Modeling the pressure-Hessian tensor using deep neural networks

Nishant Parashar,[1,*] Balaji Srinivasan,[2,†] and Sawan S. Sinha[1,‡]
[1]*Department of Applied Mechanics, Indian Institute of Technology, Delhi, New Delhi 110016, India*
[2]*Department of Mechanical Engineering, Indian Institute of Technology, Madras, Chennai 600036, India*

(Received 1 April 2020; accepted 22 October 2020; published 11 November 2020)

The understanding of the dynamics of the velocity gradients in turbulent flows is critical to understanding various nonlinear turbulent processes. Several simplified dynamical equations have been proposed earlier that model the Lagrangian velocity gradient evolution equation. A robust model for the velocity gradient evolution equation can ultimately lead to the closure of the system of equations in the Lagrangian probability distribution function method. The pressure Hessian and the viscous Laplacian are the two important processes that govern the Lagrangian evolution of the velocity gradients. These processes are nonlocal in nature and unclosed from a mathematical point of view. The recent fluid deformation closure model (RFDM) has been shown to retrieve excellent statistics of the viscous process. However, the pressure Hessian modeled by the RFDM has various physical limitations. In this work, we first demonstrate such limitations of the RFDM. Subsequently, we employ a tensor basis neural network (TBNN) to model the pressure Hessian using the information about the velocity gradient tensor itself. Our neural network is trained on high-resolution data obtained from direct numerical simulation (DNS) of isotropic turbulence at the Reynolds number of 433. The predictions made by the TBNN are evaluated against several other DNS datasets. Evaluation is made in terms of the alignment statistics of the pressure-Hessian eigenvectors with the strain-rate eigenvectors. Our analysis of the predicted solution leads to the finding of ten unique coefficients of the tensor basis of the strain-rate and the rotation-rate tensors, the linear combination of which is used to accurately capture key alignment statistics of the pressure-Hessian tensor.

DOI: 10.1103/PhysRevFluids.5.114604

I. INTRODUCTION

In a turbulent flow, various processes such as energy cascade, intermittency, and fluid element deformation are strongly related to the small scale velocity gradient field. Various experimental, direct numerical simulation, and simple dynamical models based studies have been performed to understand the dynamics of the velocity gradient tensor (Ashurst *et al.* [1], Vieillefosse [2], Cantwell [3], Martín *et al.* [4], Lüthi *et al.* [5], Kalelkar [6]). In continuation to these works, several other studies have been reported as well [1,7–26]. These studies are not just fundamental in nature, but also provide viable modeling approaches to model the processes governing the evolution of velocity gradients in turbulent flows.

Over the years, there have been various attempts to develop a simple ordinary differential equation (ODE) based closed set of dynamical equations which can provide a robust model for the Lagrangian evolution of velocity gradients [4,18,27–30]. Such models are not just intended to

[*]nishantparashar14@gmail.com
[†]sbalaji@iitm.ac.in
[‡]sawan@am.iitd.ac.in

develop a better understanding of velocity gradient dynamics, but also serve as a set of closure equations for Lagrangian PDF methods [31] for solving turbulent flows, which provides a computationally tractable way of calculating the statistics of turbulent flows of practical interest. The basic thrust of the models is to model the pressure-Hessian tensor and the viscous-Laplacian tensor, which are the two important nonlinear processes governing the evolution of the velocity gradient tensor. Although, in an Eulerian description of the flow field, the pressure field in an incompressible flow can be directly derived from the velocity field, in the Lagrangian description, the pressure field cannot be directly derived from the velocity field. This makes both the pressure-Hessian as well as the viscous-Laplacian unclosed processes in the Lagrangian evolution equation of the velocity gradient tensor.

Vieillefosse [2] made the first attempt to provide a simple ODE based model, named restricted Euler equation (REE), for the Lagrangian evolution of velocity gradients. The REE model has matured over the years due to subsequent attempts made by various researchers [4,18,27–30]. The recent fluid deformation closure model (RFDM) developed by Chevillard *et al.* [29] is currently the most advanced model for the Lagrangian velocity gradient dynamical equation. Although the RFD model robustly captures various statistics of the viscous process, it has various inherent limitations in predicting the pressure-Hessian tensor (discussed in Sec. III). These limitations arise due to an endeavour to model the pressure Hessian using velocity gradients alone, which ultimately contributes to an independent closed set of ODEs intended to model the velocity gradient dynamical equation. Such an attempt to model the pressure-Hessian tensor, which is expected to be dependent on a wide range of flow quantities, using simple algebraic functions over velocity gradients is a key bottleneck to the existing RFD model. Hence, in this work, we focus on finding more complex functions which can lead to a more sophisticated model for the pressure Hessian without changing the existing modeling paradigm of using instantaneous velocity gradient information for modeling. For this, we resort to a machine learning based approach using deep neural networks to learn a sophisticated functional mapping between pressure Hessian and velocity gradients using direct numerical simulation (DNS) data of incompressible isotropic turbulence.

In the recent past, machine learning has gained popularity in the turbulence research community. The earliest such contribution in the field of machine learning aided turbulence research was made by Duraisamy and Durbin [32], where the authors developed an intermittency transport based model for bypass transition using machine learning and inverse modeling. Since then, a large number of researchers have tried to model various turbulence processes using machine learning models [33–39]. Ling *et al.* [40] employed a deep neural network to directly model the Reynolds stress anisotropy tensor using strain-rate and rotation-rate tensors. In doing so, they developed a novel tensor basis neural network (TBNN), which can be employed to map a given tensor from known input tensors. The TBNN has been shown to achieve superior performance by embedding tensor invariance properties in the network itself. Later Fang *et al.* [41] used the TBNN for turbulent channel flow and compared their results against standard turbulence models. Sotgiu *et al.* [42] developed a new framework in conjunction with TBNN for predicting turbulent heat fluxes. Further, Geneva and Zabaras [43] developed a Bayesian tensor basis neural network for predicting the Reynolds stress anisotropy tensor.

The TBNN developed by Ling *et al.* [40] has already been shown to map tensorial quantities robustly and hence fits well for our problem of interest. In this work, we use high-resolution incompressible isotropic turbulence data from John Hopkins University turbulence database, JHTD [44–46] to train a neural network model inspired by TBNN. Further, we show that appropriate normalization of the input data and a few apt modifications in the network can lead to significant improvements in the alignment characteristics of the predicted output. The predictions made by the TBNN are compared against three different datasets that were not used for training the network: (i) incompressible isotropic turbulence at Taylor Reynolds number of 433, JHTD [44,45]; (ii) incompressible isotropic turbulence at Taylor Reynolds number of 315 (UP Madrid database [47]) [48]; and (iii) nearly incompressible decaying isotropic turbulence at Taylor Reynolds number of 67.4. Further evaluation of the neural network output helps us retrieve ten unique coefficients of the tensor basis of the strain-rate and the rotation-rate tensors, the linear combination of which can be used to predict the pressure-Hessian tensor robustly.

This paper is organized into six sections. In Sec. II, we present the governing equations. In Sec. III, we explain the limitations of the RFD model. In Sec. IV, we present the details of the tensor basis neural network architecture employed for this study. The analysis of the predicted solution from the TBNN is also presented in Sec. IV. Further, in Sec. V, we explain the modifications introduced in the TBNN network and compare its results against the RFD model. Section VI concludes the paper with a brief summary.

II. GOVERNING EQUATIONS

The governing equations of an incompressible flow field are the continuity and the momentum equations:

$$\frac{\partial V_k}{\partial x_k} = 0, \tag{1}$$

$$\frac{\partial V_i}{\partial t} + V_k \frac{\partial V_i}{\partial x_k} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \frac{\mu}{\rho}\frac{\partial^2 V_i}{\partial x_k \partial x_k}, \tag{2}$$

where V_i and x_i represent the velocity and position, respectively, and μ represents the viscosity coefficient. Density and pressure are represented by ρ and p, respectively. The velocity gradient tensor is defined as

$$A_{ij} \equiv \frac{\partial V_i}{\partial x_j}. \tag{3}$$

Taking the gradient of momentum equation (2), the exact evolution equation of A_{ij} can be derived:

$$\rho \frac{DA_{ij}}{Dt} = -\rho A_{ik}A_{kj} - \underbrace{\frac{\partial^2 p}{\partial x_i \partial x_j}}_{\mathcal{P}_{ij}} + \mu \underbrace{\frac{\partial^2 A_{ij}}{\partial x_k \partial x_k}}_{\Upsilon_{ij}}, \tag{4}$$

where $\boldsymbol{\mathcal{P}}$ and $\boldsymbol{\Upsilon}$ represent the pressure Hessian and the viscous Laplacian governing the evolution of the velocity gradient tensor. The rate of change of A_{ij} following a fluid particle is represented using the substantial derivative: $D/Dt(\equiv \partial/\partial t + V_k \partial/\partial x_k)$.

III. EVALUATION OF THE RFD MODEL

The state of the art model for the pressure-Hessian tensor in the context of incompressible flows is the RFDM developed by Chevillard *et al.* [29]. The RFD pressure Hessian ($\boldsymbol{\mathcal{P}}^{\mathbf{RFD}}$) is expressed as

$$\boldsymbol{\mathcal{P}}^{\mathbf{RFD}} = -\frac{\{\boldsymbol{A}^2\}}{\{\boldsymbol{C}_{\tau_k}^{-1}\}}\boldsymbol{C}_{\tau_k}^{-1}, \tag{5}$$

where $\boldsymbol{C}$ is the right Cauchy Green tensor modeled as $\boldsymbol{C}_{\tau k} = e^{\tau_k \boldsymbol{A}} e^{\tau_k \boldsymbol{A}^T}$, τ_k is the Kolmogorov time scale, and the symbol $\{\cdot\}$ represents the trace of the tensor. The pressure Hessian predicted by the RFD model has some inherent inconsistencies as compared to the actual pressure Hessian obtained from DNS. These limitations are listed below:

(1) $\boldsymbol{\mathcal{P}}^{\boldsymbol{RFD}}$ *is always positive definite or negative definite:* It is evident that $\boldsymbol{C}_{\tau k}$ is a positive-definite matrix as it is a product of a real matrix ($e^{\tau_k \boldsymbol{A}}$) and its transpose. Since the inverse of a positive-definite matrix is also positive definite, $\boldsymbol{C}_{\tau_k}^{-1}$ is always positive definite and $\boldsymbol{\mathcal{P}}^{\mathbf{RFD}}$ is guaranteed to be either positive definite or negative definite, depending on the sign of $-\frac{\{A^2\}}{\{C_{\tau_k}^{-1}\}}$. Therefore, the eigenvalues of $\boldsymbol{\mathcal{P}}^{\mathbf{RFD}}$ are either all negative or all positive. This behavior of $\boldsymbol{\mathcal{P}}^{\mathbf{RFD}}$ is nonphysical

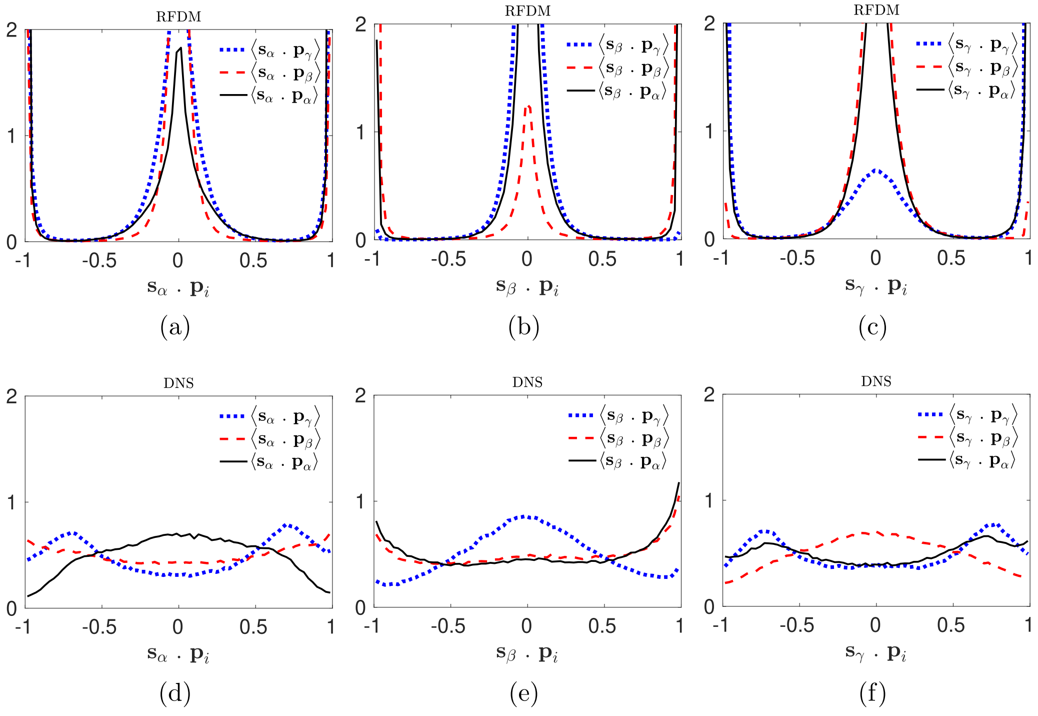

FIG. 1. Alignment of $\mathcal{P}^{\mathcal{RFD}}$ eigenvectors ($\boldsymbol{p_i}$) with $\boldsymbol{S}$ eigenvectors ($\boldsymbol{s_i}$). Here, i ($=\alpha$, β, or γ) denotes the three eigenvectors corresponding to the three eigenvalues $\alpha \geqslant \beta \geqslant \gamma$.

since the governing equations do not impose any such restriction on $\mathcal{P}$. The pressure-Hessian tensor is real symmetric by nature, and hence typically, it has at least one positive and one negative eigenvalue most of the time.

(2) *In strain-dominated regions, the eigenvectors of $\mathcal{P}^{RFD}$ coincide with the strain-rate eigenvectors:* As discussed above, $\mathcal{P}^{\mathbf{RFD}}$ is always either negative definite or positive definite. Further, it is evident that if $\boldsymbol{A}$ is close to being symmetric (strain dominant, $\boldsymbol{A} \approx \boldsymbol{S}$), the eigenvectors of $\mathcal{P}^{\mathbf{RFD}}$ will be approximately parallel or perpendicular to the eigenvectors of $\boldsymbol{S}$ itself. Hence, in strain-dominated regions $\mathcal{P}^{\mathbf{RFD}}$ is expected to show biased alignment towards the strain-rate eigenvectors which is nonphysical. In order to verify this claim, we show the alignment of the eigenvectors of $\mathcal{P}$ and $\mathcal{P}^{\mathbf{RFD}}$ with strain-rate eigenvectors in Fig. 1. In Figs. 1(a)–1(c) we show the PDF of the alignment of eigenvectors of $\mathcal{P}^{\mathbf{RFD}}$ with strain-rate eigenvectors. Further, in Figs. 1(d)–1(f) we show alignment of $\mathcal{P}$ eigenvectors with $\boldsymbol{S}$ eigenvectors for comparison. It can be observed that for a large percentage of particles, $\mathcal{P}^{\mathbf{RFD}}$ eigenvectors are either parallel or perpendicular to the S eigenvectors [Figs. 1(a)–1(c)]. On the other hand, the eigenvectors of $\mathcal{P}$ obtained from DNS show no such alignment tendencies as shown by $\mathcal{P}^{\mathbf{RFD}}$ eigenvectors.

IV. NEURAL NETWORKS BASED MODELING

It is evident from the discussion in the previous section that the functional relationship between the pressure-Hessian and local velocity gradient tensor, if any, is far too complex to be addressed by simple algebraic models. In general, the evolution of pressure Hessian of individual fluid elements is expected to be governed by higher derivatives of velocity and pressure. Nevertheless, in this work, we intend to explore the maximum potential of local velocity gradients to describe the pressure Hessian accurately. For this purpose, we employ deep neural networks. Given, a sufficiently large

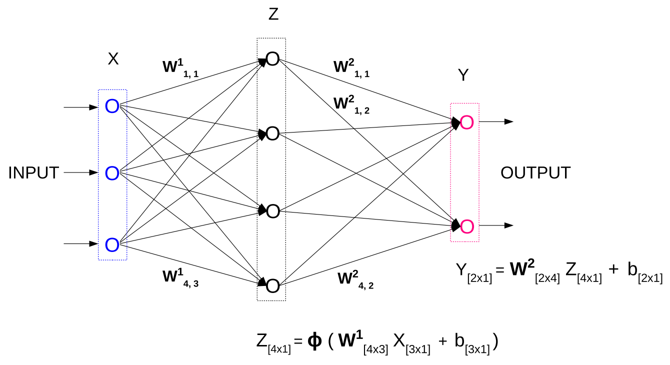
Z
X
Y
W1 1, 1
W2 1, 1
W2 1, 2
INPUT
OUTPUT
W1 4, 3
W2 4, 2
Y[2x1] = W2[2x4] Z[4x1] + b[2x1]
Z[4x1] = ϕ (W1[4x3] X[3x1] + b[3x1])

FIG. 2. Schematic of the feedforward densely connected neural network. $W^{(i)}$ and $b^{(i)}$ are the learnable parameters called weight matrix and the bias vector of the ith layer, respectively. ϕ is a nonlinear activation function.

network and training data, neural networks can potentially find some functional relationship (if any) existing between the quantities of interest. With this motivation, we resort to neural networks to provide a better mapping between the pressure-Hessian and the velocity-gradient tensors. In Sec. IV A, we provide a brief overview of artificial neural networks. In Sec. IV B, we provide a brief overview of the tensor basis neural network [40]. Further, in Sec. IV C, we present the details of the training procedure for the TBNN model. Finally, in Sec. IV D, we test the performance of the trained model.

A. Artificial neural networks

In this work, we employ a feedforward densely connected artificial neural network (ANN). In such an ANN, there are various layers, each layer consisting of a number of nodes. All the nodes of a particular layer of an ANN are connected to the nodes of the adjacent layers in a densely connected ANN. In Fig. 2, we show a basic representation of a simple densely connected feedforward ANN. In this illustration, X is the input layer, Z is a hidden layer, and Y is the output layer. In a deep ANN, there can be a large number of hidden layers. However, in Fig. 2, we have only shown one hidden layer (Z) for simplicity. Each connection in an ANN is associated with a weight. These weights are represented in the form of a matrix (W^i) containing weights of all connection from the ith layer to the $(i+1)$th layer. Further, there is also a bias vector associated with these connections.

The node values of the $(i+1)$th layer are calculated as a linear combination of the output of the ith layer and the corresponding weights followed by a nonlinear activation function ϕ:

$$Z^{i+1} = \phi(\mathbf{W}^i Z^i + b^i). \tag{6}$$

For the output layer, no activation function is used. The objective function (J) of the network (also called the loss function) estimates the difference between the output of the network and the ground truth. Mean square error is a common choice of the loss function:

$$J = \frac{1}{2m}\sum_{i=1}^{i=m}\left(Y_i - Y_i^{\text{true}}\right)^2, \tag{7}$$

where m is the number of training examples used to train the neural network.

The weight matrices and the bias vectors form the learnable parameters (parameters to be optimized) of the neural network. The loss function (J) is minimized using any appropriate gradient descent based optimizer [49]. As an outcome of this optimization process, the most optimal values of the weights and the biases are discovered by the network. This process is termed as the training phase of the neural network. The trained network's performance is validated on a separate held-out dataset other than the one used for training. There are standard ways to improve the performance of the network, in case of any underfitting or overfitting issues. The reader is referred to Goodfellow *et al.* [49] for further details on feedforward densely connected ANNs.

In this work, as starting baselines, we experimented with a simple dense, fully connected network (FCN) with different hyperparameters and activation functions. We also experimented with different ways to represent the input and the output tensors by expressing them locally in the principal coordinate system of strain rate. In these simple baselines, we observed overfitting issues even with using standard regularization techniques. Further, in these baselines, we find that the correct alignment statistics are not recovered between the predicted pressure-Hessian and the known velocity gradient tensor. To overcome these issues, we resorted to a more sophisticated neural network architecture called the tensor basis neural network [40], which is known to be robust for mapping tensors.

B. Tensor basis neural network architecture

In this work, we employ the TBNN developed by Ling *et al.* [40]. The architecture design of this network is inspired by the work of [50], where the author derived the ten trace-free integrity basis ($\boldsymbol{T}^i$) and five independent invariants (λ^i) of the strain-rate ($\boldsymbol{S}$) and the rotation-rate ($\boldsymbol{R}$) tensors for incompressible flows. These tensor bases and invariants are listed below:

$$\begin{gathered}\boldsymbol{T}^1=\boldsymbol{S},\quad \boldsymbol{T}^2=\boldsymbol{SR}-\boldsymbol{RS},\quad \boldsymbol{T}^3=\boldsymbol{S}^2-\tfrac{1}{3}\boldsymbol{I}\{\boldsymbol{S}^2\},\quad \boldsymbol{T}^4=\boldsymbol{R}^2-\tfrac{1}{3}\boldsymbol{I}\{\boldsymbol{R}^2\},\\ \boldsymbol{T}^5=\boldsymbol{RS}^2-\boldsymbol{S}^2\boldsymbol{R},\quad \boldsymbol{T}^6=\boldsymbol{R}^2\boldsymbol{S}+\boldsymbol{SR}^2-\tfrac{2}{3}\boldsymbol{I}\{\boldsymbol{SR}^2\},\quad \boldsymbol{T}^7=\boldsymbol{RSR}^2-\boldsymbol{R}^2\boldsymbol{SR},\quad \boldsymbol{T}^8=\boldsymbol{SRS}^2-\boldsymbol{S}^2\boldsymbol{RS},\\ \boldsymbol{T}^9=\boldsymbol{R}^2\boldsymbol{S}^2+\boldsymbol{S}^2\boldsymbol{R}^2-\tfrac{2}{3}\boldsymbol{I}\{\boldsymbol{S}^2\boldsymbol{R}^2\},\quad \boldsymbol{T}^{10}=\boldsymbol{RS}^2\boldsymbol{R}^2-\boldsymbol{R}^2\boldsymbol{S}^2\boldsymbol{R};\end{gathered}\tag{8}$$

$$\lambda^1=\{\boldsymbol{S}^2\},\quad \lambda^2=\{\boldsymbol{R}^2\},\quad \lambda^3=\{\boldsymbol{S}^3\},\quad \lambda^4=\{\boldsymbol{R}^2\boldsymbol{S}\},\quad \lambda^5=\{\boldsymbol{R}^2\boldsymbol{S}^2\}.\tag{9}$$

The symbol $\{\cdot\}$ represents the trace of the tensor. A linear combination of these ten tensor bases ($\boldsymbol{T}^i$) can represent any trace-free tensor that is directly derived from $\boldsymbol{S}$ and $\boldsymbol{R}$.

In Fig. 3, we present a schematic of the TBNN network. The TBNN increases the representation power of the neural network by embedding knowledge of tensor basis ($\boldsymbol{T}^i$) and invariants (λ^i) in the network itself. The TBNN network takes advantage of the Caley-Hamilton theorem, which states that any function derived from a given tensor alone can be expressed as a linear combination of the integrity basis [51] of the given tensors. The predictions made by the TBNN network are basically a linear combination of the integrity basis ($\boldsymbol{T}^i$) of the input tensors. Hence, the TBNN network explores the full spectrum of all the mappings that any input tensor can offer, by enforcing the output of the network to be a linear combination of its integrity basis. Further, the TBNN network has embedded rotational invariance, which ensures that the predictions made by the TBNN network are independent of the orientation of the coordinate system. If the input tensors are expressed in a rotated-coordinate system, the predicted output will also get rotated accordingly. Hence, the TBNN network predicts the same output tensor irrespective of the orientation of the coordinate system.

For incompressible flows, we can obtain the exact expression for the trace of the pressure Hessian from Eq. (4) (Poisson's equation):

$$\{\mathcal{P}\}=-\rho A_{ik}A_{ki}.\tag{10}$$

Since the trace of the pressure Hessian can be directly derived from the velocity gradient information, we do not need to model it. Now, the ten trace-free integrity bases (T_i) can be readily used to model the trace-free (anisotropic) part of the pressure Hessian using the TBNN. We use the symbol $\mathcal{P}_{tf}$ to denote the anisotropic part of $\mathcal{P}$. To find the relevant mapping between velocity gradient tensor and $\mathcal{P}_{tf}$ the ten coefficients (C^i) corresponding to the ten integrity bases ($\boldsymbol{T}^i$) need to be modeled. The five invariants (λ^i) of $\boldsymbol{S}$ and $\boldsymbol{R}$ form the primary input of the TBNN. The output of the last layer of the network yields the ten coefficients C^i. A secondary input containing the ten tensor basis $\boldsymbol{T}^i$ (called tensor layer) is fed to the last layer of the network. Finally, a dot product between

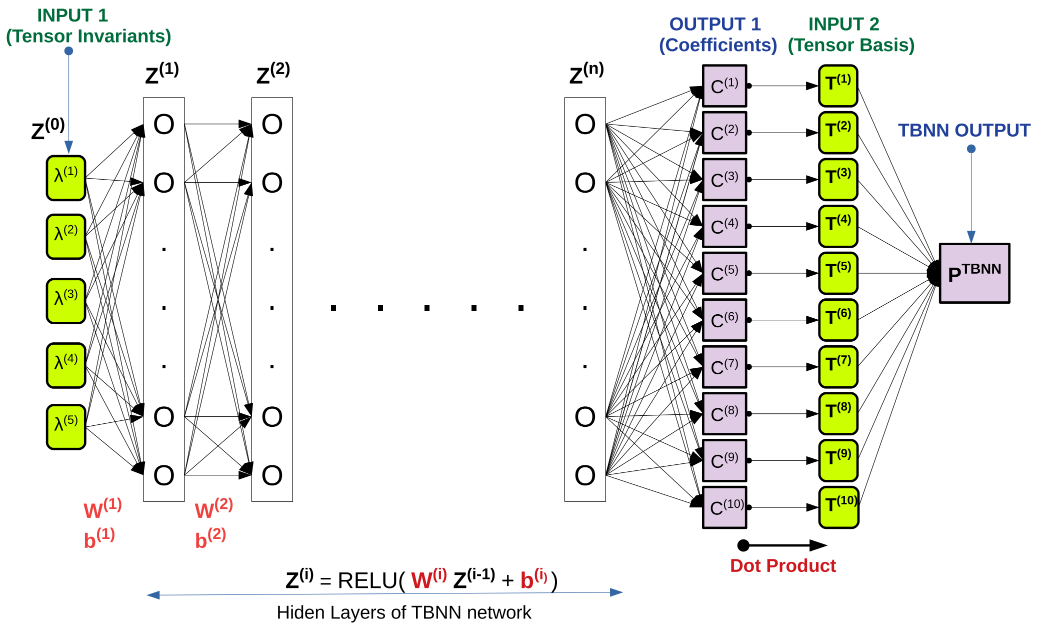

FIG. 3. Schematic of the TBNN network. $W^{(i)}$ and $b^{(i)}$ are the weight matrix and the bias vector of the ith layer. Rectified linear unit (RELU) nonlinear activation function is used. Both $W^{(i)}$ and $b^{(i)}$ are the learnable parameters of the neural network, which are optimized using the RMSprop optimizer [54].

the coefficient layer and the tensor layer of the network makes the final output of the network, which can be expressed as

$$\mathcal{P}_{tf}^{\text{TBNN}} = \sum_{i=1}^{10} C^i \boldsymbol{T}^i. \tag{11}$$

The cost function of the network can be expressed as

$$J = \frac{1}{2m} \sum_{j=1}^{m} \left[\left\| \left(\mathcal{P}_{tf}^{\text{TBNN}} - \mathcal{P}_{tf} \right)_j \right\|_F \right]^2, \tag{12}$$

where m is the number of training examples required to train the TBNN and the symbol $\| \cdot \|_F$ represents the Frobenius norm.

C. Training of the neural network

The details of the turbulence datasets employed in this study are mentioned in Table I. The employed tensor basis neural network (TBNN) model is trained using data from an isotropic incompressible flow field at the Reynolds number of 433 (dataset A, Table I). This data is taken from the John Hopkins University's Turbulence database [44,45] available online [46]. The open source library Keras [52] with TensorFlow backend is used for training the TBNN model. The velocity gradient tensor and pressure-Hessian information are extracted from the database at a particular time instant. The velocity gradient tensor is nondimensionalized with the mean value of the Frobenius norm of the whole sample of 262,144 data points. No further normalization was used for the derived tensor basis ($\boldsymbol{T}^i$) and invariants (λ^i). A total number of 262 144 unique data points are extracted from the flow field. Out of these 262 144 data points, 236 544 points are used for training the network, while the remaining 25 600 data points are reserved for the cross-validation

TABLE I. Parameters of isotropic turbulence datasets. [Symbols: (i) Re_λ: Taylor Reynolds number; (ii) M_t: turbulent Mach number; (iii) τ_k: Kolmogorov timescale; (iv) η: Kolmogorov length scale; (v) ν: dynamic viscosity; (vi) λ: Taylor microscale; (vii) ϵ: dissipation rate; (viii) $\overline{P_0}$: mean pressure; and (ix) K: turbulent kinetic energy.]

Case	Dataset type	Re_λ	τ_k	η	ν	λ	ϵ	$\overline{P}$	K
A	Training/Testing	433	0.0446	0.00287	0.000185	0.118	0.0928	0	0.695
B	Testing	67.4	0.0113	0.00077	0.1655	0.182	2.801×10^5	8.801×10^4	5.597×10^3
C	Testing	315	0.0041	0.0247	0.00066	0.1415	0.10947	0	3.2933

of the predicted solution. The training data is distributed at random into 924 minibatches of 256 data points each at the beginning of every epoch. One epoch is one complete pass through the training dataset overall minibatches. Hence, one epoch accounts for 924 iterations of the training cycle.

A deep network with 11 hidden layers and a combination of 50, 150, 150, 150, 150, 300, 300, 150, 150, 150, 100 in the consecutive hidden layers was found to yield the best performance of all the combinations that were tested. We use the Glorot normal initialization [53] for weight matrices and RELU (rectified linear unit) nonlinear activation function for the hidden layers. The RMSprop optimizer [54], with a learning rate of 1.0×10^{-6} was used to train the network. The training was stopped when the value of cost function J became stagnant. The minimum value of the training cost and the cross-validation cost recorded while training was 4.1×10^{-4} and 5.4×10^{-4}, respectively. In Fig. 4, we show the training and cross-validation cost as a function of a number of training epochs. The cross-validation cost did not show any significant rise during the training process. A low dropout rate of 10% was used to facilitate ensemble learning in the network. There was no gain in model performance with a further increase in data size and network depth.

D. Testing of the trained network

The primary testing of the trained TBNN model was performed on a separate testing dataset (other than training and validation data) of isotropic turbulence (extracted from dataset A, Table I). The relative Frobenius-norm error of the pressure Hessian obtained from the trained model on the

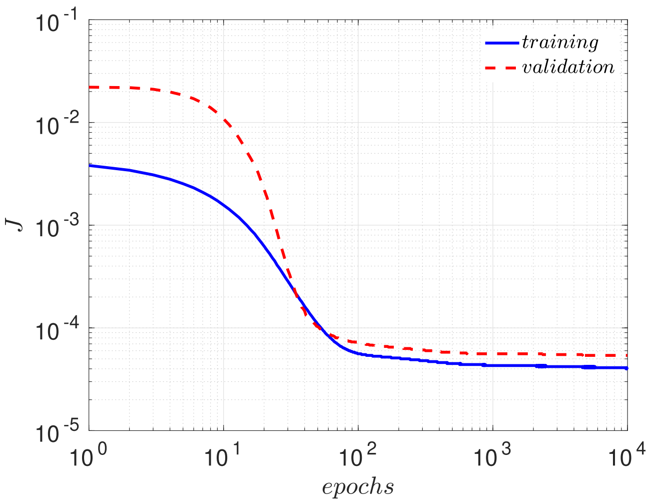

FIG. 4. Decay of cost function during training for TBNN. Minibatch size = 256; 1 epoch = 924 iterations of the optimizer.

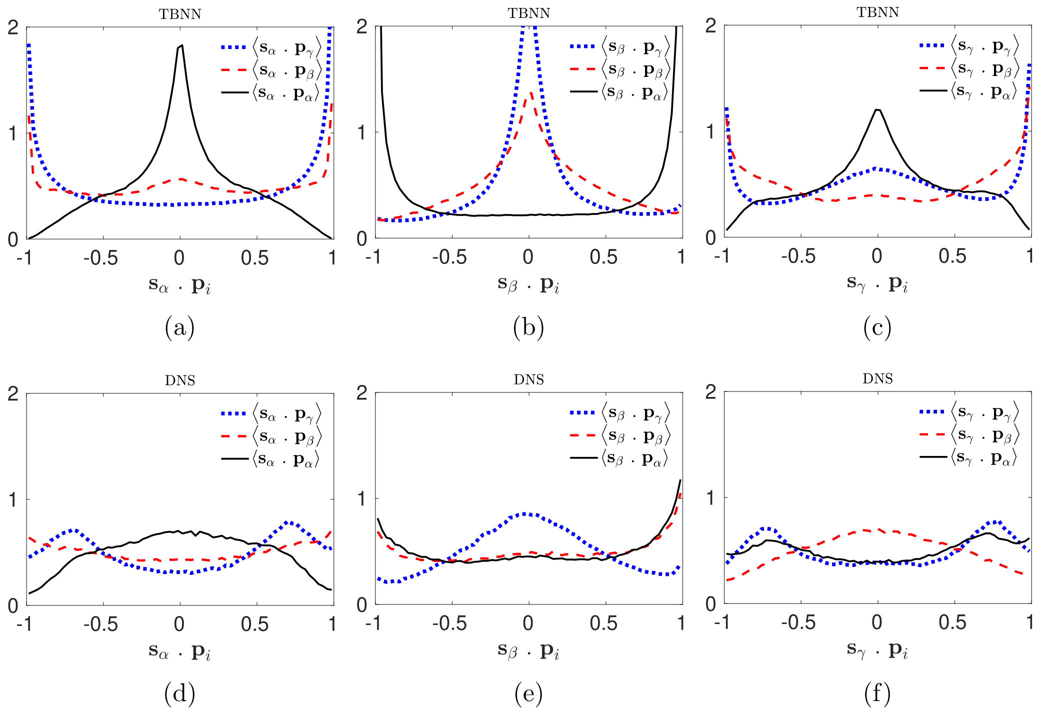

FIG. 5. Alignment of $\mathcal{P}^{TBNN}$ eigenvectors ($\boldsymbol{p}_i$) with $\boldsymbol{S}$ eigenvectors ($\boldsymbol{s}_i$). Here, i (=α, β, or γ) denotes the three eigenvectors corresponding to the three eigenvalues $\alpha > \beta > \gamma$ (dataset A, Table I).

testing dataset was found to be 0.6491. On the same dataset, an error of 0.7764 was obtained by the RFDM model. Hence, in terms of elementwise mean squared error comparison, the accuracy of the trained TBNN model is comparable to the existing RFDM model. However, just the elementwise comparison is not a wise comparison metric for comparing tensorial quantities. We have earlier (in Fig. 1) seen that the RFD model fails to capture the alignment statistics with the strain-rate tensor. In Fig. 5, we present the alignment of the pressure-Hessian eigenvectors predicted by the TBNN [Figs. 5(a)–5(c)] with the strain-rate eigenvectors compared against that obtained from DNS [Figs. 5(d)–5(f)]. We observe that although the alignment statistics [Figs. 5(a)–5(c)] have improved as compared to the RFD model results [Figs. 1(a)–1(c)], the obtained statistics are still far off from that obtained from DNS.

V. MODIFIED NEURAL NETWORK ARCHITECTURE

We have observed that TBNN is unable to capture the alignment statistics of the pressure-Hessian tensor. It implies that assuming the pressure Hessian to lie on the tensor basis of the strain-rate and the rotation-rate tensors is not an appropriate modeling assumption. Constraining the network to obey the tensor invariance properties restricts us to using only global normalization of the input tensors. However, the velocity gradients in a turbulent flow field are known to be highly intermittent. Hence, global normalization of the input tensors might not be an effective strategy. The learning of important feature mappings by a neural network relies heavily on effective normalization strategies. At this juncture, we performed several experiments on the TBNN by choosing various normalization strategies which allow TBNN to deviate from its tensor invariance characteristics. We found out through trial and error that normalizing the tensor basis such that all its elements are scaled between

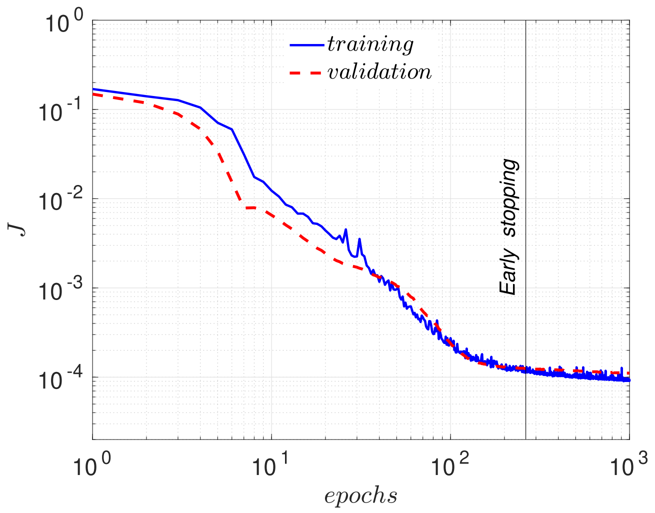

FIG. 6. Decay of cost function during training for the modified TBNN. Minibatch size = 256; 1 epoch = 924 iterations of the optimizer.

[0, 1], yields tremendous improvement in network output. Two matrices $\mathbf{F^{(i)}}$ and $\mathbf{G^{(i)}}$ are used to scale the tensor basis:

$$F_{pq}^{(i)} = \max\left(T_{pq}^{(i)}\right), \tag{13}$$

$$G_{pq}^{(i)} = \min\left(T_{pq}^{(i)}\right). \tag{14}$$

Using $\mathbf{F^{(i)}}$ and $\mathbf{G^{(i)}}$ the tensor basis can be appropriately scaled using the following relationship:

$$\boldsymbol{T}^{(i)'} = (\boldsymbol{T}^{(i)} - \boldsymbol{G}^{(i)}) \oslash (\boldsymbol{F}^{(i)} - \boldsymbol{G}^{(i)}), \tag{15}$$

where symbol $\oslash$ represents the Hadamard division between the two tensors. With this normalization, the network loses most of the properties of the original TBNN. However, it leads to significant improvements in alignment statistics of the predicted output.

We employ the modified network with the same settings (viz., the number of hidden layers, neurons per layer, activation function, learning rate, etc.) as used with the original TBNN network. In Fig. 6, we show the learning curve obtained while training the modified TBNN. We use an early stopping criterion while training the network, at the point when the validation-loss curve becomes almost flat (no further decline with increasing epochs).

A. Testing modified TBNN for isotropic turbulence flow

In Fig. 7 we show the alignment statistics obtained on the testing dataset A (Table I) with the modified TBNN. This data, although extracted from the same database (case A, Table I) used for training, consists of separate data points other than those used for training. We observe that modified TBNN predictions [Figs. 7(a)–7(c)] demonstrate excellent alignment statistics as compared to that obtained from DNS [Figs. 7(d)–7(f)]. Although this testing dataset is extracted at different grid locations other than the training dataset, it still has the same Reynolds number as the training dataset (433). To make a better judgment of the generalization of the learned pressure-Hessian mapping, we scrutinize the performance of the trained model for the isotropic turbulence dataset at two different Reynolds numbers of 67.4 (dataset B, Table I) and 315 (dataset C, Table I).

The testing dataset B is extracted from our own decaying isotropic compressible turbulence simulation [25,55]. Although this data is from a compressible simulation, the turbulent Mach

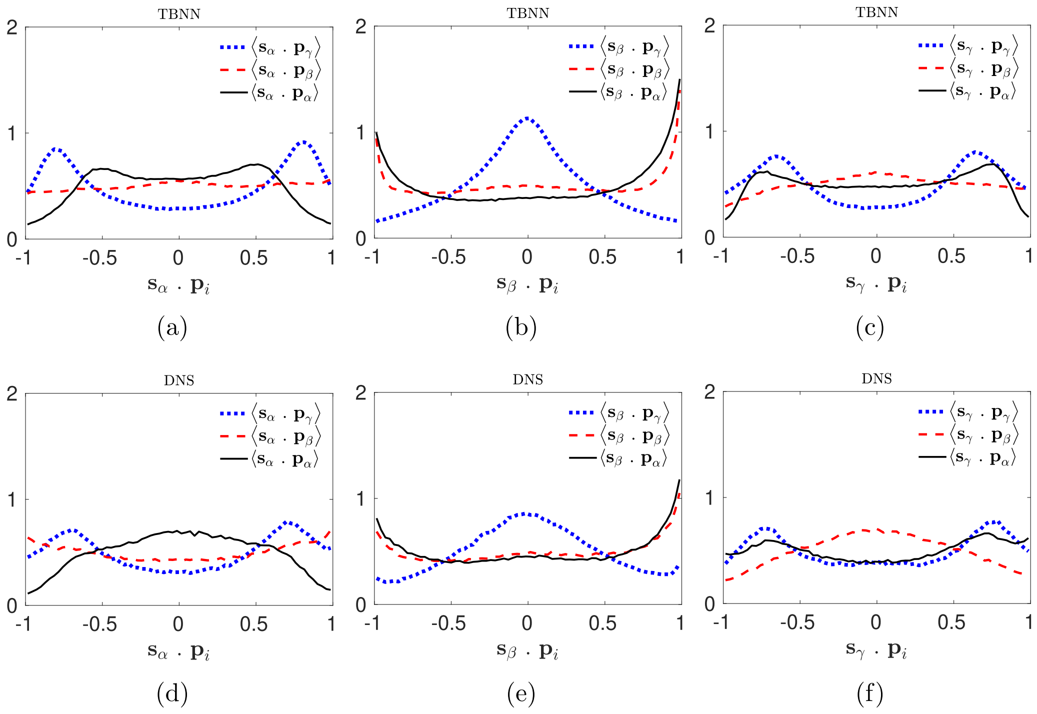

FIG. 7. Alignment of $\mathcal{P}^{\mathcal{TBNN}}$ eigenvectors ($\boldsymbol{p_i}$) obtained from modified TBNN with $\boldsymbol{S}$ eigenvectors ($\boldsymbol{s_i}$). Here, i ($=\alpha$, β, or γ) denotes the three eigenvectors corresponding to the three eigenvalues $\alpha \geqslant \beta \geqslant \gamma$ (dataset A, Table I).

number is quite low ($M_t = 0.4$) for any significant compressibility effect to be observed. Further, this data is extracted at a reference time of $t = 2.7t_0$, where t_0 is the eddy turnover time. Since it is a decaying turbulent flow, the Reynolds number of the flow has decayed to a value of 67.4 and the turbulent Mach number to a further lower value of 0.3 at this time instant of $t = 2.7t_0$. In Fig. 8, we plot the alignment statistics obtained from the learned modified TBNN model for testing dataset B. We find that statistics obtained from modified TBNN show good agreement with that obtained from the DNS dataset (Fig. 8).

The testing dataset C (Table I) is extracted from the UP Madrid turbulence dataset of incompressible isotropic turbulence simulation at the Reynolds number of 315 [48]. In Fig. 9 we plot the alignment statistics obtained from the learned modified TBNN model for testing dataset C. Since this database does not provide pressure information, we are unable to provide the exact pressure Hessian alignment statistics for this dataset. However, it can be observed from both Fig. 7 and Fig. 8 as well as from the alignment statistics reported by previous works of Kalelkar [6] that the statistics of the alignment of pressure-Hessian and strain-rate eigenvectors are fairly independent of the Reynolds number. We can expect the alignment statistics obtained from DNS dataset C (Fig. 9) to reflect the general trend as observed for other isotropic turbulence datasets as well. Hence, we can conclude that the trained modified TBNN has learned key physical features that can be generalized for an isotropic turbulent flow independent of its Reynolds number.

B. Predicted coefficients by the modified network

In Fig. 10, we show the scatter plot of the coefficients predicted by the modified TBNN. We observe that each of these ten coefficients ($C^{(i)}$) have negligible variance. The overall distribution can effectively be replaced by the mean value of the distribution of each of the coefficients. Further,

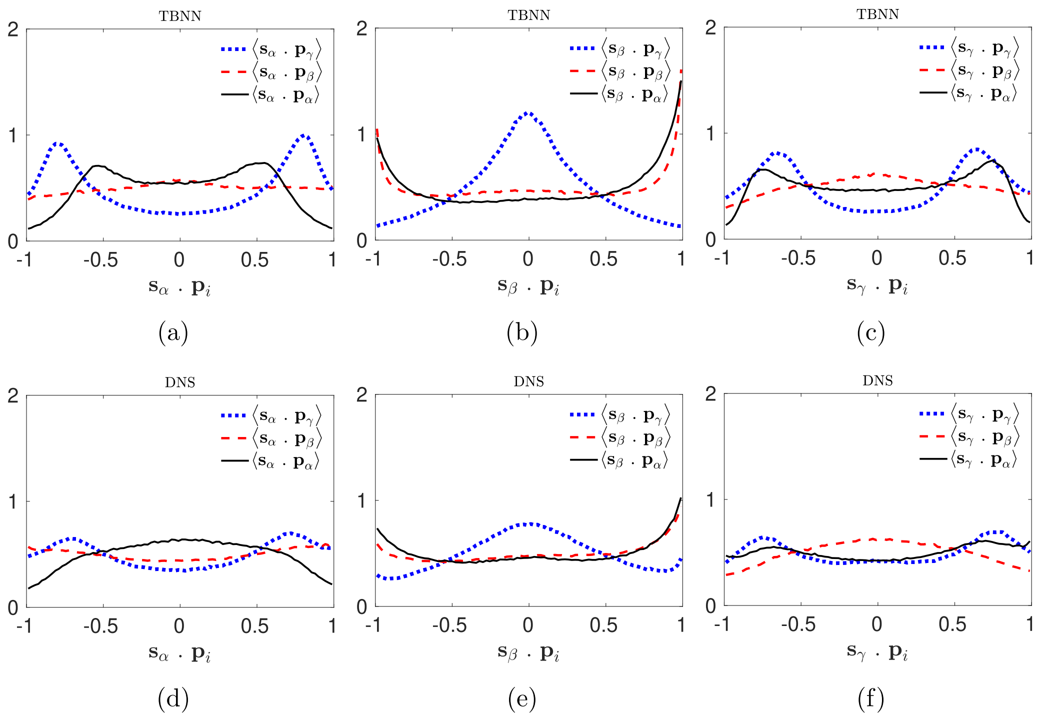

FIG. 8. Alignment of $\mathcal{P}^{\mathcal{TBNN}}$ eigenvectors ($\boldsymbol{p_i}$) obtained from modified TBNN with $\boldsymbol{S}$ eigenvectors ($\boldsymbol{s_i}$). Here, i ($=\alpha$, β, or γ) denotes the three eigenvectors corresponding to the three eigenvalues $\alpha > \beta > \gamma$. (Decaying isotropic turbulence testing dataset B, Table I, initial Reynolds number 250, and turbulent Mach number of 0.4.) Note that the field is extracted at $t = 2.7t_0$, where t_0 is the eddy turnover time. At this instant, the Reynolds number of the flow has decayed to the value of 67.4 and the turbulent Mach number is 0.3.

we find that by using the mean value of the coefficients, we retrieve the same statistics as obtained by passing the velocity gradient information through the modified TBNN.

With this revelation, it is no longer required to use the trained network for pressure-Hessian estimation. Rather, we can just use a very simple process for pressure-Hessian prediction:

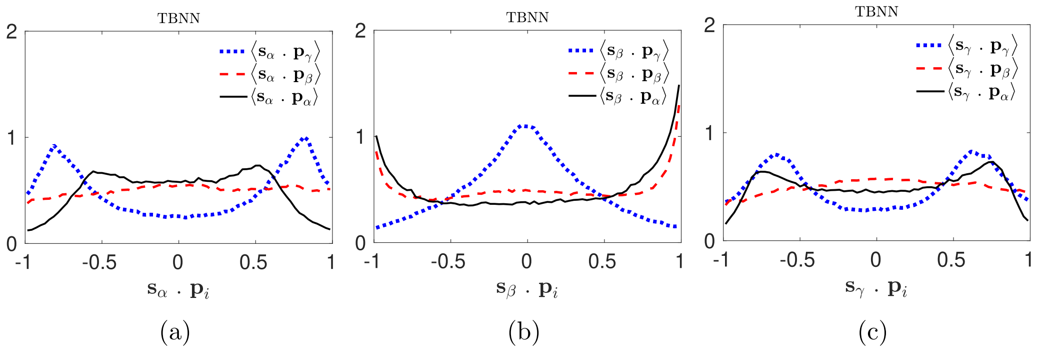

FIG. 9. Alignment of $\mathcal{P}^{\mathcal{TBNN}}$ eigenvectors ($\boldsymbol{p_i}$) obtained from modified TBNN with $\boldsymbol{S}$ eigenvectors ($\boldsymbol{s_i}$). Here, i ($=\alpha$, β, or γ) denotes the three eigenvectors corresponding to the three eigenvalues $\alpha \geqslant \beta \geqslant \gamma$ (dataset C, Table I).

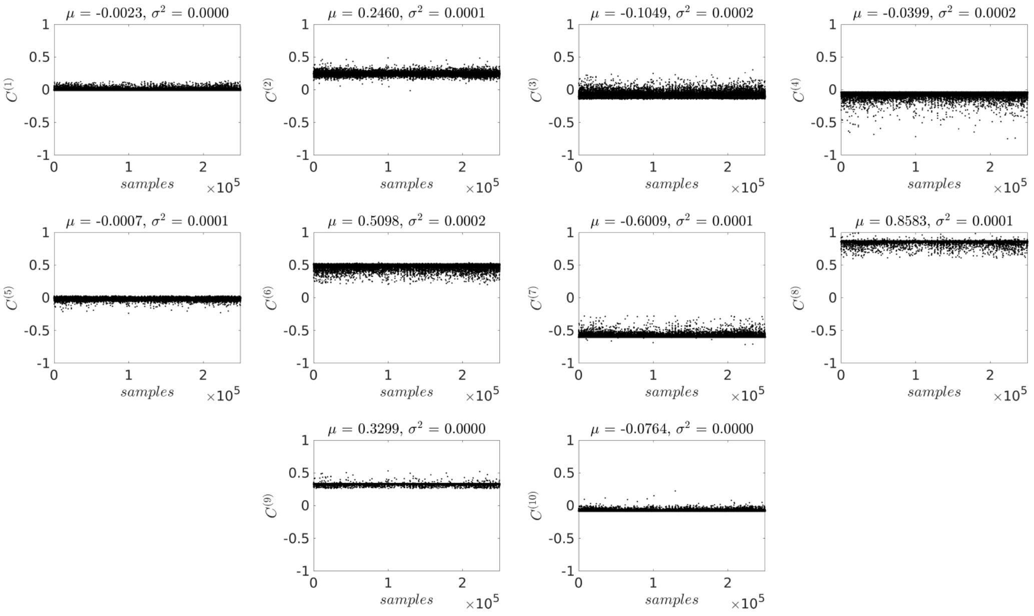

FIG. 10. Scatter plot of the ten coefficients predicted by the modified TBNN.

(1) Nondimensionalize the velocity gradient tensor, using the mean value of the Frobenius norm of the whole sample.

(2) Calculate the ten tensor bases and five independent invariants of strain-rate and rotation-rate tensors using Eqs. (8) and (9).

(3) Normalize the tensor basis using the scaling matrices used for the trained network (details in the Appendix).

(4) Take a linear combination of the tensor basis using the mean value of the coefficients obtained from the trained network. This would yield the modeled normalized pressure Hessian (refer to the Appendix).

(5) Scale the predicted pressure Hessian back to its original dimensional form, using the same scaling matrices that were used while training the network.

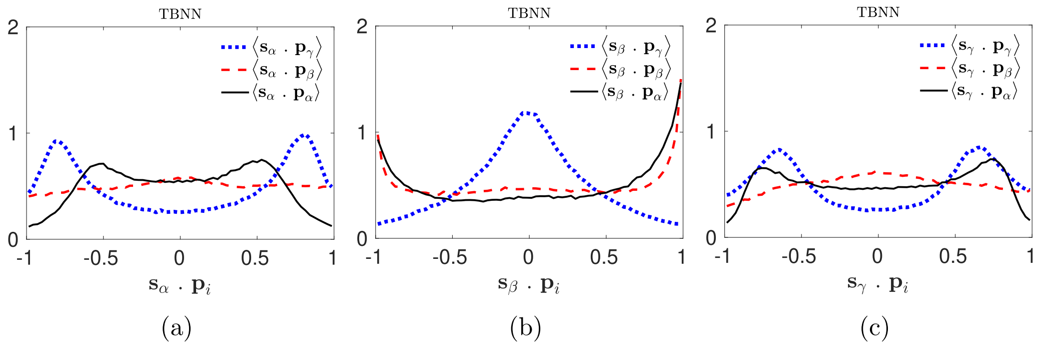

FIG. 11. Alignment of $\mathcal{P}^{\mathcal{TBNN}}$ eigenvectors ($\boldsymbol{p_i}$) obtained using mean value of coefficients (using the methodology shown in the Appendix) with $\boldsymbol{S}$ eigenvectors ($\boldsymbol{s_i}$) for testing dataset B (Table I). Here, i (=α, β, or γ) denotes the three eigenvectors corresponding to the three eigenvalues $\alpha \geqslant \beta \geqslant \gamma$. More details in the caption of Fig. 8.

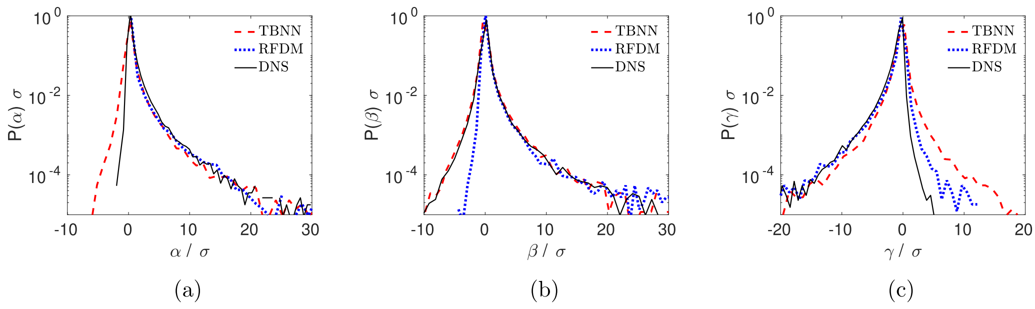

FIG. 12. PDF of eigenvalues of pressure-Hessian tensor (a) α eigenvalue, (b) β eigenvalue, and (c) γ eigenvalue. The eigenvalues are normalized by the respective standard deviation (σ).

(6) Enforce the predicted solution to have the desired trace (since the trace of $\boldsymbol{P}$ is the same as the trace of $-\boldsymbol{A}^2$). The complete details of the step-by-step process for calculation of the modeled pressure-Hessian tensor are presented in the Appendix. For illustration, we show the statistics obtained by using the mean value of the coefficients using the approach shown in the Appendix for dataset B (Table I) in Fig. 11. Excellent agreement with the statistics obtained directly using the modified TBNN (Fig. 8) is observed.

At this juncture, we also test the performance of our method (Appendix) in predicting the PDF of pressure-Hessian eigenvalues. These statistics have been reported in the previous notable works of Kalelkar [6] and Danish *et al.* [56] on the pressure-Hessian tensor. In Fig. 12, we show the PDF of eigenvalues ($\alpha > \beta > \gamma$) of the pressure Hessian predicted from our model against the DNS dataset (dataset A, Table I) and the RFDM model.

In Fig. 12(a), we show the PDF of the α eigenvalue. It can be seen thay the PDF obtained from our model is in good agreement with the DNS result and performs better than the RFDM model. Similarly, in Fig. 12(b), we observe that the PDF of β eigenvalues obtained from our model is in excellent agreement with the DNS result. In Fig. 12(c), although both RFDM and our model try to capture the shape of the PDF on both the negative and the positive side of the plane, RFDM performs slightly better than our model. However, overall we see that our model exhibits good performance while capturing the PDF of pressure-Hessian eigenvalues.

VI. CONCLUSIONS

In this work, we develop and evaluate a machine learning model for the pressure-Hessian tensor in isotropic incompressible turbulence using velocity gradient information. This model for the pressure-Hessian tensor can be used to develop a closure model for the Lagrangian velocity gradient evolution equation, which in turn can lead to the closure of the set of equations in the Lagrangian PDF method for solving turbulent flows. To achieve a functional mapping between pressure-Hessian and velocity gradient tensors, we employ a TBNN architecture [40]. This neural network is trained on high-resolution isotropic turbulence data at a Reynolds number of 433. With the help of TBNN, the pressure-Hessian tensor is modeled in terms of the trace-free and symmetric tensor basis of the strain-rate and the rotation-rate tensors.

The performance of the machine learning based model is evaluated against direct numerical simulation data for incompressible turbulence. We find that the accuracy (relative $L2$-norm error) of the predicted pressure Hessian by TBNN is comparable to that obtained from the RFD model. However, only a marginal improvement in the alignment statistics in the TBNN prediction is obtained as compared to the RFD model. We report that by scaling the tensor basis of strain-rate and rotation-rate tensors such that each element of the basis lies between [0, 1], the predicted output of the neural network yields excellent alignment statistics with the strain-rate tensor for isotropic turbulent flows at different Reynolds number. With this finding, we come to the conclusion that there does exist a relevant physical mapping between pressure-Hessian and velocity gradients which force their eigenvectors to align appropriately with each other. This mapping is found to be fairly independent of the Reynolds number for isotropic turbulence.

Finally, we find that the distribution of the coefficients of the tensor basis obtained from the neural network has a negligible variance. In the light of this finding, we have been able to identify ten unique coefficients of the tensor basis, the linear combination over which can be used to model the pressure-Hessian tensor directly. The pressure Hessian predicted by the proposed method shows good alignment statistics with the strain-rate eigenvectors. Finally, we test our model in its ability to predict the PDF of the pressure-Hessian eigenvalues. We find that our model indeed shows good agreement with DNS results.

ACKNOWLEDGMENT

The authors acknowledge the computational support provided by the High-Performance Computing Center (HPC) of the Indian Institute of Technology Delhi, India.

APPENDIX

We present a step-by-step process for the modeled pressure-Hessian calculation based on mean values of the coefficients derived from the modified TBNN.

(1) *Nondimensionalize the strain-rate and rotation-rate tensors*:

$$\epsilon = \langle\sqrt{A_{ij}A_{ij}}\rangle,$$

where $\langle\cdot\rangle$ represents the mean over the whole sample.

$$\mathbf{S} = \frac{\mathbf{A}+\mathbf{A}^T}{2\epsilon}, \quad \mathbf{R} = \frac{\mathbf{A}-\mathbf{A}^T}{2\epsilon}.$$

(2) *Find the ten tensor basis and the five independent invariants*:

$$\begin{aligned}
&\boldsymbol{T}^1 = \boldsymbol{S}, \quad \boldsymbol{T}^2 = \boldsymbol{SR}-\boldsymbol{RS}, \quad \boldsymbol{T}^3 = \boldsymbol{S}^2 - \tfrac{1}{3}\boldsymbol{I}\{\boldsymbol{S}^2\}, \quad \boldsymbol{T}^4 = \boldsymbol{R}^2 - \tfrac{1}{3}\boldsymbol{I}\{\boldsymbol{R}^2\},\\
&\boldsymbol{T}^5 = \boldsymbol{RS}^2 - \boldsymbol{S}^2\boldsymbol{R}, \quad \boldsymbol{T}^6 = \boldsymbol{R}^2\boldsymbol{S} + \boldsymbol{SR}^2 - \tfrac{2}{3}\boldsymbol{I}\{\boldsymbol{SR}^2\},\\
&\boldsymbol{T}^7 = \boldsymbol{RSR}^2 - \boldsymbol{R}^2\boldsymbol{SR}, \quad \boldsymbol{T}^8 = \boldsymbol{SRS}^2 - \boldsymbol{S}^2\boldsymbol{RS},\\
&\boldsymbol{T}^9 = \boldsymbol{R}^2\boldsymbol{S}^2 + \boldsymbol{S}^2\boldsymbol{R}^2 - \tfrac{2}{3}\boldsymbol{I}\{\boldsymbol{S}^2\boldsymbol{R}^2\}, \quad \boldsymbol{T}^{10} = \boldsymbol{RS}^2\boldsymbol{R}^2 - \boldsymbol{R}^2\boldsymbol{S}^2\boldsymbol{R};\\
&\lambda^1 = \{\boldsymbol{S}^2\}, \quad \lambda^2 = \{\boldsymbol{R}^2\}, \quad \lambda^3 = \{\boldsymbol{S}^3\}, \quad \lambda^4 = \{\boldsymbol{R}^2\boldsymbol{S}\}, \quad \lambda^5 = \{\boldsymbol{R}^2\boldsymbol{S}^2\}.
\end{aligned}$$

(3) *Normalize the tensor basis* ($\boldsymbol{T}^{(i)}$):

$$\boldsymbol{T}^{(i)'} = (\boldsymbol{T}^{(i)} - \boldsymbol{G}^{(i)}) \oslash (\boldsymbol{F}^{(i)} - \boldsymbol{G}^{(i)}),$$

where symbol $\oslash$ represents the Hadamard division between the two tensors. $\boldsymbol{F}^{(i)}$ and $\boldsymbol{G}^{(i)}$ represent the matrices used to scale the tensor basis $\boldsymbol{T}^{(i)}$:

$$\boldsymbol{G}^{(1)} = \begin{bmatrix} -3.5709 & -3.0858 & -2.7680 \\ -3.0858 & -3.0346 & -3.0692 \\ -2.7680 & -3.0692 & -4.2937 \end{bmatrix}, \quad \boldsymbol{F}^{(1)} = \begin{bmatrix} 2.7021 & 3.2914 & 2.3884 \\ 3.2914 & 3.3876 & 2.8077 \\ 2.3884 & 2.8077 & 2.8905 \end{bmatrix},$$

$$\boldsymbol{G}^{(2)} = \begin{bmatrix} -25.6104 & -31.8796 & -21.5908 \\ -31.8796 & -30.5056 & -22.2121 \\ -21.5908 & -22.2121 & -28.0164 \end{bmatrix}, \quad \boldsymbol{F}^{(2)} = \begin{bmatrix} 32.0154 & 35.4956 & 20.6083 \\ 35.4956 & 29.8806 & 30.5352 \\ 20.6083 & 30.5352 & 31.7193 \end{bmatrix},$$

$$\boldsymbol{G}^{(3)} = \begin{bmatrix} -9.2866 & -6.4438 & -10.5787 \\ -6.4438 & -6.4369 & -5.4972 \\ -10.5787 & -5.4972 & -8.4430 \end{bmatrix}, \quad \boldsymbol{F}^{(3)} = \begin{bmatrix} 8.2153 & 7.9086 & 7.8492 \\ 7.9086 & 6.0971 & 5.9908 \\ 7.8492 & 5.9908 & 9.3168 \end{bmatrix},$$

$$\boldsymbol{G}^{(4)} = \begin{bmatrix} -14.8512 & -15.3252 & -13.8140 \\ -15.3252 & -11.2078 & -11.7173 \\ -13.8140 & -11.7173 & -13.0263 \end{bmatrix}, \quad \boldsymbol{F}^{(4)} = \begin{bmatrix} 16.6632 & 9.6823 & 12.9011 \\ 9.6823 & 27.8775 & 15.5748 \\ 12.9011 & 15.5748 & 23.0473 \end{bmatrix},$$

$$\boldsymbol{G}^{(5)} = \begin{bmatrix} -54.1672 & -26.9697 & -23.7045 \\ -26.9697 & -43.5902 & -36.9865 \\ -23.7045 & -36.9865 & -30.1412 \end{bmatrix}, \quad \boldsymbol{F}^{(5)} = \begin{bmatrix} 27.5052 & 57.6332 & 33.7410 \\ 57.6332 & 53.5055 & 38.4958 \\ 33.7410 & 38.4958 & 55.2717 \end{bmatrix},$$

$$\boldsymbol{G}^{(6)} = \begin{bmatrix} -119.1511 & -78.0652 & -112.3875 \\ -78.0652 & -186.4133 & -80.4339 \\ -112.3875 & -80.4339 & -116.5743 \end{bmatrix}, \quad \boldsymbol{F}^{(6)} = \begin{bmatrix} 195.0304 & 143.0951 & 108.0702 \\ 143.0951 & 92.4415 & 169.8799 \\ 108.0702 & 169.8799 & 135.2173 \end{bmatrix},$$

$$\boldsymbol{G}^{(7)} = \begin{bmatrix} -737.6056 & -1038.4397 & -640.3922 \\ -1038.4397 & -963.7453 & -551.0073 \\ -640.3922 & -551.0073 & -537.7603 \end{bmatrix}, \quad \boldsymbol{F}^{(7)} = \begin{bmatrix} 780.8498 & 970.6533 & 540.2198 \\ 970.6533 & 808.6133 & 523.5790 \\ 540.2198 & 523.5790 & 924.0598 \end{bmatrix},$$

$$\boldsymbol{G}^{(8)} = \begin{bmatrix} -496.1792 & -376.4132 & -245.1099 \\ -376.4132 & -341.9391 & -345.2818 \\ -245.1099 & -345.2818 & -405.4714 \end{bmatrix}, \quad \boldsymbol{F}^{(8)} = \begin{bmatrix} 390.7658 & 778.0012 & 298.8682 \\ 778.0012 & 658.3773 & 509.5283 \\ 298.8682 & 509.5283 & 444.1750 \end{bmatrix},$$

$$\boldsymbol{G}^{(9)} = \begin{bmatrix} -526.6148 & -345.7790 & -276.1035 \\ -345.7790 & -188.0288 & -293.5551 \\ -276.1035 & -293.5551 & -339.0242 \end{bmatrix}, \quad \boldsymbol{F}^{(9)} = \begin{bmatrix} 381.9239 & 194.5271 & 578.1185 \\ 194.5271 & 197.7692 & 170.3597 \\ 578.1185 & 170.3597 & 598.0897 \end{bmatrix},$$

$$\boldsymbol{G}^{(10)} = \begin{bmatrix} -595.3592 & -1679.5692 & -922.0738 \\ -1679.5692 & -1466.6689 & -674.1032 \\ -922.0738 & -674.1032 & -1079.9475 \end{bmatrix}, \quad \boldsymbol{F}^{(10)} = \begin{bmatrix} 1335.8832 & 1173.0048 & 438.1600 \\ 1173.0047 & 1126.0351 & 671.8583 \\ 438.1600 & 671.8582 & 662.4745 \end{bmatrix}.$$

(4) *Take a linear combination of tensor basis using mean coefficient values*:

$$\mathcal{P}^{\mathbf{TBNN}'} = \sum_{i=1}^{10} C^{i} \boldsymbol{T}^{i'},$$

where C^i are the mean coefficient values predicted by the modified TBNN:

$$C^{(1)} = -0.0023, \quad C^{(2)} = +0.2460, \quad C^{(3)} = -0.1049, \quad C^{(4)} = -0.0400, \quad C^{(5)} = -0.0007,$$
$$C^{(6)} = +0.5098, \quad C^{(7)} = -0.6009, \quad C^{(8)} = +0.8583, \quad C^{(9)} = +0.3299, \quad C^{(10)} = -0.0764.$$

(5) *Scale the predicted pressure Hessian back to its dimensional form*:

$$\mathcal{P}^{\mathbf{TBNN}'} = [\mathcal{P}^{\mathbf{TBNN}'} \circ (\boldsymbol{F_p} - \boldsymbol{G_p}) + \boldsymbol{G_p}]\epsilon^2,$$

where the symbol $\circ$ represents the Hadamard product of two matrices and $\boldsymbol{G_p}$ and $\boldsymbol{F_p}$ are the scaling matrices:

$$\boldsymbol{G}_p = \begin{bmatrix} -26.9693 & -13.2321 & -12.5971 \\ -13.2321 & -26.0595 & -17.4419 \\ -12.5971 & -17.4419 & -24.2304 \end{bmatrix}, \quad \boldsymbol{F}_p = \begin{bmatrix} 19.1816 & 23.7427 & 17.5106 \\ 23.7427 & 27.9531 & 25.3751 \\ 17.5106 & 25.3751 & 20.1042 \end{bmatrix}.$$

(6) *Trace correction step*:

$$\mathcal{P}^{\mathrm{TBNN}} = \mathcal{P}^{\mathrm{TBNN}'} - \{\mathcal{P}^{\mathrm{TBNN}'}\}\frac{\mathbf{I}}{3} + \{-\boldsymbol{A}^2\}\frac{\mathbf{I}}{3},$$

where $\mathbf{I}$ is the identity matrix and $\{\cdot\}$ represents the trace of the matrix.

[1] W. T. Ashurst, A. R. Kerstein, R. M. Kerr, and C. H. Gibson, Alignment of vorticity and scalar gradient with strain rate in simulated Navier–Stokes turbulence, Phys. Fluids **30**, 2343 (1987).

[2] P. Vieillefosse, Local interaction between vorticity and shear in a perfect incompressible fluid, J. Phys. **43**, 837 (1982).

[3] B. J. Cantwell, Exact solution of a restricted Euler equation for the velocity gradient tensor, Phys. Fluids A **4**, 782 (1992).

[4] J. Martín, A. Ooi, M. S. Chong, and J. Soria, Dynamics of the velocity gradient tensor invariants in isotropic turbulence, Phys. Fluids **10**, 2336 (1998).

[5] B. Lüthi, A. Tsinober, and W. Kinzelbach, Lagrangian measurement of vorticity dynamics in turbulent flow, J. Fluid Mech. **528**, 87 (2005).

[6] C. Kalelkar, Statistics of pressure fluctuations in decaying isotropic turbulence, Phys. Rev. E **73**, 046301 (2006).

[7] W. T. Ashurst, J. Y. Chen, and M. M. Rogers, Pressure gradient alignment with strain rate and scalar gradient in simulated Navier–Stokes turbulence, Phys. Fluids **30**, 3293 (1987).

[8] S. S. Girimaji and S. B. Pope, A diffusion model for velocity gradients in turbulence, Phys. Fluids A **2**, 242 (1990).

[9] K. Ohkitani, Eigenvalue problems in three-dimensional Euler flows, Phys. Fluids A **5**, 2570 (1993).

[10] A. Pumir, A numerical study of the mixing of a passive scalar in three dimensions in the presence of a mean gradient, Phys. Fluids **6**, 2118 (1994).

[11] S. S. Girimaji and C. G. Speziale, A modified restricted Euler equation for turbulent flows with mean velocity gradients, Phys. Fluids **7**, 1438 (1995).

[12] P. O'Neill and J. Soria, The relationship between the topological structures in turbulent flow and the distribution of a passive scalar with an imposed mean gradient, Fluid Dyn. Res. **36**, 107 (2005).

[13] L. Chevillard and C. Meneveau, Lagrangian Dynamics and Statistical Geometric Structure of Turbulence, Phys. Rev. Lett. **97**, 174501 (2006).

[14] C. B. da Silva and J. C. F. Pereira, Invariants of the velocity-gradient, rate-of-strain, and rate-of-rotation tensors across the turbulent/nonturbulent interface in jets, Phys. Fluids **20**, 55101 (2008).

[15] L. Chevillard and C. Meneveau, Lagrangian time correlations of vorticity alignments in isotropic turbulence: Observations and model predictions, Phys. Fluids **23**, 101704 (2011).

[16] J. Soria, R. Sondergaard, B. J. Cantwell, M. S. Chong, and A. E. Perry, A study of the fine-scale motions of incompressible time-developing mixing layers, Phys. Fluids **6**, 871 (1994).

[17] S. Pirozzoli and F. Grasso, Direct numerical simulations of isotropic compressible turbulence: Influence of compressibility on dynamics and structures, Phys. Fluids **16**, 4386 (2004).

[18] S. Suman and S. S. Girimaji, Homogenized Euler equation: A model for compressible velocity gradient dynamics, J. Fluid Mech. **620**, 177 (2009).

[19] S. Suman and S. S. Girimaji, Dynamical model for velocity-gradient evolution in compressible turbulence, J. Fluid Mech. **683**, 289 (2011).

[20] S. Suman and S. S. Girimaji, Velocity-gradient dynamics in compressible turbulence: Influence of Mach number and dilatation rate, J. Turbul. **13**, N8 (2012).

[21] L. Wang and X. Y. Lu, Flow topology in compressible turbulent boundary layer, J. Fluid Mech. **703**, 255 (2012).

[22] N. S. Vaghefi and C. K. Madnia, Local flow topology and velocity gradient invariants in compressible turbulent mixing layer, J. Fluid Mech. **774**, 67 (2015).

[23] M. Danish, S. S. Sinha, and B. Srinivasan, Influence of compressibility on the Lagrangian statistics of vorticity–strain-rate interactions, Phys. Rev. E **94**, 013101 (2016).

[24] N. Parashar, S. S. Sinha, M. Danish, and B. Srinivasan, Lagrangian investigations of vorticity dynamics in compressible turbulence, Phys. Fluids **29**, 105110 (2017).

[25] N. Parashar, S. S. Sinha, and B. Srinivasan, Lagrangian investigations of velocity gradients in compressible turbulence: Lifetime of flow-field topologies, J. Fluid Mech. **872**, 492 (2019).

[26] N. Parashar, S. S. Sinha, and B. Srinivasan, Lagrangian evaluations of viscous models for velocity gradient dynamics in non-stationary turbulence, Int. J. Heat Fluid Flow **78**, 108429 (2019).

[27] M. Chertkov, A. Pumir, and B. I. Shraiman, Lagrangian tetrad dynamics and the phenomenology of turbulence, Phys. Fluids **11**, 2394 (1999).

[28] E. Jeong and S. S. Girimaji, Velocity-gradient dynamics in turbulence: Effect of viscosity and forcing, Theor. Comput. Fluid Dyn. **16**, 421 (2003).

[29] L. Chevillard, C. Meneveau, L. Biferale, and F. Toschi, Modeling the pressure Hessian and viscous Laplacian in turbulence: Comparisons with direct numerical simulation and implications on velocity gradient dynamics, Phys. Fluids **20**, 101504 (2008).

[30] C. Meneveau, Lagrangian dynamics and models of the velocity gradient tensor in turbulent flows, Annu. Rev. Fluid Mech. **43**, 219 (2011).

[31] S. B. Pope, PDF methods for turbulent reactive flows, Prog. Energy Combust. Sci. **11**, 119 (1985).

[32] K. Duraisamy and P. Durbin, Transition modeling using data driven approaches, in *Proceedings of the CTR Summer Program* (Center for Turbulence Research, Stanford, CA, 2014), pp. 427–434.

[33] K. Duraisamy, Z. J. Zhang, and A. P. Singh, New approaches in turbulence and transition modeling using data-driven techniques, in *Proceedings of the 53rd AIAA Aerospace Sciences Meeting* (AIAA, Reston, VA, 2015).

[34] B. D. Tracey, K. Duraisamy, and J. J. Alonso, A machine learning strategy to assist turbulence model development, in *Proceedings of the 53rd AIAA Aerospace Sciences Meeting* (AIAA, Reston, VA, 2015).

[35] B. Tracey, K. Duraisamy, and J. Alonso, Application of supervised learning to quantify uncertainties in turbulence and combustion modeling, in *Proceedings of the 51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition* (AIAA, Reston, VA, 2013).

[36] E. J. Parish and K. Duraisamy, A paradigm for data-driven predictive modeling using field inversion and machine learning, J. Comput. Phys. **305**, 758 (2016).

[37] Z. J. Zhang and K. Duraisamy, Machine learning methods for data-driven turbulence modeling, in *Proceedings of the 2nd AIAA Computational Fluid Dynamics Conference* (AIAA, Reston, VA, 2015).

[38] J. Weatheritt, R. Pichler, R. D. Sandberg, G. Laskowski, and V. Michelassi, Machine learning for turbulence model development using a high-fidelity HPT cascade simulation, Turbo Expo: Power Land, Sea, Air **50794**, V02BT41A015 (2017).

[39] K. Duraisamy, G. Iaccarino, and H. Xiao, Turbulence modeling in the age of data, Annu. Rev. Fluid Mech. **51**, 357 (2019).

[40] J. Ling, A. Kurzawski, and J. Templeton, Reynolds averaged turbulence modelling using deep neural networks with embedded invariance, J. Fluid Mech. **807**, 155 (2016).

[41] R. Fang, D. Sondak, P. Protopapas, and S. Succi, Deep learning for turbulent channel flow, arXiv:1812.02241 (2018).

[42] C. Sotgiu, B. Weigand, and K. Semmler, A turbulent heat flux prediction framework based on tensor representation theory and machine learning, Int. Commun. Heat Mass **95**, 74 (2018).

[43] N. Geneva and N. Zabaras, Quantifying model form uncertainty in Reynolds-averaged turbulence models with Bayesian deep neural networks, J. Comput. Phys. **383**, 125 (2019).

[44] Y. Li, E. Perlman, M. Wan, Y. Yang, C. Meneveau, R. Burns, S. Chen, A. Szalay, and G. Eyink, A public turbulence database cluster and applications to study Lagrangian evolution of velocity increments in turbulence, J. Turbul. **9**, N31 (2008).

[45] E. Perlman, R. Burns, Y. Li, and C. Meneveau, Data exploration of turbulence simulations using a database cluster, in *Proceedings of the 2007 ACM/IEEE Conference on Supercomputing (SC '07)* (IEEE, Piscataway, NJ, 2007), pp. 1–11.
[46] http://turbulence.pha.jhu.edu
[47] https://torroja.dmt.upm.es/turbdata/Isotropic
[48] J. I. Cardesa, A. Vela-Martín, and J. Jiménez, The turbulent cascade in five dimensions, Science **357**, 782 (2017).
[49] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, Cambridge, MA, 2016).
[50] S. B. Pope, A more general effective-viscosity hypothesis, J. Fluid Mech. **72**, 331 (1975).
[51] A. J. M. Spencer and R. S. Rivlin, The theory of matrix polynomials and its application to the mechanics of isotropic continua, Arch. Ration. Mech. Anal. **2**, 309 (1958).
[52] F. Chollet, Keras, https://github.com/fchollet/keras, 2015.
[53] X. Glorot and Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, Proceedings of the 13th International Conference on Artificial Intelligence and Statistics, Proc. Mach. Learn. Res. **9**, 249 (2010).
[54] T. Tieleman and G. Hinton, Lecture 6.5–RMSprop: Divide the gradient by a running average of its recent magnitude, COURSERA: Neural Networks for Machine Learning (2012).
[55] N. Parashar, S. S. Sinha, B. Srinivasan, and A. Manish, GPU-accelerated direct numerical simulations of decaying compressible turbulence employing a GKM-based solver, Int. J. Numer. Meth. Fluids **83**, 737 (2017).
[56] M. Danish, S. Suman, and B. Srinivasan, A direct numerical simulation-based investigation and modeling of pressure Hessian effects on compressible velocity gradient dynamics, Phys. Fluids **26**, 126103 (2014).