

IMECE2022-93668

PREDICTING MOTION OF ENGINE-INGESTED PARTICLES USING DEEP NEURAL NETWORKS

Travis Bowman

Graduate Student

Mechanical Engineering Department
Virginia Tech

Blacksburg, Virginia 24061

Email: travis9@vt.edu

Cairen Miranda

Graduate Student

Mechanical Engineering Department
Virginia Tech

Blacksburg, Virginia, 24061

Email: cairenjm@vt.edu

John Palmore Jr.*

Assistant Professor

Mechanical Engineering Department
Virginia Tech

Blacksburg, Virginia, 24061

Email: palmore@vt.edu

ABSTRACT

The ultimate goal of this work is to facilitate the design of gas turbine engine particle separators by reducing the computational expense to accurately simulate the fluid flow and particle motion inside the separator. It has been well-documented that particle ingestion yields many detrimental impacts for gas turbine engines. This ingestion is of concern for operation in environments where dust, ash, or ice persist. The consequences of ice particle ingestion can range from surface-wear abrasion to engine power loss. Ice particles are chosen for this study because of their relevance to civil aviation. It is known that sufficiently small particles, characterized by small particle response times (τ_p), closely follow the fluid trajectory whereas large particles deviate from the streamlines. The behavior of small particles hints at a method for larger particle trajectories because the higher order terms (HOT) in the asymptotic particle acceleration solution can be shown to be $O(\tau_p)$. By explicitly considering τ_p , these HOT can be derived. Rather than manually deriving these terms, this work chooses to implicitly derive them using machine learning (ML). Inertial particle separators are devices designed to remove particles from the engine intake flow. Particle separators contribute to both elongating the lifespan and promoting safer operation of aviation gas turbine engines. Complex flows, such as flow through a particle separator, naturally have rotation and strain present throughout the flow field. This study attempts to understand if the motion of particles within ro-

tational and strained canonical flows can be accurately predicted using supervised ML. This report suggests that preprocessing the ML training data to the fluid streamline coordinates can improve model training. Furthermore, this work provides some guidelines for applying ML, particularly deep feed-forward neural networks, with physics driven multiphase flow data. Additionally, the ML model is able to predict the particle accelerations in the fully rotational and irrotational canonical laminar flows quite well. For combined flows with rotation and strain, however, the model struggles to predict the particle accelerations.

NOMENCLATURE

$\mathbf{v}$	Particle velocity
$\mathbf{u}$	Fluid velocity
μ_g	Fluid dynamic viscosity
ρ_p	Particle density
ρ_g	Gas density
d_p	Particle diameter
τ_p	Particle response time
τ_f	Flow timescale
c_D	Drag coefficient
Re	Reynolds number
St	Stokes number
Φ	Particle volume fraction
N	Sample size
lr	Learning rate

*Corresponding author. ASME Member.

- μ Momentum parameter
- A Strain rate
- e Shear rate
- ω Rotation rate

1 INTRODUCTION

Ice has plagued aircraft operation since the birth of aviation. Ice accretion imposes the immediate threat of accumulated ice changing the aerodynamic properties of the wing airfoil, in the worst case leading to loss of sufficient lift. The majority of current research deals with ice accretion and means to mitigate or prevent this phenomena. In this work, however, the focus is turned to ice that has been ingested by the gas turbine engine through means of either ice existing in the atmosphere or pieces of accretion that have broken off of aircraft surfaces. It has been well-documented that particle ingestion can result in many detrimental impacts for gas turbine engines [1]. For ice particles, the consequences can range from surface-wear abrasion to loss of power. Ice that passes through the fan stage enters the compressor, where it likely melts and can evaporate. This formation of liquid water or vapor alters the properties of the flow from the engine's design point. As the compressor is very sensitive to such changes, surge can occur as a result [2]. While it is rare for particles to reach the combustion chamber, in the case that enough ice particulate does, the engine can experience flame-out [2, 3]. High altitude ice crystals in convective systems of concern to commercial aviation generally range from diameters of $100\mu\text{m}$ to 1mm [4]. However, hailstones often exist in the range of 0.5cm to 2.5cm [5]. Therefore, we consider ice particles over a wide range of sizes in this study.

1.1 Particle Motion Theory

It is known that the motion of particles suspended in a fluid can be characterized by the non-dimensional Stokes number, St , defined by Eq.(1), where τ_p is the particle relaxation or response time defined by Eq.(2) and τ_f is the appropriate timescale of the flow [6]. Physically, τ_p can be thought of as the time that it takes for the particle to respond to changes in the gas flow. Additionally, as expressed in Eq.(2), ρ_p and d_p are the particle density and diameter respectively, and μ_g is the dynamic viscosity of the gas [6].

$$St = \frac{\tau_p}{\tau_f} \quad (1)$$

$$\tau_p = \frac{\rho_p d_p^2}{18\mu_g} \quad (2)$$

The governing equation for a particle moving through gas, neglecting gravity, can be shown to be:

$$\frac{d\mathbf{v}}{dt} = -f \frac{\mathbf{v} - \mathbf{u}}{\tau_p} \quad (3)$$

where $\mathbf{v}$ is the particle velocity, $\mathbf{u}$ is the local gas phase velocity, and the function f introduces a nonlinear drag forcing dependence on the velocity difference $\mathbf{v} - \mathbf{u}$. For very small Stokes numbers ($St \ll 1$), the particle motion very closely follows the fluid trajectory, while large particles ($St \gg 1$) act in a more ballistic manner and are not especially driven by the flow streamlines [7]. The goal of this work is to predict particle trajectories using artificial neural networks (ANNs). The approach taken in this study is inspired by the asymptotic solution found by Maxey for the zero-gravity case [8]

$$\mathbf{v} = \mathbf{u} - \tau_p \left(\frac{D\mathbf{u}}{Dt} \right) + O(\tau_p^2) \quad (4)$$

where D/Dt is the material derivative. When Eq.(4) is substituted back into Eq.(3), the resulting expression gives

$$\frac{d\mathbf{v}}{dt} = \frac{D\mathbf{u}}{Dt} + O(\tau_p) \quad (5)$$

which confirms that the behavior of sufficiently small particles (small τ_p) closely follows that of the streamlines. However, Eq.(5) also hints at a strategy for larger particles. By considering τ_p explicitly, higher order terms in the asymptotic expansion can be derived. Rather than manually deriving these terms, they are implicitly derived using machine learning (ML). To achieve this objective, simulations of particles in steady laminar flow fields are performed in the scientific Python environment (*SciPy*), which is discussed in more detail in the Methods section. Particles of various diameters corresponding to Stokes numbers from 0.05 - 500 will be considered to capture the influence of St . These simulations are used to extract relevant particle data to train and validate a neural network for the various flow configurations and combinations thereof.

1.2 Machine Learning

Using the open source ML library *PyTorch*, multiple ANNs are trained using data generated from the canonical flows and combinations thereof. *PyTorch* is a widely deployed open-source ML framework with a strong research support community. The idea of using ANNs to compute multiphase particle data is not

a new one. For instance, Minakowska et al. [9] developed deep ANNs capable of predicting hydrodynamic forces on non-spherical particles. Raissi et al. [10] introduce a deep learning framework for data-sparse scenarios that combine data with the governing equation models, termed physics informed neural networks (PINNs). Seyed-Ahmadi and Wachs [11] apply this idea to model forces and torques in particle-laden flows with particle volume fractions (Φ) on the order of 0.1. Although traditionally used for computer vision, current research is showing how ML can be used with strictly data-based applications. A long term goal of this research is to use deep ANNs as cheap surrogates in particle separator design. Because simulations are computationally expensive and experimental studies of particle separators are costly, the deep ANN surrogates could expedite the process of designing inertial particle separators. This introductory work trains ANNs with well-defined, redundant data sets from simple flows to explore how ANNs perform in scenarios with simple combined laminar flows as well as complex flows using flow feature extraction.

2 METHODS

The scope of particle motion studies conducted assume that the particle volume fraction is small enough such that the dispersed particles do not affect the fluid physics; this is generally referred to as one-way coupled flow. Additionally, the particles considered are assumed to be perfectly spherical ice particles with an isotropic density of $\rho_p = 917 \text{ kg/m}^3$ [12]. Particle rotation is not considered, and gravitational forces are also neglected because only 2D cases are explored. The governing particle dynamics equations are shown in Eq.(6) and Eq.(7) where $\mathbf{x}_p$ and m_p are the particle position and mass, respectively. The particle mass is computed as $m_p = \rho_p \pi d_p^3 / 6$. The drag force, f_D , is given by Eq.(8) where the relative velocity is given by $\mathbf{v}_r = \mathbf{u} - \mathbf{v}$ and $A_p = \pi d_p^2 / 4$ is the projected particle area normal to the flow. The drag coefficient, c_D , is taken to be the Schiller-Naumann relation given in Eq.(9) [13]. The density and dynamic viscosity of air are taken to be $\rho_g = 1.20 \text{ kg/m}^3$ and $\mu_g = 1.51 \text{ E-}5 \text{ kg/(ms)}$, respectively.

$$\frac{d\mathbf{x}_p}{dt} = \mathbf{v} \quad (6)$$

$$m_p \frac{d\mathbf{v}}{dt} = \mathbf{f}_D \quad (7)$$

$$\mathbf{f}_D = \frac{1}{2} c_D \rho_g |\mathbf{v}_r|^2 A_p \frac{\mathbf{v}_r}{|\mathbf{v}_r|} \quad (8)$$

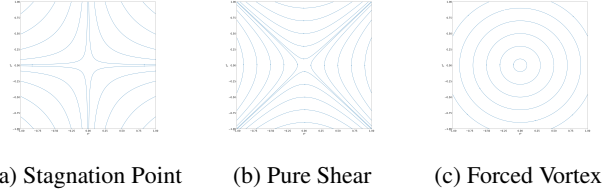


FIGURE 1: STREAMLINES OF THE CANONICAL FLOWS USED FOR GENERATING ML DATA

$$c_D = \frac{24}{Re} (1 + 0.15 Re^{0.687}) \quad (9)$$

Additionally, the Reynolds number was taken to be

$$Re = \frac{\rho_g |\mathbf{v}_r| d_p}{\mu_g} \quad (10)$$

Following [7], the motion of particles in the following steady, laminar canonical flows are used to assess the feasibility of using ANNs: stagnation point, pure (irrotational) shear, and forced (rotational) vortex. Stagnation point flow is an irrotational flow characterized by the strain rate, A , in the following 2D velocity field: $\mathbf{u} = [Ax_1, -Ax_2]$. Pure shear, similarly, is an irrotational flow given by $\mathbf{u} = [ex_2, ex_1]$ where e is the shear rate. Lastly, the forced vortex is a purely rotational flow characterized by the rotation rate, ω , in $\mathbf{u} = [-\omega x_2, \omega x_1]$. Figures 1a, 1b, and 1c show the velocity streamlines of the stagnation point, pure shear, and forced vortex for $A = e = \omega = 1.0 \text{ s}^{-1}$.

The velocity gradient tensor given by $\nabla \mathbf{u}$, can be written as the sum of a symmetric and an anti-symmetric tensor given by the strain rate tensor $\mathbf{S}$ and rotation tensor $\mathbf{\Omega}$, respectively. These tensors are shown in Eq.(11) and Eq.(12).

$$\mathbf{S} = \frac{1}{2} (\nabla \mathbf{u} + \nabla \mathbf{u}^T) \quad (11)$$

$$\mathbf{\Omega} = \frac{1}{2} (\nabla \mathbf{u} - \nabla \mathbf{u}^T) \quad (12)$$

We use the second principal invariant of $\nabla \mathbf{u}$, defined in Eq.(13), as an objective measure of the strength of the flows.

$$I_2(\nabla \mathbf{u}) = \frac{1}{2} ((tr(\nabla \mathbf{u}))^2 - tr(\nabla \mathbf{u}^2)) \quad (13)$$

For an incompressible flow, Eq.(13) can be shown to simplify as $I_2(\nabla \mathbf{u}) = -\frac{1}{2} (\nabla \mathbf{u}^T : \nabla \mathbf{u})$. Note that the convention for the inner

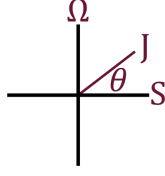


FIGURE 2: ROTATION AND SHEAR COMPONENT DECOMPOSITION

product is taken from [14] as shown in Eq.(14)

$$A : B = A_{ij}B_{ji} \quad (14)$$

where Einstein's summation convention is used. For the stagnation point, pure shear, and forced vortex flow, $I_2(\nabla \mathbf{u})$ can be shown to simplify as A^2 , e^2 , and ω^2 , respectively. The second invariant of $\mathbf{S}$ and $\mathbf{\Omega}$ are also considered, defined as $\frac{1}{2}(\mathbf{S}^T : \mathbf{S})$ and $\frac{1}{2}(\mathbf{\Omega}^T : \mathbf{\Omega})$ respectively. To simplify its use as a flow timescale, we define $J = \sqrt{|I_2(\nabla \mathbf{u})|}$. Similarly, $S = \sqrt{|I_2(\mathbf{S})|}$ and $\Omega = \sqrt{|I_2(\mathbf{\Omega})|}$ are defined as metrics to describe the relative contributions of fluid strain and rotation at a given particle location. Not surprisingly, $\Omega = 0$, $S = A$, $J = A$ and $\Omega = 0$, $S = e$, $J = e$ for the stagnation point and irrotational shear flows, respectively. Similarly, $\Omega = \omega$, $S = 0$, $J = \omega$ for the forced vortex case. For this reason, the directly comparable flow parameters A , e , and ω are taken to be $\frac{1}{\tau_f}$ for each canonical flow case. It can be shown that for a general incompressible flow, $J = \sqrt{S^2 + \Omega^2}$. Therefore, this work asserts that the decomposition of a complex flow into its rotational and sheared components can be thought of as the classical unit circle as suggested in Figure 2. To generate the ANN training data, the governing equations (Eq.(6) and Eq.(7)) of the particle dynamics are solved using numerical integration techniques in the scientific Python computing environment (*SciPy*). The particle initial conditions are taken as $\mathbf{v}(t_0) = [0, 0]$ and $\mathbf{x}_p(t_0) = [x_{1,0}, x_{2,0}]$, where the initial positions $x_{1,0}$ and $x_{2,0}$ are randomly selected. However, $x_{1,0}$ and $x_{2,0}$ are both restricted to initialize in range $[-1, 1]$. Specifically, because canonical flow systems can present stiff systems of differential equations for the particle trajectory, the LSODA method was employed [15]. Some brief testing was performed with stiff ODE solvers in *SciPy* including LSODA, Radau, and backwards differentiation. It was found that LSODA was more computationally efficient for this problem, and hence it was selected for the study. Furthermore, to non-dimensionalize the model inputs the governing equations are modified as

$$\frac{d\mathbf{v}^*}{dt^*} = \frac{1}{2} \frac{\rho_g}{\rho_p} \frac{L_{nd} A_p^* c_D^* \|\mathbf{v}_r^*\|^2}{V_p^*} \frac{\mathbf{v}_r^*}{\|\mathbf{v}_r^*\|} \quad (15)$$

$$\frac{d\mathbf{x}_p^*}{dt^*} = \mathbf{v}^* \quad (16)$$

$$c_D^* = \frac{24}{Re_p^*} (1 + 0.15(Re_p^*)^{0.687}) \quad (17)$$

$$Re_p^* = \frac{\mathbf{v}_r^* d_p^*}{\nu^*} \quad (18)$$

where the dynamic viscosity $\nu = \mu_g/\rho_g$ and the asterisk indicates a non-dimensionalized parameter (e.g. $d_p^* = \frac{d_p}{L_{nd}}$). The natural choice for the timescale for which to non-dimensionalize with is $\frac{1}{J}$, and the length scales are non-dimensionalized with $L_{nd} = 1.0 \text{ cm}$ based on the order of magnitude that hailstones normally exist at [5]. The relevant equations are solved in non-dimensional form using $L_{nd} = 1.0 \text{ cm}$ and $\frac{1}{J}$ as the characteristic values. Five different particle sizes corresponding to $St = 5 * [10^{-2}, 10^{-1}, 10^0, 10^1, 10^2]$ are considered in this study. It should be noted that the ice particle diameters corresponding to these St range from $122 \mu\text{m}$ to 1.22 cm . This range includes small atmospheric ice crystals that exist on the order of 100s of μm [4] through hailstones that exist on the order of 1 cm [5]. The relevant parameters are extracted for 5,000 particles of each St at $t^* = 1.0$ ($t^* = tJ$), and 80% of the data are used for training the ANNs and the remaining 20% are used for validation.

2.1 Neural Network Structure

The overall design space of deep feedforward neural networks is far too vast to be completely searched. Therefore, multiple design of experiments strategies are employed. First, ANNs arranged in 2-6 hidden layer structures are compared and a local optimum for the number of hidden layers (HL) is determined. The total number of hidden neurons is conservatively determined by the learning capability relations proven by Huang [16] for a two HL feed forward ANN as shown in Eq.(19) and Eq.(20),

$$L_1 = \sqrt{(m+2)N} + 2\sqrt{\frac{N}{m+2}} \quad (19)$$

$$L_2 = m\sqrt{\frac{N}{m+2}} \quad (20)$$

where N is the number of distinct training samples and m is the number of output neurons. The configurations detailed in Table

TABLE 1: HIDDEN LAYER CONFIGURATIONS FOR SINGLE FLOW ANNS

Number H.L.	L_1	L_2	L_3	L_4	L_5	L_6
2	425	142	-			
3	364	145	58	-		
4	349	140	56	22	-	
5	343	138	55	22	9	-
6	341	136	55	22	9	4

1 are used to evaluate the effect of the number of HLs. Note that the total number of neurons in each network and the decreasing ratio (approximately 2.5:1) of neurons in the HLs are both kept constant. The ratio of $\frac{L_n}{L_{n+1}} \approx 2.5$ was determined by directly calculating the ratio from Huang's expressions and considering the results of a brute-force optimization of a two layer neural network presented by Heaton [17]. Using Huang's equations, the ratio of Eq.(19) to Eq.(20) can be shown to reduce to $1 + \frac{4}{m}$ and yields $\frac{L_1}{L_2} = 3$ for this application. The optimal configuration that Heaton found for a two HL ANN had a ratio closer to 2:1 [17]. Proceeding with the observed optimal structure, several activation functions such as ReLU, tanh, and Softmax are compared to determine an optimal function for this application. Additionally, each activation function considered is tested with and without Xavier initialization [18]. Finally, using the combination of the best structure and activation function, various data preprocessing techniques are employed. These preprocessing methods are discussed in more detail later.

2.2 Optimization Algorithms

Various optimization algorithms standard in machine learning are tested to ensure that ANN training is being done efficiently. The classical stochastic gradient descent (SGD) optimizer is used as a baseline comparison for other common optimization algorithms such as Adam [19], Adagrad [20], Adadelata [21], RMSProp [22], and SGD with Nesterov momentum [23].

2.3 Training Data Preprocessing

Although not a main objective, this work does consider high level ANN hyperparameter tuning and the results presented in this report should be used as a guide for similar ML applications. The ANN structure is analyzed, along with the activation function and data preprocessing. After the structure is explored by testing five different HL configurations, several data representations are studied using the best ANN structure found. For each flow type considered, two main configurations of the *PyTorch*

ANN will be tested. The first will be a data-driven model, in which each of the relevant parameters are given directly to the ANN to compute the particle acceleration, $\frac{d\mathbf{v}}{dt}$, or a mapping of the form:

$$\frac{d\mathbf{v}}{dt} = g\left(\mathbf{v}, \mathbf{u}, d_p, \frac{D\mathbf{u}}{Dt}, \rho_p, \rho_g, \mu_g\right) \quad (21)$$

This configuration's performance will be compared with a physics-driven method, where many of the above variables are condensed into parameters with more physical significance, as the following mapping shows:

$$\frac{d\mathbf{v}}{dt} = g\left(\mathbf{v} - \mathbf{u}, \tau_p, \frac{D\mathbf{u}}{Dt}\right) \quad (22)$$

It is expected that the ANN trained with physics-driven preprocessed (PDP) data will perform just as well or better than the data-driven (DD) method because it contains essentially the same information, albeit with less data which is advantageous. In addition to testing the physics-informed approach against the data-driven method in the default Cartesian coordinates of the simulation, ANNs are also trained using the same data with a transformation of coordinates to streamline and path coordinates as shown in Figure 3. The solid line in Figure 3 represents a fluid streamline and the dashed line represents a particle path. The streamline coordinates are represented by $\hat{s}$ - $\hat{n}_s$ and the path coordinates are given by $\hat{t}$ - $\hat{n}_t$. $\hat{s}$ and $\hat{t}$ are tangent to the fluid streamline and particle path (i.e. the directions of $\mathbf{u}$ and $\mathbf{v}$), respectively and $\hat{n}_s$, $\hat{n}_t$ are the corresponding normal vectors. That is, $\hat{s} = \frac{\mathbf{u}}{\|\mathbf{u}\|}$ and $\hat{t} = \frac{\mathbf{v}}{\|\mathbf{v}\|}$. The corresponding normal vectors are computed in the direction of the streamline and path curvatures as suggested in Figure 3. This direction is determined via the direction of the fluid and particle accelerations, respectively, as shown in Eq.(23) and Eq.(24).

$$\hat{n}_s = \frac{\frac{D\mathbf{u}}{Dt} - \left(\frac{D\mathbf{u}}{Dt} \cdot \hat{s}\right)\hat{s}}{\left\|\frac{D\mathbf{u}}{Dt} - \left(\frac{D\mathbf{u}}{Dt} \cdot \hat{s}\right)\hat{s}\right\|} \quad (23)$$

$$\hat{n}_t = \frac{\frac{d\mathbf{v}}{dt} - \left(\frac{d\mathbf{v}}{dt} \cdot \hat{t}\right)\hat{t}}{\left\|\frac{d\mathbf{v}}{dt} - \left(\frac{d\mathbf{v}}{dt} \cdot \hat{t}\right)\hat{t}\right\|} \quad (24)$$

2.3.1 Data Scaling Early tests using raw input data shown in Eq.(21) to train relatively shallow (2-3 HL) ANNs showed that there is an inherent need to scale the data such that

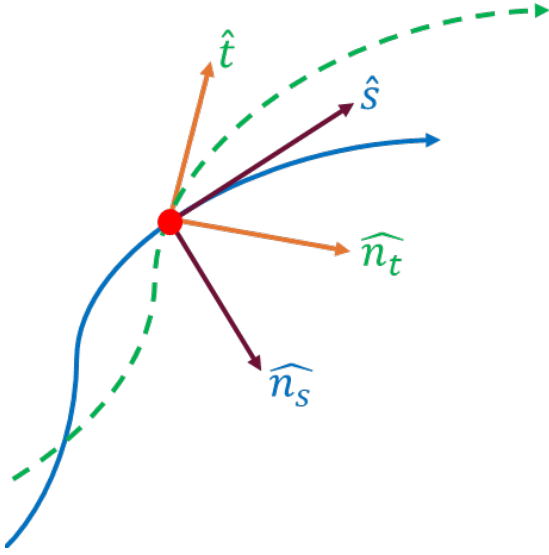


FIGURE 3: STREAMLINE AND PATH COORDINATES

the machine learning model can effectively learn from the training data set. Two such scaling strategies were explored. The first was a column-wise normalization of the input and output data tensors shown by Eq.(25), where each variable denoted by y_i is normalized by its corresponding absolute maximum in the data-set (i.e., each d_p is normalized by the largest d_p and so forth). This normalization ensures that the data is scaled within range $[-1, 1]$. The second strategy was a Gaussian standardization approach shown in Eq.(26), where $\bar{y}$ and s are the sample mean and sample standard deviation of the data set, respectively. This form of scaling does not impose any predefined limits on the scaled data set; however, for a normally distributed data set the standardized inputs $\tilde{y}_i$ should primarily be in range $[-3, 3]$.

$$\hat{y}_i = \frac{y_i}{\max(|\mathbf{y}|)} \quad (25)$$

$$\tilde{y}_i = \frac{y_i - \bar{y}}{s} \quad (26)$$

Initial testing suggested that the absolute maximum normalization was superior to the standardization approach. This is likely due in part to the wide range of particle sizes considered. It's unlikely that the data would assume a Gaussian distribution for this reason, as small particles will experience a much larger acceleration than the larger particles. The ANN training results presented in this work utilize the normalization approach for pre-processing the training data, unless otherwise specified. It is also important to note that the ANN-predicted particle accelerations are un-normalized for any post-processing error analysis

presented, as the error between scaled data is *not* the same as the un-normalized, or full-scale, data.

2.4 Generalized Approach

Expanding upon the idea that complex flows can be decomposed according to the relative contributions of rotation and strain at a given point, the forced vortex and irrotational shear flows were combined for a generalized approach. This velocity field is given by $\mathbf{u} = [(e - \omega)x_2, (e + \omega)x_1]$. Similarly to previous discussion, $S = e$, $\Omega = \omega$, and $J = \sqrt{e^2 + \omega^2}$. While keeping $J = 1.0 \text{ s}^{-1}$ in accordance with the unit circle concept presented in Figure 2, combined flows from $\theta = [0^\circ, 360^\circ)$ in 15° increments are considered. Like before, training data for 5,000 particles of each St considered are extracted at $t^* = 1.0$ for each different flow case. However, the ANN model inputs are altered to reflect the increased complexity of the flow as

$$\frac{d\mathbf{v}^*}{dt^*} = g \left(\mathbf{v}^*, \mathbf{u}^*, d_p, \frac{D\mathbf{u}^*}{Dt^*}, \rho_p, \rho_g, \mu_g, \frac{e}{J}, \frac{\omega}{J} \right) \quad (27)$$

such that the model can use the relative contributions of rotation and strain at the particle location as a means of flow feature extraction for general flows. This ANN will be referred to as the Master ANN.

2.4.1 Homogeneous Isotropic Turbulence Any flow can be decomposed into a purely rotational and purely straining component, in theory. However, in practice, the use of laminar rotational and strain may or may not be an appropriate model for real flows. Part of the complication is that while the network was primarily trained for steady laminar flows, real flows often become unsteady and turbulent. To test the efficacy of the neural network in turbulent flows, a canonical turbulent case must be generated. One such case is homogeneous isotropic turbulence (HIT). HIT is an idealized turbulent flow that has the property that its statistics are invariant with respect to position (homogeneous) and orientation (isotropic). In real flow, HIT is often a good approximation of flow behavior away from boundaries. In practice, HIT is a canonical flow type that is used to study and gain knowledge of turbulent flows [24].

An advantage of HIT compared to other flow types is that it is easy to generate such a flow field computationally. Strategies for generating such a flow are discussed in [25]. This work uses an improved version of that strategy from [26]. One disadvantage of HIT as a testing mechanism is that true HIT is inherently three dimensional, whereas the neural net developed has been developed in two dimensions. In addition to the true 3D HIT, simulations have also been performed with a 2D version of the HIT approach from [26].

For HIT there are two natural time scales [24]. The time scale of the smallest and fastest turbulent structure is given by

the Kolmogorov time scale τ_k . The time scale of the largest and slowest timescales is given by the integral timescale, τ_{HIT} . Here the appropriate choice of τ_f for computing St is τ_k . Better predictions can be made if the particle response time is corrected with the Schiller-Naumann drag model to be $\tau_{p,HIT}$ as defined in Eq.(28) [27] where u' is used for the characteristic velocity. As discussed in [25], for the forcing technique deployed here, the integral length ℓ is around 40% of the domain size, and the mesh size is chose to be about half the Kolmogorov length scale to ensure proper grid resolution.

$$\tau_{p,HIT} = \frac{\tau_p}{1 + 0.15Re_p^{0.687}} \quad (28)$$

For HIT simulations in this study, Eulerian-based gas phase transport equations and Lagrangian-based particle tracking equations (Eq.(6) & Eq.(7)) are utilized for the unique challenges that particle laden flows present [28] [29]. This Euler-Lagrange method uses Direct Numerical Simulation (DNS) to solve the transport equations, which are given by the Navier-Stokes equations. 1000 particles each of $St = 2 * [10^{-2}, 10^{-1}, 10^0, 10^1]$ are considered in a cubic domain. Particle data are extracted at multiples of τ_k to evaluate the master ANN.

3 RESULTS AND DISCUSSION

3.1 Neural Network Investigation

The HL structure testing detailed in Table 1 was performed using training data from the forced vortex case ($\omega = 1.0s^{-1}$) of the Cartesian DD approach. Each ANN was trained using the hyperbolic tangent activation function (tanh) on an NVIDIA V100 GPU for one hour. Increasing the number of HL does appear to have some marginal benefit. This benefit is especially clear between the 2 HL and 3 HL cases as shown in Table 2. As the number of HL is increased to 3, the number of epochs (iterations over the entire data set) completed in the allotted training time increases by about 20%. Subsequently, the final training loss, computed as mean squared error (MSE), is smaller. Additionally, it was seen that ANNs with fewer HLs trained more quickly during the first part of training but more slowly near the end of the allotted time frame. This can likely be attributed to the lower number of backpropagation calculations necessary for the shallow ANNs and the increased capability of deep neural nets to more easily capture complex phenomena in training data. From the ANN structures tested here, the 4 HL appears to facilitate the fastest training and lowest overall loss.

Once the 4 HL structure was deemed the local optimum of the design space considered, several activation functions were compared as shown in Figure 4. The forced vortex Cartesian DD training set was again used to compare activation functions. As

TABLE 2: HL STRUCTURE PERFORMANCE. Trained with Cartesian DD approach data set using tanh activation function.

Number H.L.	No. Iterations	Training Loss
2	1.36E+05	2.43E-03
3	1.64E+05	1.91E-03
4	1.67E+05	1.58E-03
5	1.57E+05	1.65E-03
6	1.45E+05	1.91E-03

before, each neural network shown was trained on an NVIDIA V100 GPU for one hour. Additionally, the ANN training results shown by the dashed line for each corresponding activation function represent an ANN with HL weights generated using the uniform Xavier (or Glorot) initialization presented by [18]. Clearly, the Softmax and LogSigmoid functions fail to train the ANN in any meaningful way. Since the 4 HL net is within the realm of deep neural networks, it was expected that the ReLU class of activation functions would be the best for training; however, this is not exactly the case. The Rectified Linear Unit (ReLU) and LeakyReLU activation functions perform slightly better than the Softmax and LogSigmoid functions, only to suffer from what appears to be the dying ReLU problem. This is the case for both the standard and Xavier initializations. One possible explanation for the lack of training with the ReLU functions is that their function structure is not well-suited to handling data where positive and negative values are present as is the case for velocity and acceleration data. This is interesting because the ReLU activation function is the most widely used in deep learning applications for both research and practice [30]. The tanh activation function appears to have performed reasonably well. Additionally, it is clear that the Xavier initialization provides some advantage for use with tanh because the loss decreases faster than the standard case. Lastly, the Scaled Exponential Linear Unit (SELU) activation function, which causes the ANN to exhibit some self-normalizing properties [31], outperforms the other activation functions considered and achieves a loss of less than $1.0E-4$ in the 1 hour of training time. The Xavier initialization does not appear to provide any immediately obvious advantages in use with SELU; however, it does not negatively impact the training and may provide some benefits in a case trained for a longer duration. For this reason, SELU and the uniform Xavier initialization are used in subsequent training.

Figure 5 shows the training results of the six different combinations of data preprocessing methods for the forced vortex flow, and Table 3 shows the final training loss for each flow case where $A = e = \omega = 1.0s^{-1}$. Again, each neural network shown

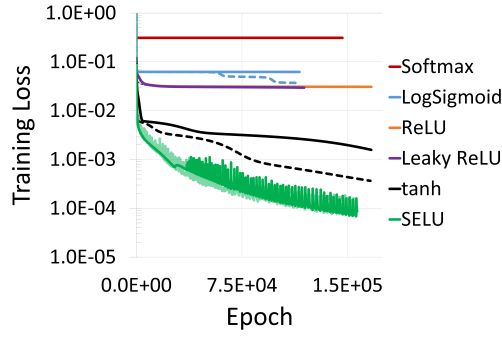


FIGURE 4: ACTIVATION FUNCTION COMPARISON. Standard SGD was used as the optimizer for each case shown here, and the dashed lines indicate use of the uniform Xavier weight initialization. Note that the SELU activation function with Xavier initialization is shown by the translucent green line.

was trained on an NVIDIA V100 GPU for one hour. It is worth noting that the trends shown here may not be comprehensive because the loss is only sampled every 50 epoch. As shown in Table 3, the training data in the streamline and path coordinate systems generally achieve lower loss than their Cartesian counterparts. Stagnation point flow is the only real exception in this case as the ANN trained with Cartesian DD training data closely compares to the streamline DD trained ANN. This can likely be attributed to the fact that the Cartesian and streamline coordinate systems are similar near the x and y axes for stagnation point flow. It is also clear from Figure 5 that while the PDP approach tends to perform slightly better during the initial training, the DD approach achieves a considerably lower loss by the end of the training time. Hence, path and streamline DD emerge as the top two approaches considered. Their ANN training results are quite similar for the forced vortex and irrotational shear cases. However, this study proceeds with the streamline DD approach because it appears to have less indeterministic behavior overall compared to the path DD approach, especially in the forced vortex case shown in Figure 5.

Having now determined an optimum activation function and data preprocessing method, the forced vortex ($\omega = 1.0s^{-1}$) streamline DD training data set was used to explore the performance of typical optimization algorithms. For useful comparison, all optimizers are assigned $lr = 10^{-3}$, and the stopping criteria for the training loss was 10^{-6} . As shown in Figure 6, the Adam algorithm trains the network faster than all other algorithms, stopping in just 597 epoch. Adagrad and SGD with Nesterov momentum ($\mu=0.9$) exhibit similar performance and each meet the stopping criteria in around $9.0E+4$ epoch. Note that standard SGD and SGD with standard momentum were also tested and SGD with Nesterov momentum outperformed these two. Lastly, the Adadelta and RMSProp algorithms do not train

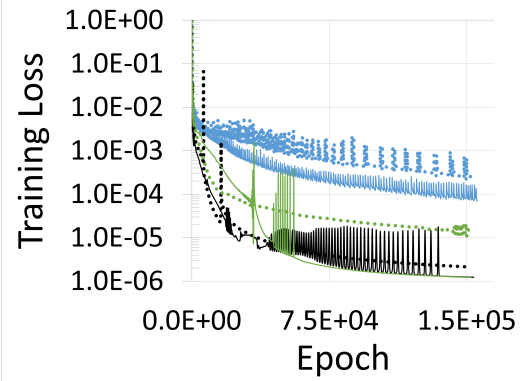


FIGURE 5: DATA PREPROCESSING EFFECTS. The Cartesian, path, and streamline coordinates are denoted by the blue, black, and green lines, respectively. The DD and PDP data are indicated with solid and dotted lines, respectively.

TABLE 3: DATA PREPROCESSING EFFECTS. Trained with SGD using SELU activation function. Final training loss after one hour is shown for each flow case and data preprocessing method.

Forced Vortex	DD	PDP
Cartesian	6.98E-05	2.39E-04
Path	1.23E-06	2.12E-06
Streamline	1.21E-06	1.94E-05
Irrotational Shear	DD	PDP
Cartesian	9.41E-05	3.71E-04
Path	2.30E-05	9.35E-05
Streamline	2.75E-05	7.14E-05
Stagnation Point	DD	PDP
Cartesian	6.36E-05	2.44E-04
Path	3.91E-05	4.98E-05
Streamline	5.80E-05	1.22E-04

the ANN to the stopping criteria. RMSProp exhibits heavy oscillation during training that was found to be independent of lr . While Adam does exhibit some oscillation, the overall training trend is obvious. In the case that one has access to computing resources without time restrictions, Adam is the obvious choice. However, if time usage of computing resources is limited, SGD

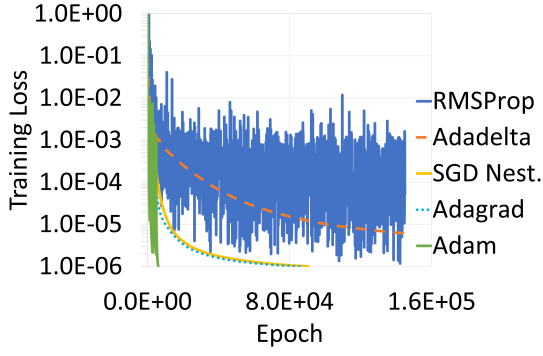


FIGURE 6: OPTIMIZATION ALGORITHM COMPARISON ($lr = 10^{-3}$).

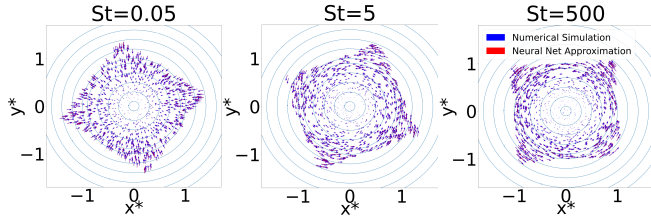


FIGURE 7: PARTICLE ACCELERATION VECTOR FIELD - FORCED VORTEX SIMULATION RESULTS VS ANN PREDICTION

with Nesterov momentum is likely the better choice as its convergence seems to be monotonic. Additionally, Adam is a good development tool to use to determine how much convergence is possible for a given ANN.

3.2 Canonical Flows

After training ANNs for each of the canonical flows considered to $MSE < 10^{-6}$ with the Adam algorithm, the particle acceleration from the governing equations was compared to the ANN-predicted acceleration. Figure 7 shows this visual comparison for the forced vortex case by St , where the simulation acceleration and ANN-predicted acceleration are shown with the blue and red arrows, respectively. Upon visual inspection, the ANN-predicted accelerations appear to match the actual accelerations well. However, it is difficult to see some of the smaller vectors near the origin.

Now that a qualitative analysis of the ANN performance has been demonstrated, a quantitative analysis is in order to better characterize the model error. The directional error β and the error vector $\mathbf{a} - \mathbf{a}_{NN}$ are defined as shown in Figure 8, where $\mathbf{a} = \frac{d\mathbf{v}}{dt}$ as defined in Eq.(7) and $\mathbf{a}_{NN}$ is the particle acceleration predicted by the ANN. The angle β is computed using the dot

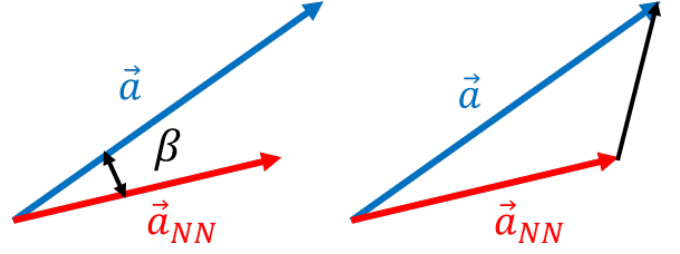


FIGURE 8: ANN ERROR DEFINITIONS.

product definition as shown in Eq.(29). The relative error is defined as $\mathbf{e} = (\mathbf{a} - \mathbf{a}_{NN})/||\mathbf{a}||$ and is used to ensure that the metric is comparable for various St . Additionally, a third relative scalar error metric is defined as $E = (||\mathbf{a}_{NN}|| - ||\mathbf{a}||)/||\mathbf{a}||$, which describes how well the magnitude of the acceleration was predicted relative to $\mathbf{a}$ and if it was over- or under-predicted.

$$\beta = \arccos\left(\frac{\mathbf{a} \cdot \mathbf{a}_{NN}}{||\mathbf{a}|| ||\mathbf{a}_{NN}||}\right) \quad (29)$$

Figure 9 shows the mean value of β calculated from the validation data sets (5000 total particles) as a function of St for each canonical flow considered. As shown, the mean directional error remains less than 0.5° for $St \leq 1$ or lower, and increases for larger St . It is also useful to know if the error is entirely random or if the mean error is influenced by outliers. Figure 10 shows surface plots of directional error β as a function of $\mathbf{x}_p^*$ for the rotational vortex, where the red color indicates higher error. Clearly, the ANN struggles to accurately predict the acceleration direction near the origin of each canonical flow. This is likely the case because $\mathbf{v}_r$ is small in this region of the flow, and is near the edge of the model's working range. Although only $St = 0.05, 5, \text{ and } 500$ are shown, this trend was consistent for all St . Note the varying z-axis (and corresponding colorbar) scales, which show that the maximum directional error for $St = 0.05$ is much smaller than that of $St = 500$. Going one step further, Figure 11 shows the forced vortex directional error histograms by St with bin widths of 1° . Similar distributions were observed for stagnation point and irrotational shear flow. Note that Figure 11's y-axes are shown with a log scale for the purpose of highlighting the range of β experienced by each St in the respective flow case. The largest β bin for each case is from $0 - 1^\circ$, and the second largest β bin has a count that is an order of magnitude smaller for every histogram except for the $St = 500$ particles in the forced vortex. The small $St = 0.05$ particles are comparatively well-characterized for the forced vortex and irrotational shear as the directional error does not exceed 5° for either case. One interesting observation for the $St = 0.5$ particles in irrota-

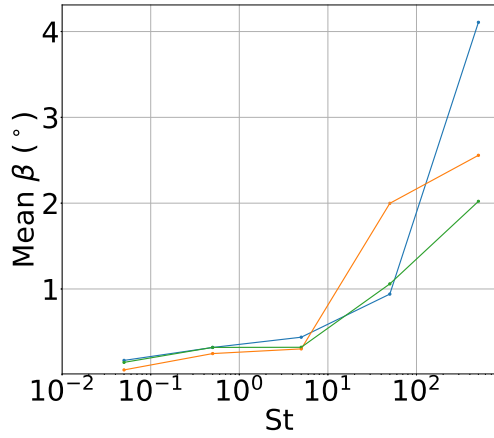


FIGURE 9: MEAN DIRECTIONAL ERROR β . The results of the forced vortex, irrotational shear, and stagnation point flows are shown with by the orange, green, and blue lines, respectively.

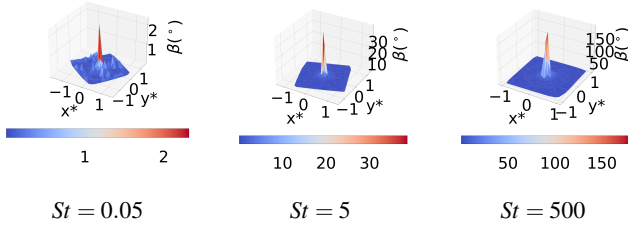


FIGURE 10: DIRECTIONAL ERROR (β) FORCED VORTEX POSITION DEPENDENCE

tional shear and the forced vortex is that there is a larger maximum β observed for these than for the $St = 5$ particles which is a break in the trend of increasing maximum β with St . This could possibly be attributed to the more complicated preferential concentration observed in the literature for particles where $St \sim O(1)$ but $St < 1$ [32]. However, this trend was not observed in stagnation point flow.

Figure 12 shows the root mean square (RMS) of the error vector magnitude $\|\mathbf{e}\|$ vs St for the three single canonical flows. Like β , the RMS of $\|\mathbf{e}\|$ generally increases with St . The most glaring exception shown in the graph is for the forced vortex case at $St = 0.5$ and $St = 50$. Considering the y-axis scale of Figure 12, this could be a reflection of statistical noise. Again, it is useful to know how this error depends on position in the flow and if there are any systematic trends. Figure 13 shows the magnitude of $\mathbf{e}$ as a function of $\mathbf{x}_p^*$ for the stagnation point and forced vortex flows. For the stagnation point flow, the highest errors occur

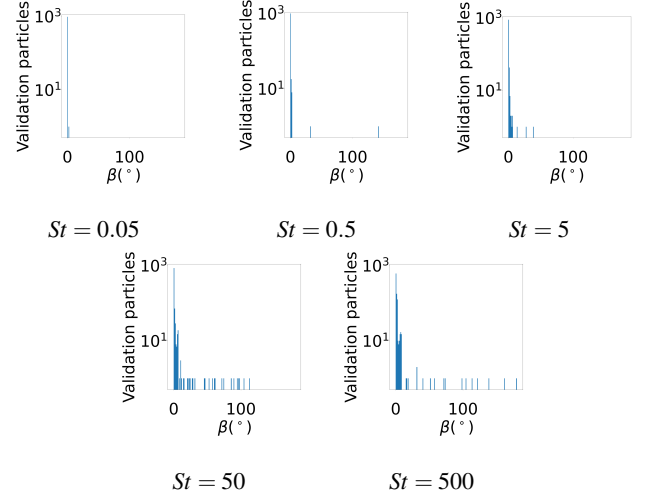


FIGURE 11: FORCED VORTEX DIRECTIONAL ERROR (β) DISTRIBUTION. The bin widths are 1° .

along the y-axis primarily near the origin and near the limits of existing particle position. For the stagnation point flow, the y/x axes are streamlines that define and divide the overall motion of the flow. These patterns in the error are thought to be the result of particles deviating from and crossing these streamlines. The forced vortex case exhibits its peak error near the origin for all St and does not suggest error correlation with the streamlines.

Lastly, Figure 14 shows an error averaged by St previously defined as E . E describes how magnitude of the ANN-predicted acceleration compares to the actual particle acceleration. If E is negative, the model has under-predicted the acceleration magnitude and vice-versa. While no strong trends can be identified, this plot does suggest that the ANN is producing accelerations on the correct order of magnitude as the maximum average error observed is approximately 0.1 or 10%. This maximum error was found for $St = 500$ in stagnation point flow. The stagnation point and irrotational shear seem to follow similar trends, and the forced vortex appears to trend in the opposite manner. Additionally, it does appear that the models generally tend to over-predict **a**.

As a means to validate the end goal of predicting particle motion with machine learning, the ANN trained for the forced vortex was substituted for Eq.(7) in the LSODA solver. Figure 15 shows the ANN's prediction of particle motion for ten particles of $St = 5$ and $St = 500$, and the figures on the left column were initialized with no velocity while the right column figures were all initialized with a random velocity in range $\mathbf{v}^* = [\pm 1, \pm 1]$. The ANN does seem to predict the initial direction of motion for most cases; however, the predicted trajectories quickly diverge. As shown in Figure 15, the $St = 500$ case with zero initial velocity matches the governing equation solutions much better

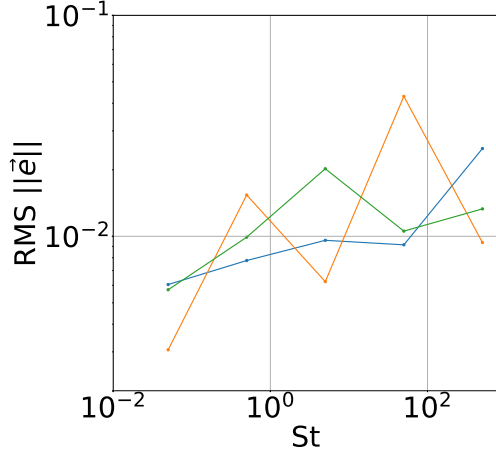


FIGURE 12: RMS OF $\|\mathbf{e}\|$ VS St . The results of the forced vortex, irrotational shear, and stagnation point flows are shown with the orange, green, and blue lines, respectively.

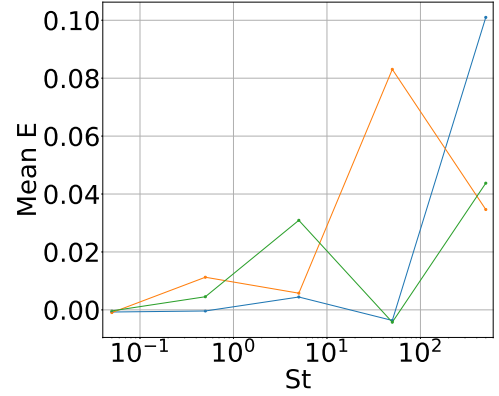


FIGURE 14: MEAN E VS St . The results of the forced vortex, irrotational shear, and stagnation point flows are shown by the orange, green, and blue lines, respectively.

3.3 Combined Flows

The original decreasing HL size ANN structure proved impractical for training with multiple sets of combined flow data. Therefore, following [31], a rectangular-like block structure ANN was tested and slightly tuned. The rectangular block structure has the same number of neurons in each HL, and this work refers to the structure as rectangular-like because the last two HL are made smaller such that some form of abstract generalization is forced. This ANN structure proved more feasible with the combined flow particle data from $\theta = 15^\circ - 360^\circ$. After testing various ANNs up to 10 HL deep, a 7 HL rectangular-like block structure ANN (150 neurons in first five HL and 100 in the last two HL) was trained with the Adam optimizer and SGD with Nesterov momentum. The Adam optimizer converged reached the stopping criteria of $1.0E-5$ in around 10.25 hours, while SGD reached a loss of $5.24E-5$ after training for 48 hours. It should be noted that the streamline data-driven (DD) preprocessing approach presented in Section 3.1 is used for all results shown in this section, unless otherwise noted. Specifically, the ANN input vector is $[\hat{v}_s^*, \hat{v}_n^*, \hat{u}_s^*, \hat{d}_p, \frac{D\hat{u}_s^*}{Dt^*}, \frac{D\hat{u}_n^*}{Dt^*}, \hat{\rho}_p, \hat{\rho}_g, \hat{\mu}_g, \frac{\hat{e}}{J}, \frac{\hat{\omega}}{J}]$, and the model output is $[\frac{d\hat{v}_s^*}{dt^*}, \frac{d\hat{v}_n^*}{dt^*}]$ where the $(\hat{\cdot})$ notation refers to data normalized by the absolute maximum values of the corresponding variable in the data set as previously defined by Eq.(25). Thus, all normalized ANN inputs are scaled within range $[-1, 1]$. The subscripts s and n indicate the stream-wise and normal vector components, respectively. That is, $\mathbf{v} = v_s \hat{s} + v_n \hat{n}_s$, where the unit vectors $\hat{s}$ and $\hat{n}_s$ are defined in Section 2.3. It should be noted that $\hat{u}_s^* = \|\hat{\mathbf{u}}^*\|$ because the streamline coordinate $\hat{s} = \frac{\hat{\mathbf{u}}}{\|\hat{\mathbf{u}}\|}$ is defined as the unit vector in the direction of fluid velocity.

The following data analysis comes from the Master ANN

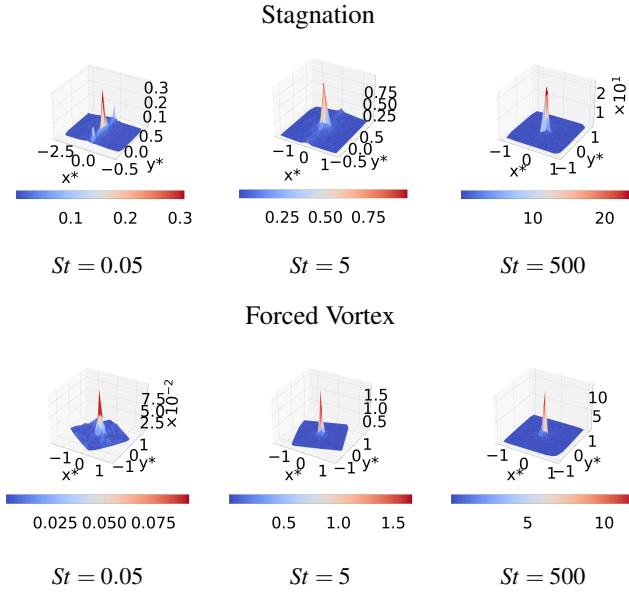


FIGURE 13: $\|\mathbf{e}\|$ POSITION DEPENDENCE

than the other zero initial velocity cases. Additionally, the $St = 5$ case with random initial velocity appears to match the governing equation solutions better than the other random initial velocity cases.

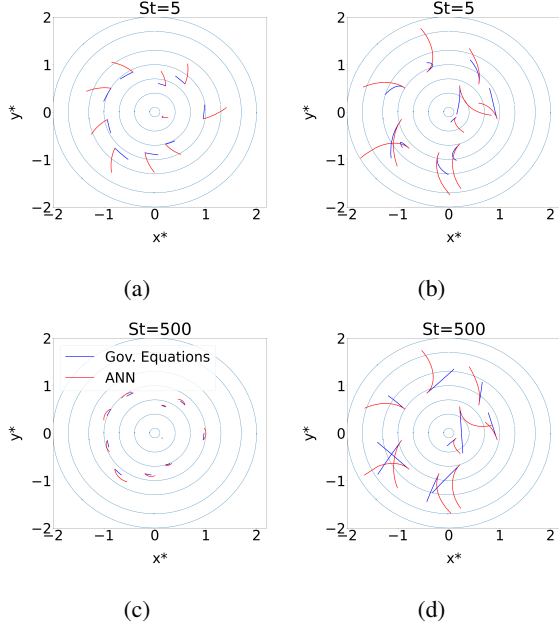


FIGURE 15: ANN PARTICLE MOTION VALIDATION. The exact particle trajectories (blue) and ANN approximation (red) are shown superimposed over the circular streamlines. Particles were initialized with zero (left) or random velocity (right). Trajectories are for $t^* = [0, 1]$ except (c), shown for $t^* = [0, 3]$.

trained with the Adam optimizer. As before, the directional error is first considered. Figure 16 shows the distribution of β for the five different St considered. Note that the y-axes are no longer displayed with a logarithmic scale. The Master ANN is much more adept at predicting the acceleration direction for the smaller particles. For the largest particles, it appears that about half of the model predictions are in the exact opposite direction of the actual acceleration ($\beta = 180^\circ$). For $St = 0.05$ and $St = 0.5$, the distribution is concentrated around 0° . For $St = 5$ and above, a bimodal distribution is observed, where the concentrated regions are around 0° and 180° . While the model performance is clearly sub-par, it is encouraging to see that most densely populated region of the histogram is near the lower error.

Additionally, $\|\mathbf{e}\|$ is shown as a function of $\mathbf{x}_p^*$ by St in Figure 17. Note that a more tilted view of the scatter plot is given for $St = 0.05$ because the isometric view is not easily interpretable for this St . While these errors are much higher than observed in the single flow cases, several interesting trends can be observed. First, the magnitude of $\mathbf{e}$ generally increases with St . However, it should also be noted that the $St = 0.5$ case produced several outliers higher than any value observed in the $St = 5$ case. This suggests that the complex preferential concentration at $St O(1) < 1$ previously mentioned may be affecting the ANN's performance. Additionally, many of the peaks observed on these error plots

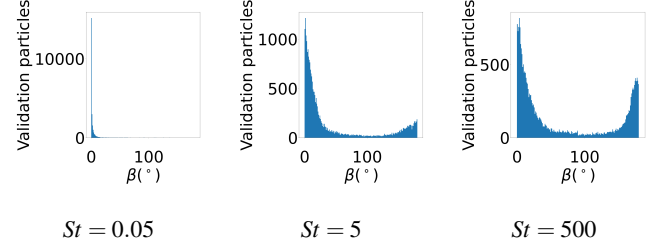


FIGURE 16: COMBINED FLOW DIRECTIONAL ERROR DISTRIBUTION

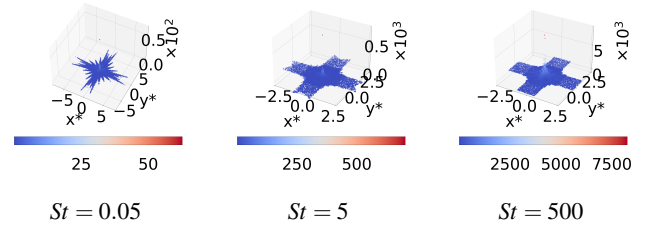


FIGURE 17: COMBINED FLOW RELATIVE ERROR MAGNITUDE ($\|\mathbf{e}\|$) POSITION DEPENDENCE

occur near the origin, which is consistent with the results shown in Figure 13. Beyond this, it is not possible to identify error dependence on position because all 24 variations of combined flow ($0^\circ \leq \theta < 360^\circ$) are shown in this plot.

It follows that the error dependence on θ , as defined in Figure 2, is examined. As shown in Figure 18, the error β appears to have some dependence on θ for $St = 0.05$ and $St = 0.5$. For these St , error decreases as θ increases from 0° to 180° and then increases from 180° to 360° . This does not indicate any preference by the ANN for flow dominated by strain or rotation, as $\theta = 0^\circ$ and $\theta = 180^\circ$ are both irrotational. It could simply mean that the ANN has some implicit numerical preference for the streamlines configured with $\theta = 180^\circ$. Additionally, while not as strong as the smaller St , $St = 5$ appears to exhibit some symmetry about $\theta = 180^\circ$.

Figure 19 shows the the RMS of $\|\mathbf{e}\|$ vs θ by St , which is again shown to increase with St . Similarly to the trends observed in Figure 18, the errors for $St = 0.05$ and $St = 0.5$ exhibit some symmetry about $\theta = 180^\circ$, and the higher St have no discernible dependence on θ .

3.4 Homogeneous Isotropic Turbulence

It is not surprising that the neural network trained with 2D data does not do a great job of capturing the behavior of the 3D HIT. Since it cannot predict motions out of plane, the predicted

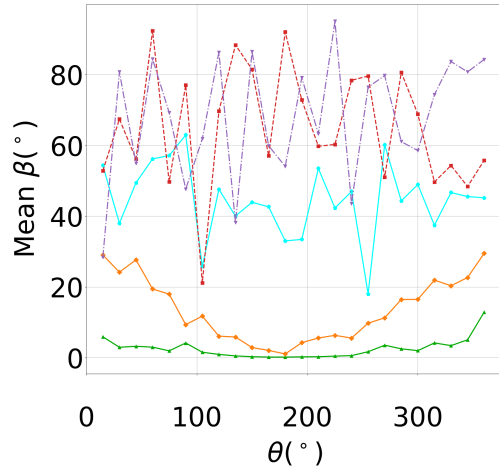


FIGURE 18: β VS θ . The green triangles, orange diamonds, blue pentagons, red squares, and purple upside-down triangles correspond to $St = 0.05, 0.5, 5, 50$ and 500 , respectively.

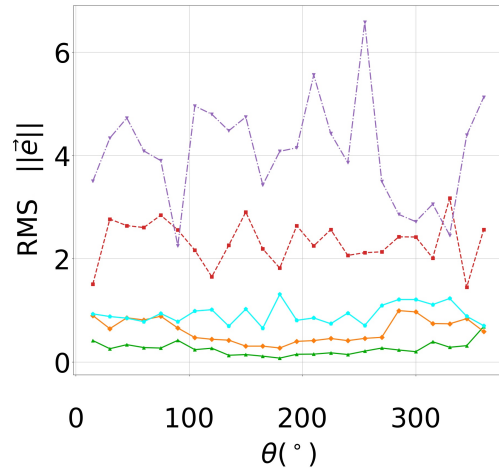


FIGURE 19: RMS OF $\|\mathbf{e}\|$ VS θ . The green triangles, orange diamonds, blue pentagons, red squares, and purple upside-down triangles correspond to $St = 0.05, 0.5, 5, 50$ and 500 , respectively.

acceleration will almost always be incorrect. Figure 20 shows the directional error vs θ by St . It is important to note for this case that the flow characteristic is calculated as $\theta = \arctan(\frac{\Omega}{S})$ and is therefore limited to the range $(0^\circ, 90^\circ)$ since Ω and S are positive by definition. The average directional error is shown to be no less than 80° . Interestingly, the highest error is associated with the smallest St in this case. No obvious trends with θ are observed other than β for $St = 0.2, 2$, and 20 increase as θ approaches 90° ,

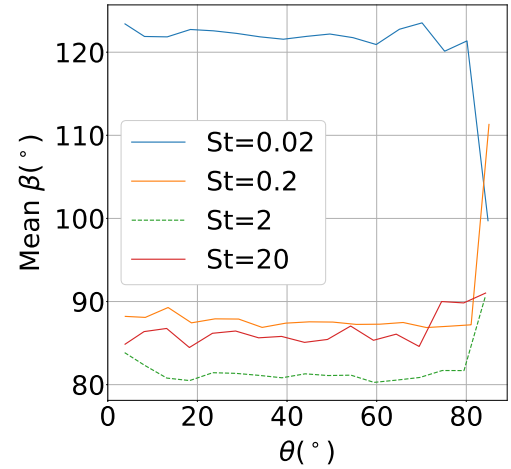


FIGURE 20: 3D HIT FLOW β VS θ

which corresponds to flow where rotation is dominant. The $St = 0.02$ directional error decreases as θ approaches 90° . Figure 21 shows the average magnitude of $\mathbf{e}$ as a function of θ by St . $St = 2$ results in the highest overall error for this metric and appears to be increasing with θ . The other three St do not appear to have any direct correlation with θ ; however, they do seem to decrease as θ approaches 90° . Additionally, it should be noted that a large spike in error occurs around $\theta = 60^\circ$ for $St = 20$. Lastly, Figure 22 shows the mean of E as a function of θ by St . The lowest and highest errors for this metric occur for $St = 0.2$ and $St = 2$, respectively. The data for these two St , while different in value, follow similar trends. Interestingly, the data for $St = 0.02$ and $St = 20$ also follow the same overall trend, although not as closely.

4 CONCLUSION

The viability of deep, feed-forward neural nets to predict particle acceleration has been studied for several canonical flows. For each canonical flow, a four hidden-layer ANN with successively smaller hidden layers was built and trained using *PyTorch*. The directional error and the RMS of the relative error vector magnitude were found to increase with St . Three-dimensional representations of the error $\|\mathbf{e}\|$ as a function of particle position for the stagnation point flow suggest that there is some correlation between the observed error and the fluid streamlines. The error is slightly larger for particles located on the dividing streamline for the stagnation point flow (y -axis), which could be a result of the particles deviating from this streamline. Despite not being trained with transient data sets, the ANN can predict the trajectory of $St = 500$ particles with no initial velocity and $St = 5$ particles initialized with a random initial velocity with

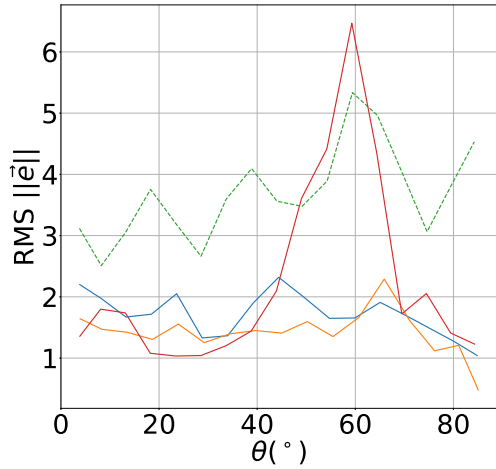


FIGURE 21: 3D HIT FLOW RMS OF $\|\vec{e}\|$ VS θ

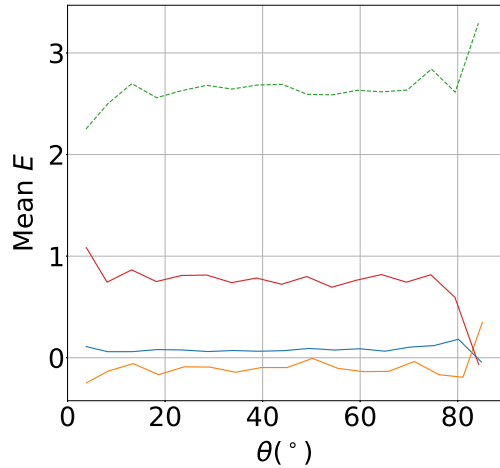


FIGURE 22: 3D HIT FLOW MEAN E VS θ . The blue, orange, dashed green, and red lines represent $St = 0.02, 0.2, 2$, and 20 , respectively.

some accuracy.

For laminar flow combining shear and vortex flows, the decreasing hidden-layer size network architecture was not sufficient for meaningful model training with the more complex combined flow case. Therefore, a seven hidden-layer ANN with a rectangular-like block structure (five hidden layers of the same size followed by two smaller hidden layers) was used. However, it is clear that the ANN does not fully model the function space. Of the particles considered, the combined flow ANN's performance was the best for the two smallest St considered in

the combined laminar flows. For the larger St in the combined flow cases, the predicted particle acceleration was often in the exact opposite direction of the actual vector. Additionally, for the 3D HIT case, the combined flow ANN failed to predict the particle accelerations in any meaningful manner. This is not surprising, as the combined flow ANN was trained with steady 2D laminar combined flow data and that network struggled to learn even the 2D space.

Future Studies The results of this study suggest that the the deep feed-forward neural nets presented here are not sufficient to capture the function space of particle acceleration. It will be the topic of future work to use more advanced strategies for developing neural nets. For example, the current work focused only on feed-forward neural networks; however, convolutional neural networks are more popular in practice. The work can also use more sophisticated manners of preprocessing data for the neural network. The first alternative solution is to consider more complex variations of ANN models such as generative adversarial networks (GANs), which have demonstrated capability of forecasting high-fidelity simulations [33]. Separately, a physics-informed neural network (PINN) approach [10] with a loss function informed by the relevant governing equations may offer some advantage. Lastly, improvements can be made to the neural network itself by training with three-dimensional unsteady data. These three approaches could be implemented independently to demonstrate their unique advantages and downfalls and combined accordingly such as in [34].

ACKNOWLEDGMENT

We would like to thank the Virginia Space Grant Consortium and the Sloan Scholars Mentoring Network for their funding and support of this research. The authors also acknowledge Advanced Research Computing at Virginia Tech for providing computational resources and technical support that have contributed to the results reported within this paper. URL: <https://arc.vt.edu/>

REFERENCES

- [1] NATO, 1994. "Erosion Corrosion and Foreign Object Damage Effects in Gas Turbines". *AGARD Conference Proceedings*.
- [2] Pan, H., and Render, P. M., 1996. "Impact characteristics of hailstones simulating ingestion by turbofan aeroengines". *Journal of Propulsion and Power*, **12**(3), pp. 457–462.
- [3] Flegel, A. B., 2017. "Ice crystal icing research at NASA". *9th AIAA Atmospheric and Space Environments Conference*, 2017(November).
- [4] Leroy, D., Fontaine, E., Schwarzenboeck, A., Strapp, J. W., Korolev, A., McFarquhar, G., Dupuy, R., Gourbeyre, C., Lilie, L., Protat, A., Delanoe, J., Dezitter, F., and Grandin, A., 2017. "Ice Crystal Sizes in High Ice Water Content Clouds. Part II: Statistics of Mass Diameter Percentiles in Tropical Convection Observed during the HAIC/HIWC Project". *Journal of Atmospheric and Oceanic Technology*, **34**(1), 1, pp. 117–136.
- [5] Matson, R. J., and Huggins, A. W., 1980. "The Direct Measurement of the Sizes, Shapes and Kinematics of Falling Hailstones". *Journal of the Atmospheric Sciences*, **37**(5), 5, pp. 1107–1125.
- [6] Brandt, L., and Coletti, F., 2021. "Particle-Laden Turbulence: Progress and Perspectives". *Annual Review of Fluid Mechanics*.
- [7] Esmaily-Moghadam, M., and Mani, A., 2016. "Analysis of the clustering of inertial particles in turbulent flows". *Physical Review Fluids*, **1**(8), 12, p. 084202.
- [8] Maxey, M. R., 1987. "The gravitational settling of aerosol particles in homogeneous turbulence and random flow fields". *Journal of Fluid Mechanics*, **174**, pp. 441–465.
- [9] Minakowska, M., Richter, T., and Sager, S., 2021. "A Finite Element/Neural Network Framework for Modeling Suspensions of Non-spherical Particles: Concepts and Medical Applications". *Vietnam Journal of Mathematics*, **49**(1), 3, pp. 207–235.
- [10] Raissi, M., Perdikaris, P., and Karniadakis, G. E., 2019. "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". *Journal of Computational Physics*, **378**, 2, pp. 686–707.
- [11] Seyed-Ahmadi, A., and Wachs, A., 2022. "Physics-inspired architecture for neural network modeling of forces and torques in particle-laden flows". *Computers & Fluids*, **238**, 4, p. 105379.
- [12] Cutnell, J. D., and Johnson, K. W., 2012. *Physics*, 9 ed. Wiley.
- [13] Clift, R., Grace, J. R., and Weber, M. E., 1978. *Bubbles, Drops, and Particles*. Academic Press, Inc., New York.
- [14] Panton, R. L., 2013. *Incompressible Flow*, 4 ed. John Wiley & Sons, Inc., Hoboken, NJ, USA.
- [15] Hindmarsh, A., and Petzold, L., 2005. LSODA, Ordinary Differential Equation Solver for Stiff or Non-Stiff System.
- [16] Huang, G. B., 2003. "Learning capability and storage capacity of two-hidden-layer feedforward networks". *IEEE Transactions on Neural Networks*, **14**(2), pp. 274–281.
- [17] Heaton, J., 2015. *Artificial Intelligence for Humans: Neural Networks and Deep Learning*, 1 ed., Vol. 3. Heaton Research, Inc., Chesterfield, MO.
- [18] Glorot, X., and Bengio, Y., 2010. "Understanding the difficulty of training deep feedforward neural networks". *International Conference on Artificial Intelligence and Statistics in Proceedings of Machine Learning Research*, **9**, pp. 249–256.
- [19] Kingma, D. P., and Ba, J. L., 2015. "Adam: A Method for Stochastic Optimization". *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 12.
- [20] Duchi, J., Hazan, E., and Singer, Y., 2011. "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization". *Journal of Machine Learning Research*, **12**(61), pp. 2121–2159.
- [21] Zeiler, M. D., 2012. "ADADELTA: An Adaptive Learning Rate Method". *arXiv*.
- [22] Graves, A., 2014. "Generating Sequences With Recurrent Neural Networks". *arXiv*.
- [23] Sutskever, I., Martens, J., Dahl, G., and Hinton, G., 2013. "On the importance of initialization and momentum in deep learning". *International Conference on Machine Learning*, **28**.
- [24] Pope, S. B., 2000. *Turbulent Flows*. Cambridge University Press.
- [25] Palmore, J. A., and Desjardins, O., 2018. "Technique for forcing high Reynolds number isotropic turbulence in physical space". *Physical Review Fluids*, **3**(3), 3, p. 034605.
- [26] Miranda, C., and Palmore, J., 2020. "High Stokes Number Droplets in Homogeneous Isotropic Turbulent Flow". *Eastern States Section of the Combustion Institute - Spring Technical Meeting*, 3.
- [27] Hwang, W., and Eaton, J. K., 2006. "Homogeneous and isotropic turbulence modulation by small heavy ($St=50$) particles". *Journal of Fluid Mechanics*, **564**, 10, pp. 361–393.
- [28] Capecelatro, J., and Desjardins, O., 2013. "An Euler-Lagrange strategy for simulating particle-laden flows". *Journal of Computational Physics*, **238**, pp. 1–31.
- [29] Desjardins, O., Blanquart, G., Balarac, G., and Pitsch, H., 2008. "High order conservative finite difference scheme for variable density low Mach number turbulent flows". *Journal of Computational Physics*, **227**(15), 7, pp. 7125–7159.
- [30] Rasamoelina, A. D., Adjailia, F., and Sincak, P., 2020. "A Review of Activation Function for Artificial Neural Network". *SAMI 2020 - IEEE 18th World Symposium on Applied Machine Intelligence and Informatics, Proceedings*,

- 1, pp. 281–286.
- [31] Klambauer, G., Unterthiner, T., Mayr, A., and Hochreiter, S., 2017. “Self-Normalizing Neural Networks”. *Advances in Neural Information Processing Systems*, **2017-December**, 6, pp. 972–981.
 - [32] Eaton, J. K., and Fessler, J. R., 1994. “Preferential concentration of particles by turbulence”. *International Journal of Multiphase Flow*, **20**(SUPPL. 1), pp. 169–209.
 - [33] Pourbagian, M., and Ashrafizadeh, A., 2021. “Super-resolution of low-fidelity flow solutions via generative adversarial networks:”. *SIMULATION: Transactions of the Society for Modeling and Simulation International*, 12.
 - [34] Li, M., and McComb, C., 2022. “Using Physics-Informed Generative Adversarial Networks to Perform Super-Resolution for Multiphase Fluid Simulations”. *Journal of Computing and Information Science in Engineering*, **22**(4), 8.