerous interests of computational modeling.

In this work, a novel neural network architecture called the Deep Galerkin Method (DGM) is employed to simulate the flow of human blood through a stenosed artery model. In essence, the problem condenses to solving the Navier-Stokes equations governing the dynamics of fluid flow. Therefore we demonstrate a method to solve the system of PDEs describing the rheological behavior of blood as a machine learning problem. The Carreau fluid model is used to represent human blood and the results are validated against CFD simulation results of blood flow from relevant literature.

A summary of the DGM algorithm [36] is presented in section 2 along with practice details. Formulation of the blood flow problem and implementation details of the DGM algorithm described in section 4 and 5 respectively.

2. The Deep Galerkin Method

The Deep Galerkin Method (DGM) [36] is a merger of the Galerkin Methods and machine learning and is a viable solution to the 'curse of dimensionality' in solving high dimensional PDEs [37]. However, it stands untested in its ability to be useful in engineering simulations. The main advantage of DGM algorithm is its mesh free approach, simplicity of its formulation and an intuitive loss function and network architecture

The architecture of DGM is a single feed forward neural network capable of approximating high dimensional functions. The network architecture of DGM consists of a single or a stack of serially connected DGM layers bounded by the input and output layers. A striking feature of the network is the repetition of these identical DGM layers suggesting that its learning capabilities are similar to the LSTM cells [38]. In addition, the input vector to the network is passed on to each of DGM layers in addition to the input layer.

During training, output layer of the network is connected to differential operators that calculate the residues of the system of PDEs along with initial and boundary conditions. Each forward propagation of the network involves the calculation of the dependent variable z at the data point x and subsequent calculation of the residues. The loss function of the network is formulated as the square sum of these residues.

The network is then trained by mini-batch stochastic gradient descent using the well known Adam algorithm [39]. Convergence of the network weights so as to satisfy the differential operators in the loss function renders the network as an approximate solution to the PDE [36].

3. Blood Rheology and Problem Details

The primary challenge in simulating blood flow arises from its non-Newtonian properties such as shear thinning, viscoelasticity and thixotropy. Although viscoelasticity and thixotropy have only secondary significance in the physiology of blood flow [40], shear thinning becomes prominent when blood flows at shear rates less than $100s^{-1}$ as in small arteries[41, 42, 43, 44]. The characteristics of blood flow therefore necessitates the choice of a blood model that can incorporate the non-Newtonian behavior of blood for an accurate simulation.

The rheology of blood flow can be approximated by several models which account for non-Newtonian nature of the fluid. Numerous blood models were designed, studied and pioneered by researchers [44, 45, 46, 47, 48, 49, 50] but apparently no particular model is acknowledged to have an infallible definition of hemorheology. Among these, the Carreau model reasonably predicts the non-Newtonian nature of blood over a wide range of shear rates and concurs well with the experimental data [51, 52].This model tends towards a constant viscosity at higher shear rates and also does not over-predict the non-Newtonian behavior at low shear rates[52]. That being said, the blood flow simulation attempted in this paper should be considered as a representation for the effectiveness of the machine learning approach.

We attempt to solve the flow parameters, when blood flows through a stenosed artery model with constant Reynolds number of 300. The Carreau fluid model is used to represent the rheology of blood which considers blood as an incompressible fluid with viscosity dependent on the local strain rate.

The geometry of arterial vessel and cosine representation of the stenosis is adopted from the work of Zhou *et al.* [53] as given in Fig.1. The degree of stenosis η_s, defined as the ratio of the blockage area to the cross sectional area of the normal vessel, is given by

$$\eta_s = \frac{\pi R_0^2 - \pi R_t^2}{\pi R_0^2} = 1 - \left(\frac{R_t}{R_0}\right)^2 \tag{1}$$

where, R_0 is the radius of the artery and R_t is the radius at maximum constriction.

The radius of the arterial model is 5mm and the length of the stenosis is set to be four times the radius of artery. The mean velocity at the inlet is taken as 0.0986 m/s and the inlet velocity profile is parabolic. The outlet of

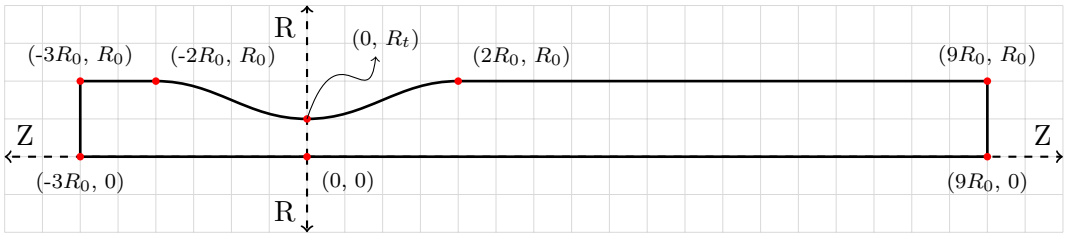

Figure 1: Geometry of the stenosed artery model.

the artery is set to have zero pressure and the stenosis at the rigid arterial wall is modeled using cosine function as

$$R = R_0 \left[1 - \frac{1}{2} \left(1 - \sqrt{1 - \eta_s} \right) \left(1 + \cos \left(\frac{2\pi}{L_s} \left(z - z_s \right) \right) \right) \right] \tag{2}$$

4. Formulation of the problem

The generalized Navier-Stokes equations [53] in vector form can be written as

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \mathbf{U}) &= 0 \\ \frac{\partial \rho \mathbf{U}}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \mathbf{U}\mathbf{U}) &= -\boldsymbol{\nabla} p + \boldsymbol{\nabla} \cdot \boldsymbol{\tau} + \rho \overrightarrow{f} \end{aligned} \tag{3}$$

where the non-Newtonian characteristic is contributed by the strain-rate dependent stress τ of Carreau model [53] given by

$$\tau = \mu(\dot{\gamma})(\nabla U + \nabla U^T) \tag{4}$$

and the Carreau viscosity [44] is defined as

$$\mu(\dot{\gamma}) = \mu_\infty + (\mu_0 - \mu_\infty)(1 + (\lambda \dot{\gamma})^2)^{\frac{n-1}{2}} \tag{5}$$

The scalar, local shear rate $\dot{\gamma}$, is related to the second invariant of rate of strain tensor [44] $\dot{\gamma}$ as

$$\dot{\gamma} = \sqrt{\frac{1}{2}\Pi} = \sqrt{\frac{1}{2}\left[\sum_i \sum_j \dot{\gamma}_{ij} \dot{\gamma}_{ji} \right]} \tag{6}$$

where $\dot{\gamma}_{ij} = U_{i,j} + U_{j,i}$ is the local rate of strain tensor.

The flow of blood through the stenosed artery model can be modeled by the steady incompressible form of Eqn.3 in axisymmetric coordinate system [54] as

$$\frac{\partial U_Z}{\partial Z} + \frac{\partial U_R}{\partial R} + \frac{U_R}{R} = 0 \tag{7}$$

$$\begin{aligned}\rho\left(U_R\frac{\partial U_Z}{\partial R} + U_Z\frac{\partial U_Z}{\partial Z}\right) + \frac{\partial P}{\partial Z} - \mu(\dot{\gamma})\left[\frac{\partial^2 U_Z}{\partial R^2} + \frac{1}{R}\frac{\partial U_Z}{\partial R} + \frac{\partial^2 U_Z}{\partial Z^2}\right] \\ -2\frac{\partial \mu}{\partial Z}\frac{\partial U_Z}{\partial Z} - \frac{\partial \mu}{\partial R}\left(\frac{\partial U_R}{\partial Z} + \frac{\partial U_Z}{\partial R}\right) = 0\end{aligned} \tag{8}$$

$$\begin{aligned}\rho\left(U_R\frac{\partial U_R}{\partial R} + U_Z\frac{\partial U_R}{\partial Z}\right) + \frac{\partial P}{\partial R} - \mu(\dot{\gamma})\left[\frac{\partial^2 U_R}{\partial Z^2} + \frac{1}{R}\frac{\partial U_R}{\partial R} + \frac{\partial^2 U_R}{\partial R^2} - \frac{U_R}{R^2}\right] \\ -2\frac{\partial \mu}{\partial R}\frac{\partial U_R}{\partial R} - \frac{\partial \mu}{\partial Z}\left(\frac{\partial U_R}{\partial Z} + \frac{\partial U_Z}{\partial R}\right) = 0\end{aligned} \tag{9}$$

The local strain rate in axisymmetric coordinate system can be obtained as

$$\dot{\gamma} = \sqrt{2\left[\left(\frac{\partial U_Z}{\partial Z}\right)^2 + \left(\frac{\partial U_R}{\partial R}\right)^2 + \left(\frac{U_R}{R}\right)^2\right] + \left[\frac{\partial U_R}{\partial Z} + \frac{\partial U_Z}{\partial R}\right]^2} \tag{10}$$

Training of the neural network is significantly enhanced by normalizing the data [55]. Therefore we chose to normalize the data and governing equations without which the network failed to converge. Therefore, equations (7), (8) and (9) are non-dimensionalized by defining new variables

$$\begin{gathered}z = \frac{Z}{R_0} \qquad r = \frac{R}{R_0} \qquad u_r = \frac{U_R}{U_0} \qquad u_z = \frac{U_Z}{U_0} \qquad p = \frac{P}{\rho U_0^2} \\ \tau = \frac{R_0}{U_0} \qquad \zeta = \frac{\mu_0}{\mu_\infty} \qquad Re = \frac{2R_0 \cdot U_0 \cdot \rho}{\mu_\infty} \qquad \Lambda = \frac{\lambda}{\tau} \qquad \Gamma = \tau \cdot \dot{\gamma}\end{gathered} \tag{11}$$

The non-dimensional equations in axisymmetric coordinate system is obtained as

$$\frac{\partial u_z}{\partial z} + \frac{\partial u_r}{\partial r} + \frac{u_r}{r} = 0 \tag{12}$$

$$u_r\frac{\partial u_z}{\partial r}+u_z\frac{\partial u_z}{\partial z}+\frac{\partial p}{\partial z}-2\frac{\chi}{Re}\left(\frac{\partial^2 u_z}{\partial z^2}+\frac{1}{r}\frac{\partial u_z}{\partial r}+\frac{\partial^2 u_z}{\partial r^2}\right) -\frac{2}{Re}\frac{\partial \chi}{\partial r}\left(\frac{\partial u_r}{\partial z}+\frac{\partial u_z}{\partial r}\right)-\frac{4}{Re}\frac{\partial \chi}{\partial z}\frac{\partial u_z}{\partial z}=0 \tag{13}$$

$$u_r\frac{\partial u_r}{\partial r}+u_z\frac{\partial u_r}{\partial z}+\frac{\partial p}{\partial r}-2\frac{\chi}{Re}\left(\frac{\partial^2 u_r}{\partial z^2}+\frac{1}{r}\frac{\partial u_r}{\partial r}+\frac{\partial^2 u_r}{\partial r^2}-\frac{u_r}{r^2}\right) -\frac{2}{Re}\frac{\partial \chi}{\partial z}\left(\frac{\partial u_r}{\partial z}+\frac{\partial u_z}{\partial r}\right)-\frac{4}{Re}\frac{\partial \chi}{\partial r}\frac{\partial u_r}{\partial r}=0 \tag{14}$$

where, the non-dimensionalized viscosity $\chi=\dfrac{\mu(\dot{\gamma})}{\mu_\infty}$ is obtained as

$$\chi=1+(\zeta-1)(1+(\Lambda\Gamma)^2)^{\frac{n-1}{2}} \tag{15}$$

where, non-dimensionalized shear rate Γ is obtained as

$$\Gamma=\sqrt{2\left[\left(\frac{\partial u_z}{\partial z}\right)^2+\left(\frac{\partial u_r}{\partial r}\right)^2+\left(\frac{u_r}{r}\right)^2\right]+\left[\frac{\partial u_r}{\partial z}+\frac{\partial u_z}{\partial r}\right]^2} \tag{16}$$

Inlet and no-slip condition at rigid arterial wall is obtained as

$$u_z=2u_0(1-\frac{r^2}{R_0^2}),\ \text{for } z=z_{min} \tag{17}$$

$$\frac{\partial u_z}{\partial r}=\frac{\partial u_r}{\partial r}=\frac{\partial p}{\partial r}=0,\ \text{for } r=0 \tag{18}$$

$$u_z=u_r=0,\quad \forall(z,r_s)\in\mathbf{R}^2\quad \text{such that} \tag{19}$$

$$r_s=\begin{cases} r_{max} & \text{if } |z-z_s|\geq\dfrac{L_s}{2} \\ r_{max}\left[1-\dfrac{1}{2}\left(1-\sqrt{1-\eta_s}\right)\left(1+\cos\left(\dfrac{2\pi}{L_s}\left(z-z_s\right)\right)\right)\right] & \text{if } |z-z_s|<\dfrac{L_s}{2}\end{cases} \tag{20}$$

where, η_s is the degree of stenosis, L_s is the length of stenosed area and (z_s,r_s) is the center of stenosis on the axis.

5. Implementation of DGM algorithm

5.1. Architecture

In the two dimensional steady incompressible Navier-Stokes equations given in section 4, independent variables are the non-dimensional spatial coordinates z and r and the outputs are the axial and radial velocity components u_z and u_r and a scalar pressure p. Thus we initialize the network weights as a tensor of rank 3 representing a stack of three weight matrices for each of the independent variables. Indeed this amounts to stacking three independent networks giving three scalar outputs. Since, the DGM algorithm approximates the solution fields u_z, u_r, and p with the neural network $\mathrm{F}(\Theta)$, we have

$$F(\Theta) = \begin{Bmatrix} f_1(\theta_1) \\ f_2(\theta_2) \\ f_3(\theta_3) \end{Bmatrix} \tag{21}$$

where, Θ is the state of the network weights and $\theta_1, \theta_2, \theta_3$ represents the states of weight matrices of independent networks corresponding to three output variable. Use of independent network weights was supported by the notion that it is always better to have less amount of information in each of the network weights [56].

Transformations:. It is well known that the training process of a neural network requires the system to traverse along the loss surface. The stochastic gradient descent method substantially improves training dynamics but the multi-agent multitask learning process like that of our implementation still suffers from the problem of local minima and opposing gradients. However, it is evident that reducing the span of this configuration space by explicitly filtering the undesired configurations can reduce the effort to achieve the Pareto optimality.

Thus unlike specifying the boundary conditions in the form of residuals in the objective function[1], the model is effectively restricted to take the form of a function that strictly satisfies the zero Dirichlet boundary conditions as follows:-

[1]Boundary conditions in the form of loss function terms attracts the obvious problem of lack of exactness at the boundary. The residual at the boundary will initially decrease but will stagnate when conflicting gradients arise during later training phase

Let the constrained output of the network be $\tilde{F}(\Theta)$ and the constraint be applied as

$$\tilde{F}(\Theta) = \begin{Bmatrix} \tilde{u}_z \\ \tilde{u}_r \\ \tilde{p} \end{Bmatrix} = \begin{Bmatrix} f_1(\theta_1) \cdot (r - r_w) \\ f_2(\theta_2) \cdot (r - r_w) \\ f_3(\theta_3) \cdot (z - z_{max}) \end{Bmatrix} \tag{22}$$

where, $\tilde{u}_z$, $\tilde{u}_r$, $\tilde{p}$ are outputs of the DGM network, z_{max} is the outlet location and r_w is given by the piece-wise function in Eqn.20 representing the stenosed artery wall. For the purpose of training, the network outputs are fed into a system of differential operators which calculate the value of the loss function for every point fed to the network.

The network used for the blood flow simulation in this work contains 5 DGM layers with 100 hidden units. The activation function used is ReLU [57].

5.1.1. Objective Function

The problem presented in section 4 reduces to the solution of three coupled partial differential equations subjected to boundary conditions. Although this inherently is a problem of a multi-agent multi-objective optimization from a machine learning perspective, the understanding of the physical phenomena provides opportunity to have another take on this. The DGM proposes to select the loss function terms as the square of residues to be minimized which in this case can be administered as the square sum of the individual PDE residues. Moreover, using a constrained form of the network as given by Eqn.21 alleviates the use of zero Dirichlet boundary residues in the objective function.

For the purpose of mean centering the input data, the origin is shifted to (z_m, r_m), where z_m and r_m is the mean of z and r coordinates of the data points respectively. Therefore, after the transformation, z $\leftarrow z - z_m$ and $r \leftarrow r - r_m$, the four loss expressions to be optimized can be formulated as below.

$$J_1(\Theta) \leftarrow \left\| \frac{\partial \tilde{u_z}}{\partial z} + \frac{\partial \tilde{u_r}}{\partial r} + \frac{\tilde{u_r}}{r - r_m} \right\|_\Omega^2 \tag{23a}$$

$$J_2(\Theta) \leftarrow \left\| \begin{array}{c} \tilde{u_r} \dfrac{\partial \tilde{u_z}}{\partial r} + \tilde{u_z} \dfrac{\partial \tilde{u_z}}{\partial z} + \dfrac{\partial \tilde{p}}{\partial z} - \dfrac{4}{Re} \dfrac{\partial \chi}{\partial r} \left(\dfrac{\partial \tilde{u_r}}{\partial z} + \dfrac{\partial \tilde{u_z}}{\partial r} \right) \\ -2 \dfrac{\chi}{Re} \left(\dfrac{\partial 2 \tilde{u_z}}{\partial z^2} + \dfrac{1}{r - r_m} \dfrac{\partial \tilde{u_z}}{\partial r} + \dfrac{\partial^2 \tilde{u_z}}{\partial r^2} \right) - \dfrac{2}{Re} \dfrac{\partial \chi}{\partial z} \dfrac{\partial \tilde{u_z}}{\partial z} \end{array} \right\|_\Omega^2 \tag{23b}$$

$$J_3(\Theta) \leftarrow \left\| \begin{aligned} &\tilde{u}_r \frac{\partial \tilde{u}_r}{\partial r} + \tilde{u}_z \frac{\partial \tilde{u}_r}{\partial z} + \frac{\partial \tilde{p}}{\partial r} - \frac{4}{Re}\frac{\partial \chi}{\partial z}\left(\frac{\partial \tilde{u}_r}{\partial z} + \frac{\partial \tilde{u}_z}{\partial r}\right) - \frac{2}{Re}\frac{\partial \chi}{\partial r}\frac{\partial \tilde{u}_r}{\partial r} \\ &-2\frac{\chi}{Re}\left(\frac{\partial^2 \tilde{u}_r}{\partial z^2} + \frac{1}{r - r_m}\frac{\partial \tilde{u}_r}{\partial r} + \frac{\partial^2 2\tilde{u}_r}{\partial r^2} - \frac{\tilde{u}_r}{(r - r_m)^2}\right) \end{aligned} \right\|_{\Omega}^2 \tag{23c}$$

$$J_b(\Theta) = \left\| \frac{\partial \tilde{u}_z}{\partial r} \right\|_{r=0}^2 + \left\| \frac{\partial \tilde{u}_r}{\partial r} \right\|_{r=0}^2 + \left\| \frac{\partial \tilde{p}}{\partial r} \right\|_{r=0}^2 + \left\| \tilde{u}_z - 2(1 - r^2) \right\|_{z=z_{min}}^2 \tag{23d}$$

where, Θ is the state of the network weights, Ω is the 2D spatial domain of the problem. Note that since the initial flow field velocities are set to be zero, the development of a flow field during training follows a similar pattern as it would have developed in physical reality. In a sense, the efficacy of training algorithm is to update the velocity field so as to better satisfy continuity and momentum conservation equations at randomly selected space time points in each subsequent training steps.

Another important aspect of the training process is that method of solution should benefit from the physical understanding of its underlying phenomenon. Thus a training procedure is prepared to guide the system to evolve into the solution state like how it would have developed in an analogous physical reality. To this effect, a dynamic weighting scheme is used while calculating the gradients for back propagation. The scheme is so set up that initially the network is trained to satisfy only the inlet and boundary conditions. In subsequent training steps, the weight of continuity residue is raised until it matches with that of boundary conditions.[2] After the flow field has been fully established to represent an inviscid incompressible flow, the weights of the momentum conservation residues is increased in small steps. This introduces the effect of viscosity and pressure gradient to the flow field. The small step size for the weights of PDE residues is effective in bypassing the situation where the high residual magnitude of momentum equations leads the system towards the trivial solution.

[2]Note the weighting for PDE residues should be adequate so that the pressure can evolve simultaneously without affecting the development of velocity fields

Therefore, loss function of the network is formulated as

$$J(\Theta) = \sum_{i=1}^{3} w_i J_i(\Theta) + w_b J_b(\Theta) \tag{24}$$

where, Θ is the state of the network weights. Note that $J_b(\Theta)$, the loss expression capturing the boundary conditions, does not contain zero Dirichlet boundary values since this is enforced through transformation effected in Eqn.22.

5.2. Data Generation

To generate training data, a CAD model of the stenosed artery was made and used to generate an unstructured mesh. The meshed geometry was generated by triangulating the geometry using Gmsh [58], an open source meshing tool. The coordinate data was extracted and the boundary points were appended with non-zero Dirichlet and Neumann boundary conditions. To compile the data set, equal number of points from the domain and boundary regions were randomly selected and shuffled to enforce randomness. The coordinate data was then transformed using Eqn.11 and mean centered.

During the course of our experimentation we observed that the population ratio of various boundary regions and the domain region plays a significant role in the training dynamics. Therefore, attention was given to select a uniform distribution of points in equal ratio from various boundary regions. A diagram showing the scatter of points along with boundary conditions is given in Fig.2.

5.3. Initialization

For simulating flow velocities, it is advantageous to have small values of the order of 10^{-2} as initial predictions. This is beneficial since it will impose control over the variance of various initial residues. If this variance is large, the training naturally becomes biased to solve one of the PDEs or boundary or initial conditions leading to stagnation or oscillation of objective function value during initial training phase.

Furthermore, the PDE residues are observed to be much larger compared to the boundary residues during initial training. This shifts the model spuriously towards the trivial solution of the PDE. On the other hand, selecting the initial weights from a random distribution w $\in$ [-0.01, 0.01] renders the training smooth and following an optimal learning pace.

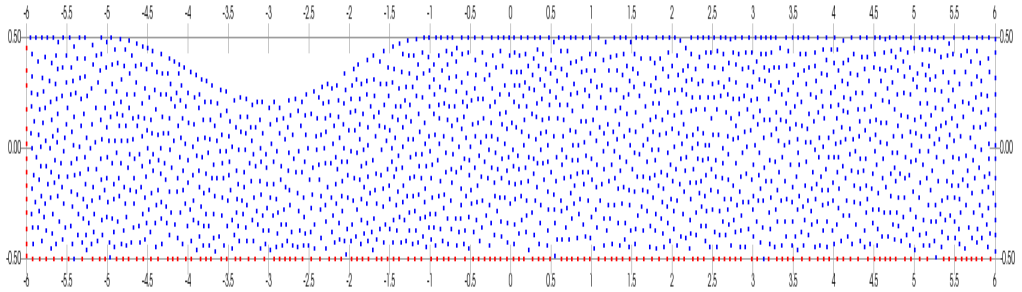

Figure 2: Scatter of a small sample of points generated using Gmsh meshing tool. The points used for non-zero Dirichlet boundary conditions are marked red while points used for calculating PDE residues are marked blue.

5.4. Training

The objective function for training the network as given in Eqn.24 is minimized during training using Adam [39]. The algorithm is set to default implementation in TensorFlow using β_1=0.9 and β_2=0.999. The scheduling of learning rate is handcrafted using details from relevant literature [59, 60] and also by trial and error method. The weights for the loss function terms as mentioned in Eqn.24 are scheduled to effect the argument in 5.1.1. The schedule of all parameters used for obtaining the results presented in this work is as represented by Table.1.

5.5. Monitoring and optimization

During the training, the progress of convergence of the solution can be monitored using the batch averaged value of objective function. This conventional method however can be deceptive when the solution approaches a local minima. An alternative choice was to to visualize the predicted field distribution during course of training. This in fact helps in understanding the training dynamics of the network since the evolution of the network as a physical parameter is closely related to the evolution of the physical system itself. On the other hand, the value of the loss function may provide deceptive information in case the training is progressing towards the trivial solution or a local minima representing an impossible configuration.

5.6. Resources

The model and training sessions are programmed using TensorFlow 1.13 and Python 3.6. The training sessions were carried out on a CPU with

Table 1: Schedule for training with dynamic loss weighting and learning rate

Steps	Learn Rate	W0	W1	W2	W4	batch-size
200	0.001	1	0.1	0.1	1	16
5200	0.001	1	0.1	0.1	1	32
8200	0.0001	1	0.2	0.2	1	32
11200	0.0001	1	0.3	0.3	1	32
14200	0.0001	1	0.4	0.4	1	32
17200	0.0001	1	0.5	0.5	1	32
20200	0.0001	1	0.6	0.6	1	32
23200	0.0001	1	0.7	0.7	1	32
26200	0.0001	1	0.8	0.8	1	32
29200	0.0001	1	0.9	0.9	1	32
32200	0.0001	1	1	1	1	32
132200	1E-05	1	1	1	1	32
>350000	5E-06	1	1	1	1	32

Intel i7 processor supported with 16GB of RAM [38]. Pursuing an optimal data size, training time, speed and accuracy, a dataset of 10,000 points were extracted from each of $\vec{x} \in \Omega$ and $\vec{x} \in \partial\Omega$. The data was fed to the network in mini batch of 32.

5.7. Estimation of parameters

The WSS can be inferred from the flow field solution of the simulation using differential relationship.

$$WSS = \mu(\dot{\gamma}) \frac{\partial u}{\partial y}\bigg|_{wall} \tag{25}$$

The non-dimensionalized WSS can be directly obtained from the non-dimensionalized velocity u_z and u_r as

$$\tau_{wall} = \mu_\infty \frac{\chi}{T} \frac{\partial u_z}{\partial r} \tag{26}$$

Wall pressure can be obtained by forward propagation of the network over a selected number of points on the artery wall. The flow velocity and pressure field values can be deduced from the non-dimensionalized field variables using equations given in Eqn.11.

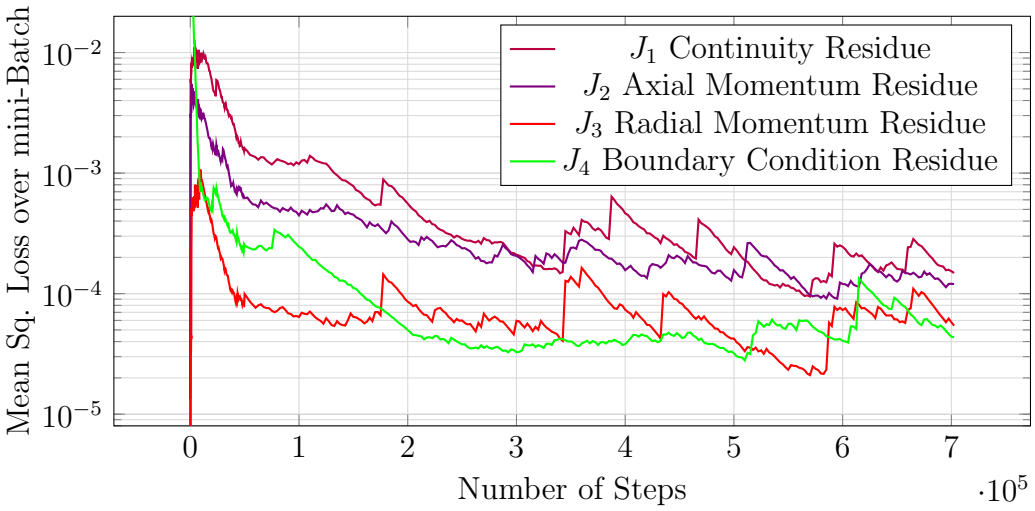

Figure 3: Mean Squared Residuals J_1, J_2, J_3 and J_b over a training batch of 32 as plotted against training steps.

6. Results

Convergence:. The mean of squared residuals over the mini-batch was monitored during the training process to ensure progress in the solution. The mean squared residual takes an initial value close to zero since the network was initialized using small values and increases rapidly in the initial steps before progressively decreasing. The batch averaged residues converged to the order of 10^{-5} after which no further improvement in the solution fields is effected. The highest residue at the time of halting training was below 2×10^{-5}.

The values of mean squared residuals of training batches plotted against training steps is given in Fig.3.

Velocity:. The velocity fields are obtained by visualizing the network function after convergence occurred. The axial and radial velocity fields obtained agrees with the simulation results obtained using CFD as shown in Fig.4 and Fig.5.

Pressure:. The pressure field obtained after convergence is as shown Fig.6. The wall pressure curve obtained by forward propagating the network on select points on the wall also shows satisfactory agreement with that obtained

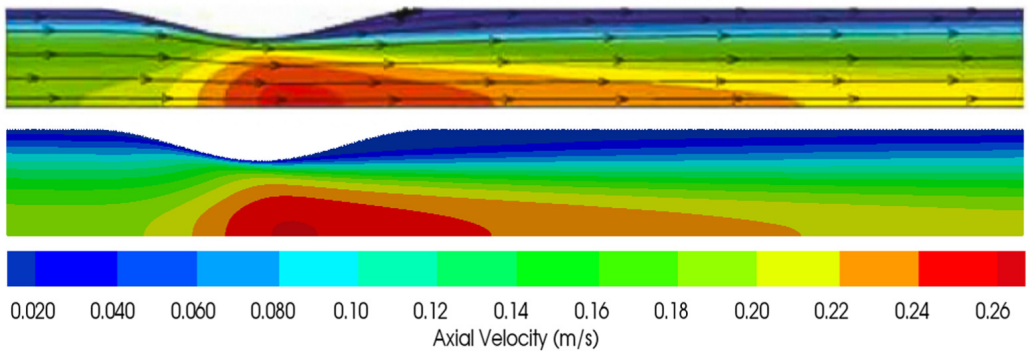

Figure 4: Axial velocity field (m/s) as obtained using CFD simulation(top). Reprinted from the work of Zhou *et al.* [53]; Solution of axial velocity field obtained using DGM (bottom)

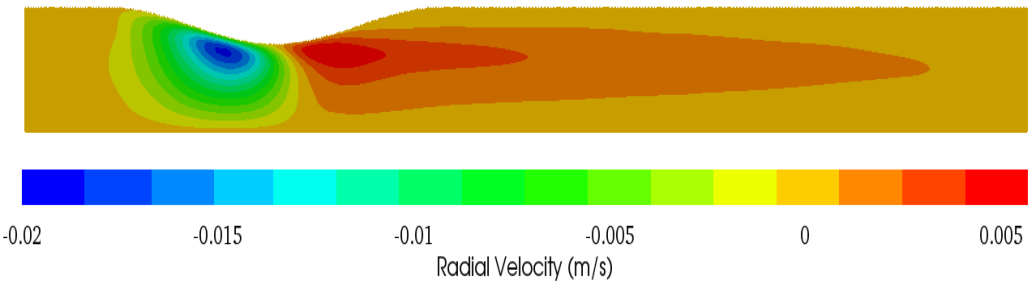

Figure 5: Radial velocity field (m/s) as obtained using DGM.

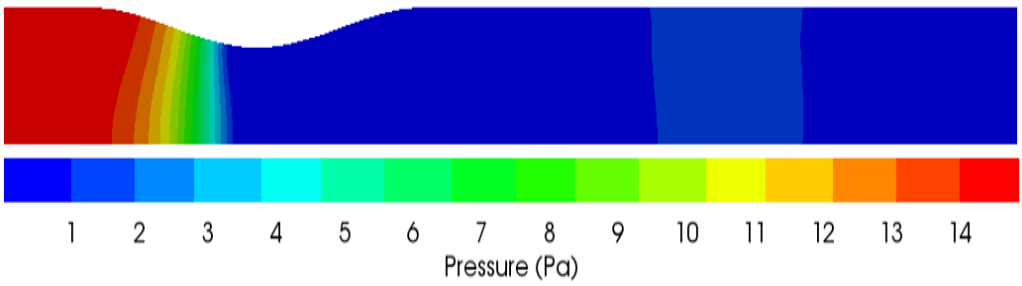

Figure 6: Pressure field inside the stenosed artery obtained using DGM

using CFD solution by Zhou *et al.* [53] over most of the domain. The accumulated error as over the inlet location can can be seen as decreasing at a progressively slower rate as the training progresses. This suggests that the accuracy and convergence property can be increased using a larger model and higher number of hidden units.The wall pressure plot over the length of the artery is obtained as shown in Fig.7.

Wall Shear Stress:. The WSS, an important function in view of the biological effects of blood flow through arteries can easily be predicted by processing the velocity field at points on the wall. The WSS curve obtained along the artery wall also have good agreement with the CFD simulation results in [53] as in Fig.8.

Conclusion

The results obtained by solving the PDEs governing non-Newtonian fluid flow using the Deep Galerkin Method approach is promising. The sum of Root Mean Square of the residuals of continuity, axial and radial momentum conservation equations – J_1, J_2 and J_3 respectively – plotted after convergence over 1000 uniformly distributed points yields a value of the order of 10^{-3}. The axial velocity field predicted agrees very well with CFD solution while the wall pressure and WSS distribution obtained agrees satisfactorily. Accuracy of the these solutions suggests adequacy of the training method.

The caveat is that the results of initial attempts using DGM approach is competed with that from tested and highly optimized commercial solvers. The accuracy of the DGM solution obtained in this work is limited by the

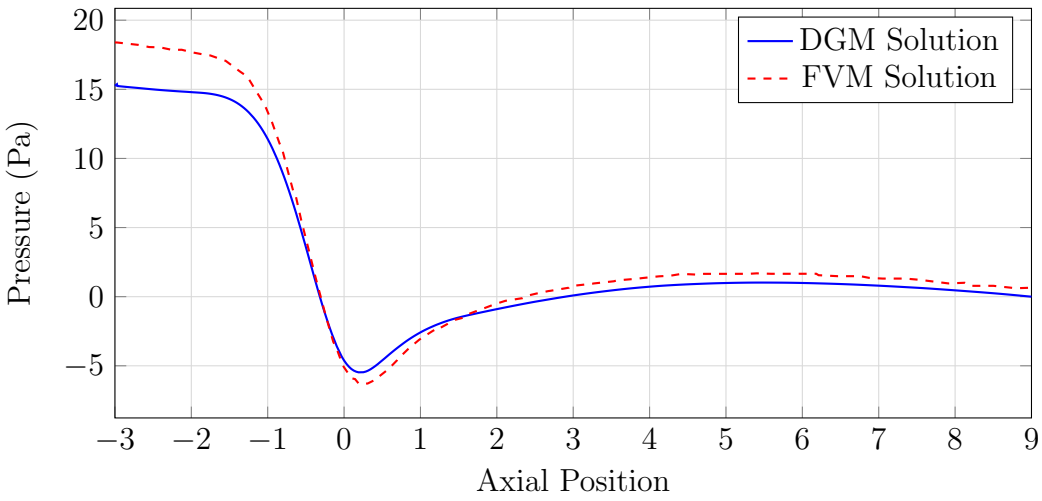

Figure 7: Wall Pressure obtained using DGM and CFD

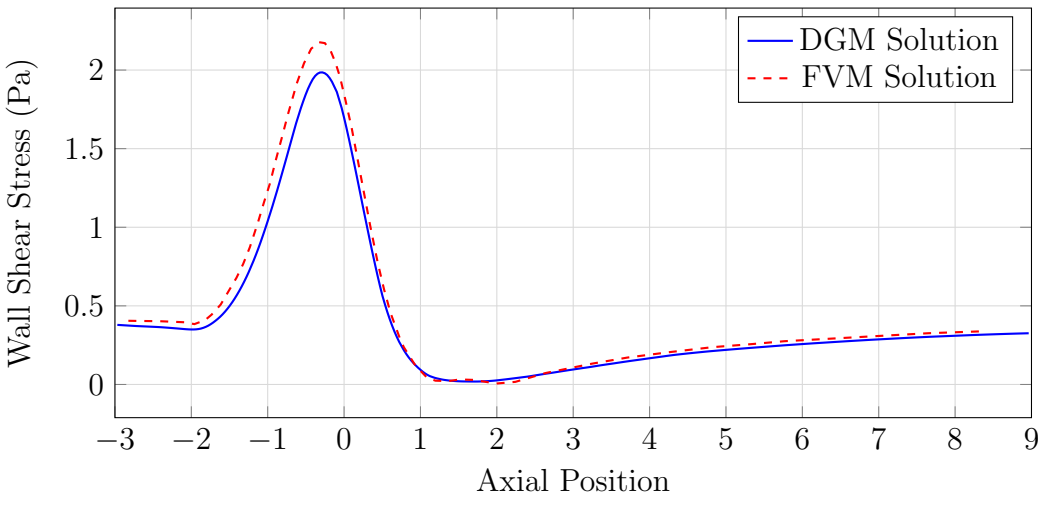

Figure 8: Wall Shear Stress obtained using DGM and CFD

paucity of resources committed. The convergence plot in Fig.3 suggests that a training session with higher batch size and larger dataset can continue to better approximate the solution. Moreover, a larger DGM network with significantly high number of hidden units and layers can be better approximators of the solution. The method also obviates the rigorous procedures for discretizing the equations and solving on a computational grid and is simpler to be implemented than conventional numerical schemes. The Neural Network approach employed in this work can therefore be employed in cases of complex fluids where the system of equation are complex or highly nonlinear.

Future Work

The spatio-temporal version of the network can be employed to attempt the solution of an unsteady and oscillating shear flow of blood though this will require much larger dataset, network size and computational power. Also, the Carreau model simulation is representative of the efficacy of the machine learning approach which can be employed to simulate more complicated fluid models. These can be the directions of a future work.

References

[1] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (2015) 436. doi:10.1038/nature14539.

[2] Z. Q. Zhao, P. Zheng, S. Xu, X. Wu, Object detection with deep learning: A review, IEEE transactions on neural networks and learning systems (2019).

[3] P. Mamoshina, A. Vieira, E. Putin, A. Zhavoronkov, Applications of deep learning in biomedicine, Molecular pharmaceutics 13 (5) (2016) 1445–1454.

[4] Y. Liu, The review of intelligent mechanical engineering based on artificial neural network, in: International Conference on Intelligent Systems Research and Mechatronics Engineering, Atlantis Press, 2015 (2015). doi:10.2991/isrme-15.2015.405.

[5] Z. Liu, C. Wu, M. Koishi, A deep material network for multiscale topology learning and accelerated nonlinear modeling of heterogeneous materials, Computer Methods in Applied Mechanics and Engineering 345 (2019) 1138–1168. doi:10.1016/j.cma.2018.09.020.

[6] J. Pan, Y. Zi, J. Chen, Z. Zhou, B. Wang, Liftingnet: A novel deep learning network with layerwise feature learning from noisy mechanical data for fault classification, IEEE Transactions on Industrial Electronics 65 (6) (2018) 4973–4982. doi:10.1109/TIE.2017.2767540.

[7] O. Faust, Y. Hagiwara, T. J. Hong, O. S. Lih, U. R. Acharya, Deep learning for healthcare applications based on physiological signals: A review, Computer methods and programs in biomedicine 161 (2018) 1–13.

[8] I. E. Lagaris, A. Likas, D. I. Fotiadis, Artificial neural networks for solving ordinary and partial differential equations, IEEE transactions on neural networks 9 (5) (1998) 987–1000.

[9] I. E. Lagaris, A. C. Likas, D. G. Papageorgiou, Neural-network methods for boundary value problems with irregular boundaries, IEEE Transactions on Neural Networks 11 (5) (2000) 1041–1049. doi:10.1109/72.870037.

[10] Y. M. A. Hashash, S. Jung, J. Ghaboussi, Numerical implementation of a neural network based material model in finite element analysis, International Journal for Numerical Methods in Engineering 59 (7) (2004) 989–1005. doi:10.1002/nme.905.

[11] A. Barkaoui, B. Tlili, A. Vercher-Martínez, R. Hambli, A multiscale modelling of bone ultrastructure elastic proprieties using finite elements simulation and neural network method, Computer Methods and Programs in Biomedicine 134 (2016) 69–78. doi:10.1016/j.cmpb.2016.07.005.

[12] N. KılıÇ, B. Ekici, S. Hartomacıoğlu, Determination of penetration depth at high velocity impact using finite element method and artificial neural network tools, Defence Technology 11 (2) (2015) 110–122. doi:10.1016/j.dt.2014.12.001.

[13] K. Rudd, S. Ferrari, A constrained integration (cint) approach to solving partial differential equations using artificial neural networks, Neurocomputing 155 (2015) 277–285. doi:10.1016/j.neucom.2014.11.058.

[14] M. Stoffel, F. Bamer, B. Markert, Artificial neural networks and intelligent finite elements in non-linear structural mechanics, Thin-Walled Structures 131 (2018) 102–106. doi:10.1016/j.tws.2018.06.035.

[15] X. Yu, L. Deng, X. Zhang, M. Chen, F. Kuang, Y. Wang, Accurate numerical computation of hot deformation behaviors by integrating finite element method with artificial neural network, International Journal of Precision Engineering and Manufacturing 19 (3) (2018) 395–404. doi:10.1007/s12541-018-0047-6.

[16] S. Shahane, N. R. Aluru, P. Ferreira, S. G. Kapoor, S. P. Vanka, Genetic algorithm based multi-objective optimization of solidification in die casting using deep neural network as surrogate model, CoRR abs/1901.02364 (2019).

[17] L. Liang, M. Liu, C. Martin, W. Sun, A deep learning approach to estimate stress distribution: a fast and accurate surrogate of finite-element analysis, Journal of The Royal Society Interface 15 (2018). doi:10.1098/rsif.2017.0844.

[18] C. W. Chang, N. T. Dinh, Classification of machine learning frameworks for data-driven thermal fluid models, International Journal of Thermal Sciences 135 (2019) 559–579.

[19] J. Berg, K. Nyström, A unified deep artificial neural network approach to partial differential equations in complex geometries, Neurocomputing 317 (2018) 28–41. doi:10.1016/j.neucom.2018.06.056.

[20] M. Baymani, A. Kerayechian, S. Effati, Artificial neural networks approach for solving stokes problem, Applied Mathematics 1 (04) (2010) 288.

[21] M. Chiaramonte, M. Kiener, Solving differential equations using neural networks, Machine Learning Project (2013).

[22] M. F. McCracken, Artificial neural networks in fluid dynamics: A novel approach to the navier-stokes equations, in: Proceedings

of the Practice and Experience on Advanced Research Computing, PEARC '18, ACM, New York, NY, USA, 2018, pp. 80:1–80:4 (2018). doi:10.1145/3219104.3229262.

[23] M. Raissi, A. Yazdani, G. E. Karniadakis, Hidden fluid mechanics: A navier-stokes informed deep learning framework for assimilating flow visualization data, arXiv preprint arXiv:1808.04327 (2018).

[24] G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of Control, Signals and Systems 2 (4) (1989) 303–314. doi:10.1007/BF02551274.

[25] H. N. Mhaskar, Neural networks for optimal approximation of smooth and analytic functions, Neural Computation 8 (1) (1996) 164–177. doi:10.1162/neco.1996.8.1.164.

[26] B. Hanin, Universal function approximation by deep neural nets with bounded width and relu activations, arXiv preprint arXiv:1708.02691 (2017).

[27] U. Shaham, A. Cloninger, R. R. Coifman, Provable approximation properties for deep neural networks, Applied and Computational Harmonic Analysis 44 (3) (2018) 537–557. doi:10.1016/j.acha.2016.04.003.

[28] J. H. Ferziger, M. Peric, Computational Methods for Fluid Dynamics, Springer Science & Business Media, 2012.

[29] O. C. Zienkiewicz, R. L. Taylor, P. Nithiarasu, J. Zhu, The Finite Element Method, Vol. 3, McGraw-hill London, 1977.

[30] P. Wriggers, Nonlinear finite element methods, Springer Science & Business Media, 2008.

[31] R. Fåhræus, T. Lindqvist, The viscosity of the blood in narrow capillary tubes, American Journal of Physiology-Legacy Content 96 (3) (1931) 562–568. doi:10.1152/ajplegacy.1931.96.3.562.

[32] J. Stoltz, S. Gaillard, F. Paulus, O. Henri, P. Dixneuf, Experimental approach to rouleau formation. comparison of three methods, Biorheology 23 (s1) (1984) 221–226.

[33] M. Menut, L. Boussel, X. Escriva, B. Bou-Saïd, H. W. L. Berre, Y. Marchesse, A. Millon, N. D. Schiava, P. Lermusiaux, J. Tichy, Comparison between a generalized newtonian model and a network-type multiscale model for hemodynamic behavior in the aortic arch: Validation with 4d mri data for a case study, Journal of Biomechanics 73 (2018) 119–126. doi:10.1016/j.jbiomech.2018.03.038.

[34] A. Apostolidis, M. Armstrong, A. Beris, Modeling of human blood rheology in transient shear flows, Journal of Rheology 59 (2015) 275–298. doi:10.1122/1.4904423.

[35] N. S. Akbar, S. Nadeem, K. Mekheimer, Rheological properties of reiner-rivlin fluid model for blood flow through tapered artery with stenosis, Journal of the Egyptian Mathematical Society 24 (1) (2016) 138–142. doi:10.1016/j.joems.2014.10.007.

[36] J. Sirignano, K. Spiliopoulos, Dgm: A deep learning algorithm for solving partial differential equations, Journal of Computational Physics 375 (2018) 1339–1364. doi:10.1016/j.jcp.2018.08.029.

[37] J. Han, A. Jentzen, W. E, Solving high-dimensional partial differential equations using deep learning, Proceedings of the National Academy of Sciences 115 (34) (2018) 8505–8510. doi:10.1073/pnas.1718942115.

[38] A. Al-Aradi, A. Correia, D. Naiff, G. Jardim, Y. Saporito, Solving nonlinear and high-dimensional partial differential equations via deep learning, arXiv preprint arXiv:1811.08782 (2018).

[39] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).

[40] P. K. Mandal, An unsteady analysis of non-newtonian blood flow through tapered arteries with a stenosis, International Journal of Non-Linear Mechanics 40 (1) (2005) 151–164.

[41] T. J. Pedley, The Fluid Mechanics of Large Blood Vessels, Cambridge Monographs on Mechanics, Cambridge University Press, 1980. doi:10.1017/CBO9780511896996.

[42] D. N. Ku, Blood flow in arteries, Annual review of fluid mechanics 29 (1) (1997) 399–434.

[43] S. S. Shibeshi, W. E. Collins, The rheology of blood flow in a branched arterial system, Applied rheology (Lappersdorf, Germany: Online) 15 (6) (2005) 398.

[44] Y. I. Cho, K. R. Kensey, Effects of the non-newtonian viscosity of blood on flows in a diseased arterial vessel. part 1: Steady flows, Biorheology 28 (3-4) (1991) 241–262.

[45] P. Ballyk, D. Steinman, C. Ethier, Simulation of non-newtonian blood flow in an end-to-side anastomosis, Biorheology 31 (5) (1994) 565–586.

[46] F. J. Walburn, D. J. Schneck, A constitutive equation for whole human blood, Biorheology 13 (3) (1976) 201–210.

[47] J. Fang, R. G. Owens, Numerical simulations of pulsatile blood flow using a new constitutive model, Biorheology 43 (5) (2006) 637–660.

[48] M. Moyers-Gonzalez, R. G. Owens, J. Fang, A non-homogeneous constitutive model for human blood. part 1. model derivation and steady flow, Journal of Fluid Mechanics 617 (2008) 327–354.

[49] R. G. Owens, A new microstructure-based constitutive model for human blood, Journal of non-newtonian fluid mechanics 140 (1-3) (2006) 57–70.

[50] W. H. Herschel, R. Bulkley, Konsistenzmessungen von gummi-benzollösungen, Colloid & Polymer Science 39 (4) (1926) 291–300.

[51] Q. H. Nguyen, N. D. Nguyen, Incompressible non-newtonian fluid flows, in: Continuum Mechanics-Progress in fundamentals and Engineering applications, IntechOpen, 2012 (2012).

[52] B. M. Johnston, P. R. Johnston, S. Corney, D. Kilpatrick, Non-newtonian blood flow in human right coronary arteries: steady state simulations, Journal of Biomechanics 37 (5) (2004) 709–720. doi:10.1016/j.jbiomech.2003.09.016.

[53] Y. Zhou, C. Lee, J. Wang, The computational fluid dynamics analyses on hemodynamic characteristics in stenosed arterial models, Journal of healthcare engineering (2018).

[54] G. Pontrelli, Blood flow through an axisymmetric stenosis, Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine 215 (1) (2001) 1–10.

[55] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, arXiv preprint arXiv:1502.03167 (2015).

[56] S. J. Nowlan, G. E. Hinton, Simplifying neural networks by soft weight-sharing, Neural Computation 4 (4) (1992) 473–493. doi:10.1162/neco.1992.4.4.473.

[57] P. Ramachandran, B. Zoph, Q. V. Le, Searching for activation functions, arXiv preprint arXiv:1710.05941 (2017).

[58] C. Geuzaine, J. F. Remacle, Gmsh: A 3-d finite element mesh generator with built-in pre- and post-processing facilities, International Journal for Numerical Methods in Engineering 79 (2009) 1309–1331. doi:10.1002/nme.2579.

[59] W. Zaremba, I. Sutskever, O. Vinyals, Recurrent neural network regularization, arXiv preprint arXiv:1409.2329 (2014).

[60] Y. Gal, Z. Ghahramani, A theoretically grounded application of dropout in recurrent neural networks, in: Advances in neural information processing systems, 2016, pp. 1019–1027 (2016).

LaTeX Source Files

Click here to download LaTeX Source Files: latex_source_files.7z