

Citation:

Hasan, MD Badrul, et al. "Sub-grid Scale Modeling of Meso-scale Hurricane Boundary Layer Flows using Machine Learning," AIAA SCITECH 2023 Forum (19 January, 2023): 2023-2487.

<https://doi.org/10.2514/6.2023-2487>.

DOI:

<https://doi.org/10.2514/6.2023-2487>

Access to this work was provided by the University of Maryland, Baltimore County (UMBC) ScholarWorks@UMBC digital repository on the Maryland Shared Open Access (MD-SOAR) platform.

Please provide feedback

Please support the ScholarWorks@UMBC repository by emailing scholarworks-group@umbc.edu and telling us what having access to this work means to you and why it's important to you. Thank you.

Sub-grid Scale Modeling of Meso-scale Hurricane Boundary Layer Flows using Machine Learning

Md Badrul Hasan* and Meilin Yu†

University of Maryland, Baltimore County, Baltimore, MD, 21250, USA

Heng Xiao‡

University of Stuttgart, Germany

Modeling turbulent flows in natural hazard like hurricanes at the meso-scale is still challenging. Large-eddy simulations (LES) can be used to solve meso-scale atmospheric flows, either in an implicit approach or augmented with an explicit sub-grid scale (SGS) model. Most SGS models assume that the turbulent kinetic energy is solely transported from large scales to smaller ones. Therefore, the SGS model is purely dissipative. However, it has been realized that quantifying and modeling the opposite energy transport from small scales to larger ones (i.e., energy backscatter) in conditions like hurricane modelling is important to incorporate correct dynamics into the complex system. In the meso-scale regime there is very limited work on characterizing effects of inter-scale energy transfer. Data-driven machine/deep learning methodology is examined in this work as possible alternatives to traditional SGS modeling approaches. Basic feed-forward neural network is used to model the meso-scale energy transfer, including both forward energy cascade and energy backscatter, using filtered input data from fine-scale LES simulation of a hurricane-like vortex.

I. Introduction

Large-eddy simulation (LES) has been widely adopted in the simulation of complex fluid flows, such as atmospheric boundary layer (ABL) flows over an offshore wind farm [1]. Since the computational cost of LES can be very high to resolve turbulent motions, especially for very high Reynolds number flows, such as geophysical flows [2], subgrid-scale (SGS) modeling at meso-scale (e.g., at the kilometer (km) scale for ABL flows) is usually unavoidable to save computational cost. We note that the state-of-the-art SGS models in LES focus on turbulent behaviors at the inertial range (e.g., at the meter (m) scale for ABL flows), where forward kinetic energy cascade prevails. In the geophysical flow community, such SGS models are named “micro-scale” models, and that is why LES of very high Reynolds number geophysical flow is very expensive. Meso-scale modeling can significantly reduce the computational cost; however, the effect of energy backscatter from small-scale turbulence to large-scale flow structures may not be neglected. In a recent work [3], organized kinetic energy backscatter has been observed in the hurricane boundary layer from radar measurements. This indicates that SGS modeling for meso-scale flows may need to include non-equilibrium flow effects to remedy missing dynamics. In this work, we use machine/deep learning neural networks to develop backscatter-admitted meso-scale SGS models for turbulent hurricane boundary layer flows. The ultimate goal is to use this model in the meso-scale simulation of hurricane boundary layer flows [4].

For completeness, we first briefly review several recent works on turbulence modeling with machine/deep learning; see a more comprehensive review in [5]. Using high-fidelity simulations and experimental data, Ling et al. [6] learned a model for the Reynolds stress anisotropy tensor using artificial neural networks (ANN). Srinivasan et al. [7] evaluated the ability of neural networks to predict temporally changing turbulent flows and concluded that long-short-term-memory (LSTM) networks outperform traditional machine learning (ML) techniques. Milano and Koumoutsakos [8] compared the results from the neural network methods with proper orthogonal decomposition (POD) for turbulent flow near the wall, where neural networks give better prediction and reconstruction. Recently, ANNs have been used to classify models at the sub-grid scale [9]. With machine learning-based techniques, data-driven approaches, such as ANNs, are set to become the foundation of the next significant advance in closure modeling for nonlinear conservation laws.

*Graduate Student, Department of Mechanical Engineering and AIAA Student Member. Email: mdbadrh1@umbc.edu

†Associate Professor, Department of Mechanical Engineering and AIAA Senior Member. Email: mlyu@umbc.edu

‡Professor, Cluster of Excellence SimTech, Stuttgart Center for Simulation Science. Email: heng.xiao@simtech.uni-stuttgart.de

Most of the works on SGS modelling are limited on fine-scale LES. For example, Lapeyre et al. [10] and Beck et al. [11] studied about how ML could be used to create SGS models for LES. Some research has examined the SGS model's accuracy in calculating the SGS term as a filtered version of resolved flow [12–16]. The data-driven SGS models showed better performance than the physics-based models when it comes to model inter-scale energy transfers. Most of these studies have also stated that the LES models become unstable when the data-driven SGS model is paired with a less dissipative numerical solver which can lead to flows that are not realistic [11, 14, 17, 18]. Deep convolutional neural networks (CNNs) can accurately predict SGS forcing terms and inter-scale transfers in *a priori* tests if the model is trained with a sufficient amount of data [19]; instead, when the training dataset is small, CNNs are less accurate at collecting energy backscatter (anti-diffusion), which causes numerical instabilities. Transfer learning, which includes retraining the CNN with a modest amount of new flow data, permits accurate and stable LES-CNN for meso-scale flows [19].

In this work, we developed a meso-scale SGS model using high-fidelity simulation data of a hurricane-like vortex and machine/deep learning techniques, and carry out *a priori* tests of the model performance. Different from most of the recent works are focused on the SGS modeling for micro-scale LES and stabilizing the simulations, we give special attention to the meso-scale energy forward cascade and backscatter in this work. In section II, the machine/deep learning approach which is used in this work is described in detail. Section III explains details about the dataset used for model training. Section IV presents the results obtained from machine learning-based models training and testing and section V gives the concluding remarks of this work and discusses the future work.

II. Machine Learning Methods

The term “machine learning” refers to the process by which computers learn from the data and algorithms provided to them in order to complete a task without being explicitly programmed. The human brain is used as a model for the intricate algorithmic structure that is used in deep learning. Neural networks consisting of several layers are used in deep learning, a sub-field of machine learning. The nodes of a neural network are linked to each other. There is an activation function for each node that turns the inputs x into the output $f(x)$. It is common for the outputs from one layer to serve as inputs to the next in a deep neural network. Deep neural networks can capture complicated, hierarchical connections between features by utilizing multiple layers of transformations. Due to their inability to be interpreted in a physical sense, the layer between input and output are known as hidden layers. Back propagation with gradient descent was used in the training of the neural networks. Calibration of a regression model is akin to training a neural network. In training, a portion of the data is used to fit the model, and the weights and biases of the nodes are adjusted repeatedly until the mean squared error between the predicted and actual output values is as little as possible. Number of hidden layers, nodes per hidden layer, and learning rate in gradient descent training are the major hyper-parameters for neural network topology. These three parameters can all have a big impact on model performance. However, larger networks (with more hidden layers and nodes per layer) are prone to over-fitting data. A simple neural network architecture is shown in Fig. 1.

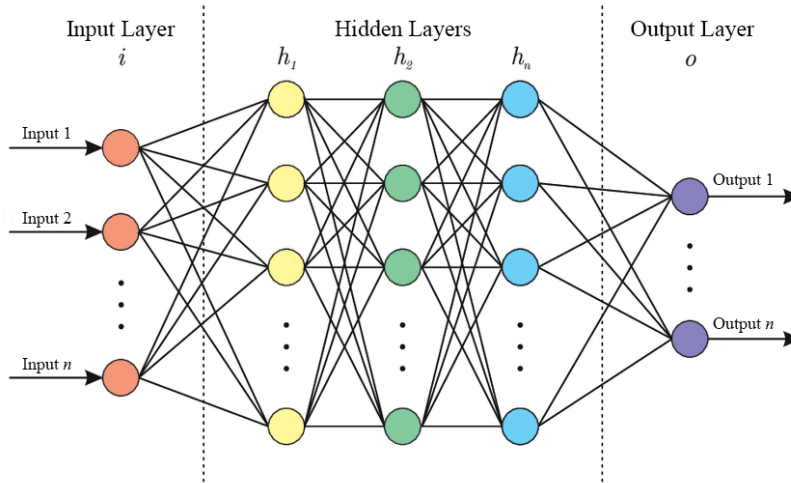


Fig. 1 An illustration of a basic feedforward neural network.

For this application, we have gone with a dense, fully connected, feed-forward DNN:

$$\begin{aligned}
h_1 &= \sigma_1 (w_1 x + b_1) \\
h_2 &= \sigma_2 (w_2 h_1 + b_2) \\
&\vdots \\
h_n &= \sigma_n (w_n h_{n-1} + b_n) \\
f(x) &= (w_{out} h_n + b_{out})
\end{aligned}$$

Here, x denotes the input vector; $f(x)$ is the predicted output vector; w_1, w_2, \dots, w_n , and w_{out} are matrices of trainable weights; b_1, b_2, \dots, b_n , and b_{out} are “bias vectors”; $\sigma_1, \sigma_2, \dots, \sigma_n$ are the nonlinear activation functions; and h_1, h_2, \dots, h_n are the “hidden” vectors whose scalar components are called “neurons.” Note that activation functions are only applied on the hidden vectors.

For NN training, gradient-based algorithms with backpropagation are used to minimize the cost function. Specifically, a nonlinear least squares problem is set up, and the Levenberg-Marquardt optimization with Bayesian regularization [20] is used to estimate the weights and bias in the feedforward NN.

III. Data Set Preparation

Neural networks will be trained and validated on a database of idealized hurricane simulations which are examined at four distinct sea surface temperatures of 26°C, 27°C, 28°C, and 29°C. Warm sea surface temperature, which is more than 26°C, is a contributing factor in the genesis and development of hurricanes. In addition, the power and structure of hurricanes are extremely susceptible to variations in the temperature of the sea surface, which can result in varying degrees of devastation for coastal regions. The data used here are from the high-fidelity numerical simulation by Ren et al. [2], and the finest flow field resolution is at about 62 m. The numerical simulations used the LES mode of the Weather Research and Forecasting Model (WRF) [21], with which small-scale turbulence patterns at the order of meters can be captured. More simulation details can be found in [2, 22].

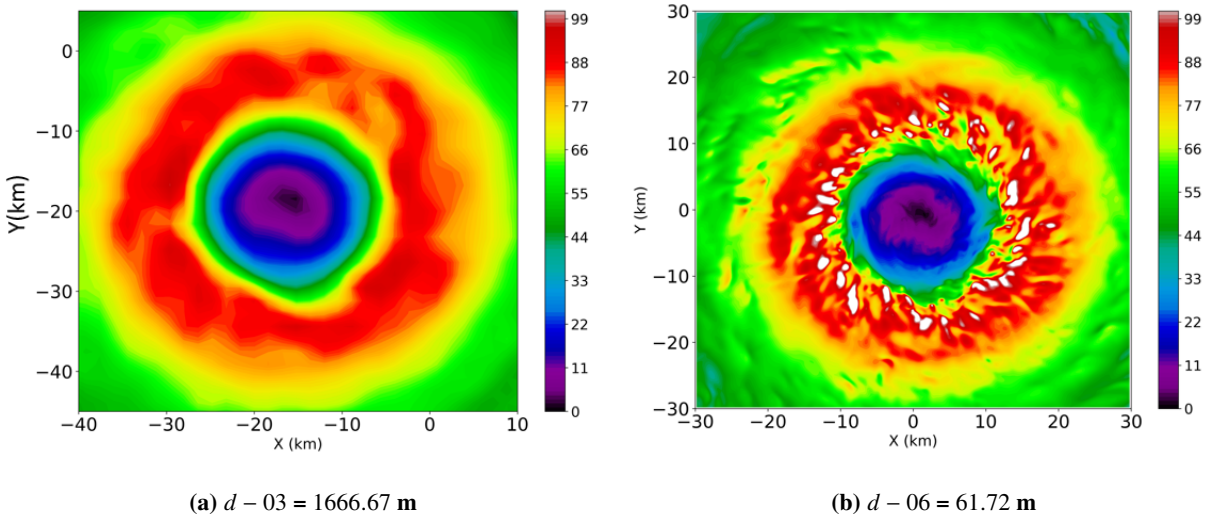


Fig. 2 Instantaneous horizontal windspeed (m/s) plots.

A typical horizontal windspeed structure of the hurricane at 29°C sea surface temperature using two different numerical resolutions is shown in Fig. 2. The grid spacing in Fig. 2(a) is 1.6 km, and that in Fig. 2(b) is 61.72 m. From the two panels, it is clear that more turbulent eddies are observed from the simulation with a finer resolution, especially around 10-20 km away from the hurricane center. We will use the fine-scale simulation data to calculate eddy viscosity and Smagorinsky constant (C_s), and use them to train the machine/deep learning model. The data preparation step is described briefly in Subsection III.B.

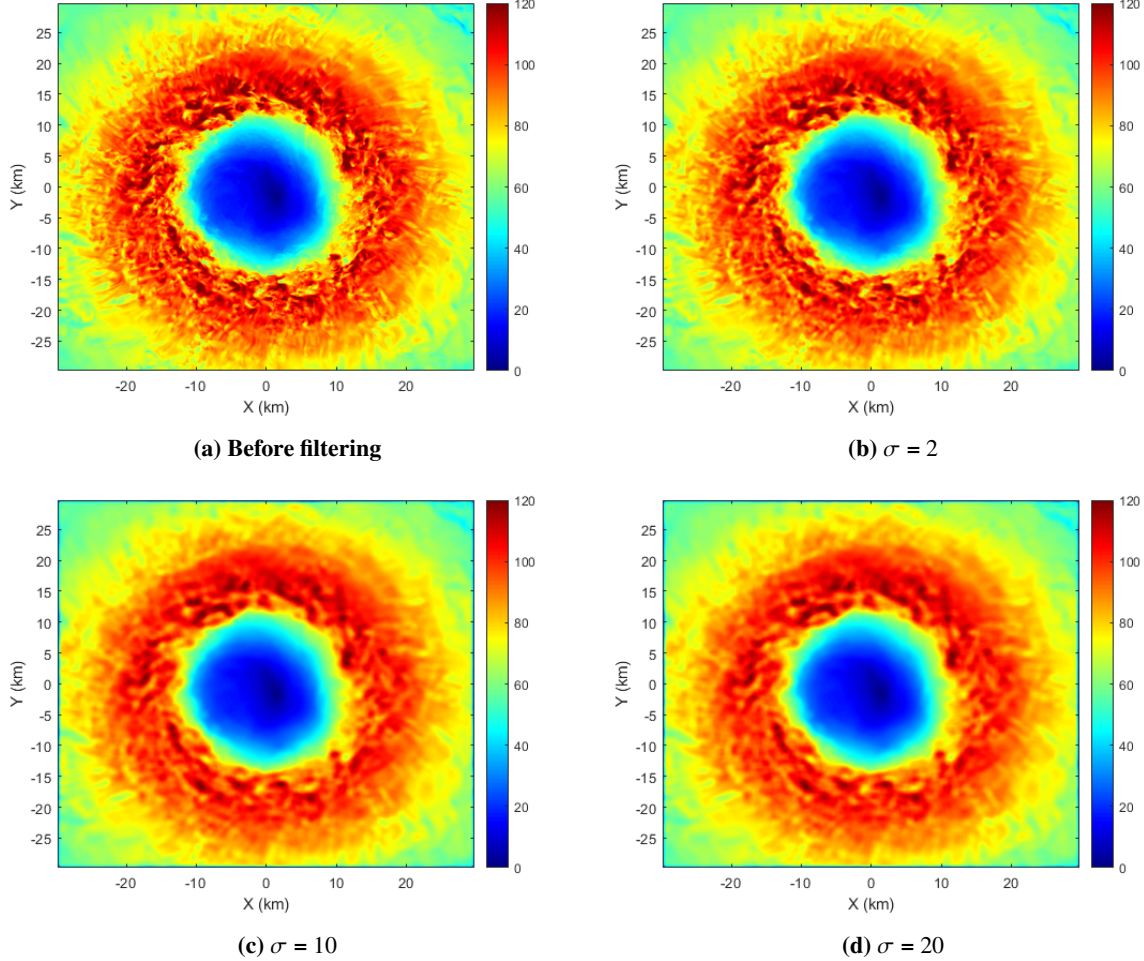


Fig. 3 Horizontal wind speed field and fields after using 1 km gaussian filter at 303 m vertical level.

A. Filtering LES data

From the simulation data [2], we can get the three-dimensional field of velocity components (u, v, w). The velocity components (u, v, w) are filtered using a Gaussian filter. The weights of the filter members depend on their location relative to the center one. Specifically, the center filter member receives the maximum weight, and the weight decreases when a filter member departs from the center. A straightforward formulation of the Gaussian filter is shown below. Let n denote the number of points in a given filter direction, and σ denotes the standard deviation. The term “standard deviation” refers to a measurement of the data’s dispersion from the mean. A low standard deviation implies that the data are grouped around the mean, whereas a large standard deviation shows that the data are more dispersed. The weight at the $\pm i$ position relative to the filter center is calculated as follows:

$$\hat{w}_{\pm i}(\Delta, n) = \exp\left(-\frac{(i\Delta/n)^2}{2\sigma^2}\right), \quad w_{\pm i}(\Delta, n) = \frac{\hat{w}_{\pm i}(\Delta, n)}{\sum \hat{w}_{\pm i}(\Delta, n)},$$

where Δ is the filter width. The Gaussian filter requires an odd number of points to create the Gaussian kernel. We selected the number of points based on the grid spacing used in meso-scale simulations. Usually, 1 km or 2 km grid spacing is used for the meso-scale simulations. From the data set, the finest LES simulation has a grid spacing of 62 m only. That is why we selected 17 points in x and y coordinates to create the Gaussian kernel for 1 km filtering. Similarly, we used 35 points for 2 km filtering using Gaussian distribution.

Fig. 3(a) shows the horizontal wind speed field at the 19th hour of the 29°C sea surface temperature simulation. We have applied the 1 km filtering which uses a Gaussian kernel that has 17 points. The gaussian filter is applied in the x and y components of u , v , and w as there is little impact from the z-direction on these components. We can use different values for σ on the Gaussian kernel calculation. Then the filtered u , v , and w are used to calculate filtered horizontal wind speed fields. Here we are showing the effects of different σ on 1 km filtering in Fig. 3(b)-(d). For $\sigma = 2$, we can see the horizontal wind speed field has a less intense structure. But wind speed fields become more smooth when σ becomes larger, such as 10 or 20. With larger values of σ , Gaussian kernel weights go closer to a constant value with less variability like a straight line. We can also see there is less difference between when σ is equal to 10 and 20 values, although $\sigma = 20$ is relatively more smoother than $\sigma = 10$. A similar kind of structure was observed during the 2 km filtering as well.

Kinetic energy spectra can be calculated from the simulation data and the filtered ones. Fig. 4 shows the kinetic energy spectra of horizontal windspeed in the x-direction at $y = 0$ km and on $z = 303$ m vertical level. We also did the same plot for $x = 0$ km in the y-direction which showed a similar structure and tendency. From Fig. 4 it can be observed that all kinetic energy spectra follow the theoretical slope of $-5/3$ in the resolved inertial range of turbulent flows. After filtering, resolved small-scale features are reduced to 1 and 2 km. For different values of σ , we can observe that there exist differences between the filtering. More kinetic energies have been dissipated when σ becomes larger. Similar to Fig. 3(c)-(d), we can observe small differences between $\sigma = 10$ and 20 in Fig. 4 as well. In this study, we used 1 km filtering with $\sigma = 2$ to calculate flow features, such as eddy viscosity, in the meso-scale region to provide training data for feedforward NN. We mention that data with different filtering widths can be all used to train the machine learning SGS model, which may be generally applicable to numerical simulations with different flow resolutions. These results will be reported in our future publication.

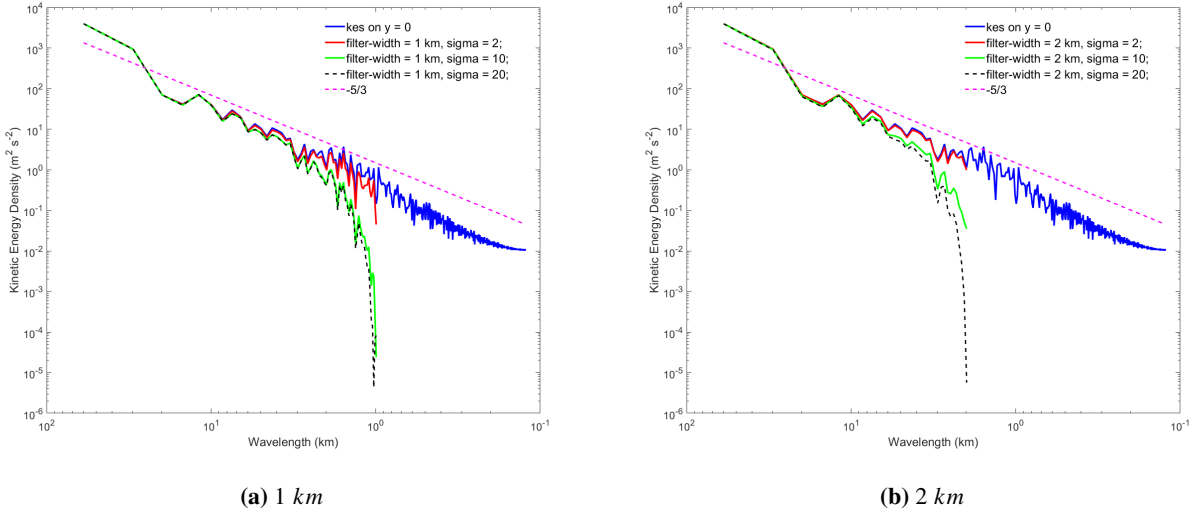


Fig. 4 Kinetic energy spectra after using (a) 1 km and (b) 2 km gaussian filter at 303 m vertical level.

B. Smagorinsky Model Coefficient Evaluation

We can calculate the eddy viscosity from the concept of the production of turbulent kinetic energy. Specifically, we can quantify production based on the stress and strain tensor through its definition (φ^{def} ; Eqn. 1). We can also quantify production based on the strain tensor and eddy viscosity values (Eqn. 2). Using both of these formulations we can get the values of eddy viscosity as we can directly calculate the stress and strain tensors from the filtered data.

$$\varphi^{\text{def}} = -(\overline{u_i u_j} - \bar{u}_i \bar{u}_j) \frac{\partial \bar{u}_i}{\partial x_j} = -(\overline{u_i u_j} - \bar{u}_i \bar{u}_j) \bar{S}_{ij}, \quad (1)$$

$$\varphi = \nu_T \left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i} \right) \frac{\partial \bar{u}_i}{\partial x_j} = 2\nu_T \bar{S}_{ij} \bar{S}_{ij}, \quad (2)$$

where \bar{S}_{ij} is the strain rate tensor defined as $\bar{S}_{ij} = (\partial \bar{u}_i / \partial x_j + \partial \bar{u}_j / \partial x_i) / 2$. Eddy viscosity fields from this calculation are shown in Fig. 5(a). We observed that the eddy viscosity has both positive and negative values in the fields. Note that a negative diffusion coefficient indicates anti-diffusion, and is usually avoided in SGS modeling as it violates the Boussinesq assumption. However, we assume that the anti-diffusion effect can be used to model energy backscatter. For simplicity, we directly use the signed eddy viscosity concept in this study. Specifically, the positive eddy viscosity indicates energy cascade, and the negative one indicates energy backscatter. As a result, the Smagorinsky constant C_s is also treated as a signed one in our work.

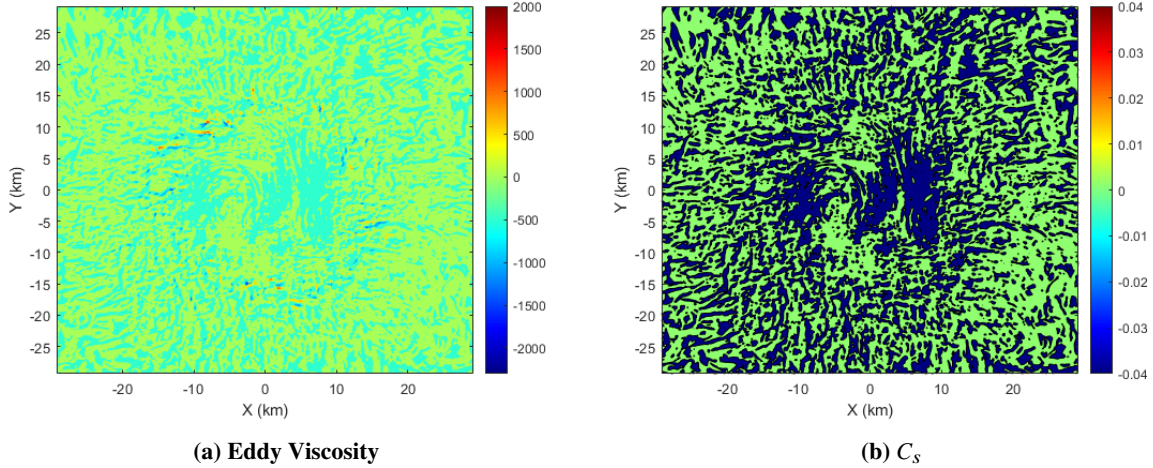


Fig. 5 Eddy viscosity and C_s values on horizontal field at 303 m vertical level

The eddy viscosity model from the Smagorinsky [23] scheme can be written as

$$\nu_t = C_s \Delta^2 |\bar{S}|, \quad (3)$$

where Δ is the filter size and $|\bar{S}|$ is defined as $\sqrt{2\bar{S}_{ij}\bar{S}_{ij}}$. Fig. 5(b) shows the calculated values of C_s in the selected domain. There is large variability on this x-y field which can hardly be distinguished from the color bar. To show more details from the actual fields we presented several line plots at positions of $y = -15, -10, 0, 10, \text{ and } 15 \text{ km}$ positions relative to the hurricane center in Fig. 6. From Fig. 6, large variability of the values of eddy viscosity and Smagorinsky constant (C_s) can be seen. This lack of consistency or fixed pattern on these values makes SGS modeling a challenging task. Note that the y-positions are chosen based on Fig. 3, in which flow structures of the hurricane eye and eye-wall exhibit large spatial variations. These positions will also be used for evaluate the accuracy of the trained machine learning-based SGS model in *a priori* tests.

We mention that in the original Smagorinsky model, the model constant is considered as a positive one and as a constant value over the whole field. Based on the WRF documentation C_s is a constant of $0.25^2 = 0.0625$. If we use this constant with the filter size as the characteristic length and the strain tensor, the eddy viscosity can be directly calculated from Eq. 3. In our previous work [4], this approach was used to evaluate the eddy viscosity distribution in the meso-scale simulation of a hurricane-like vortex. Therein, the averaged eddy viscosity is at the order of about $O(10^3) \text{ m}^2/\text{s}$, similar to the peak values observed in this study.

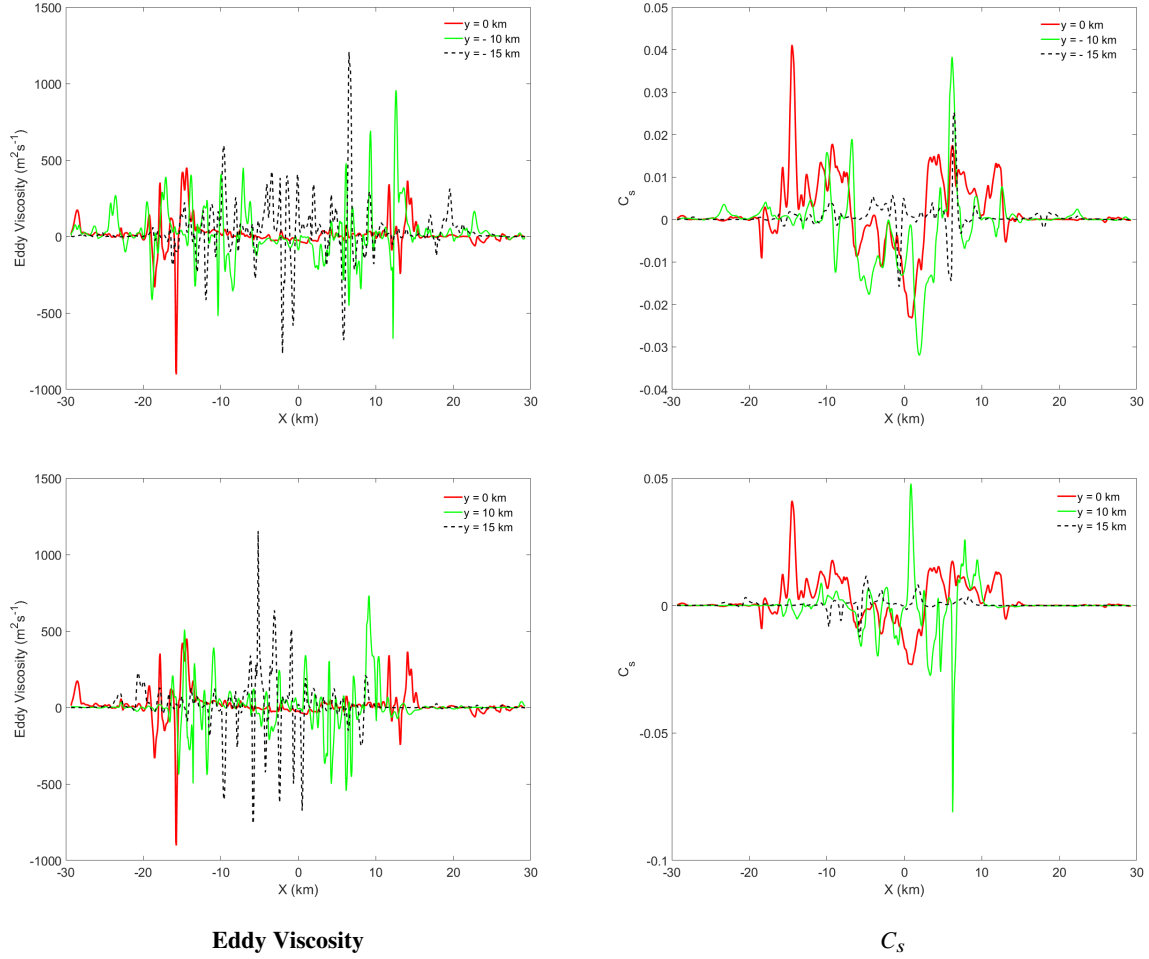


Fig. 6 Eddy viscosity and C_s values at different positions of y on x -directions at 303 m vertical level

IV. Machine Learning SGS Models with Feed-forward Neural Networks

Machine/deep learning algorithms are used to develop an SGS model to predict the value and sign of Smagorinsky constant C_s in hurricane boundary layers for meso-scale simulations. For the machine/deep learning algorithm, we have prepared the input data set including the resolved kinetic energy which is calculated from the filtered u , v , and w components of wind speed. We have also included the gradients of these filtered fields in three dimensions (x , y , z). The C_s values presented in Fig. 5 are used as the target data-set field to train and test the machine learning-based SGS model. As a first step, we select a height level (303 m) from the three-dimensional (3D) flow field. The dimension of data is (950, 950) in x and y directions. Then we use 70% randomly selected sample points as the training data set and the remaining 30% as the testing data set. For simplicity, data splitting is only based on their x coordinates. As a result, the size of training and testing data set is (665, 950) and (285, 950), respectively. All the input data sets are normalized using the absolute maximum values before being used in model training or testing.

Five different feed-forward neural networks are used where the model contains 2, 3, or 4 hidden layers with each containing 16 or 32 neurons on each layer. We have used Matlab's Deep Learning Toolbox for the training of neural networks. The Bayesian regularization [24, 25] is used as the train function which uses Levenberg-Marquardt optimization to change the weight and bias values. It finds the best combination of squared errors and weights to make a network that generalizes well. This is done by minimizing a set of squared errors and weights. The performance parameter is defined as the mean squared error (MSE) between the prediction and test/training data set. Fig. 7 shows mean square values which are calculated during the training process of the neural networks. All neural networks were trained for 2000 epochs using 24 CPU cores. From Fig. 7 we can observe that the error converges about four orders after 400 epochs, and the absolute training errors of deeper NNs are smaller than those of shallower NNs.

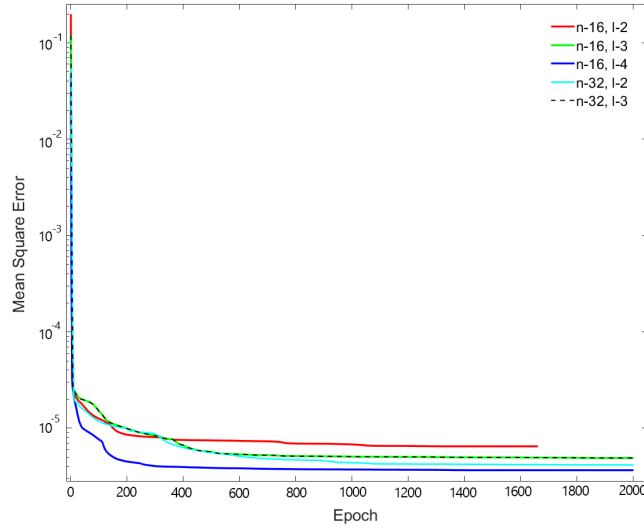


Fig. 7 Training error of different neural network setups.

In Fig. 8 the performance of different neural networks evaluated with the test data set are resented. The figure is plotted as C_s vs. the indices of sample points. Note that the test data set consists of 270,750 sample points, and we are only showing a small part (200 points) to visualize the differences here. The blue lines indicate the testing data which was separated before training neural networks. Red lines show the predictions from different setups of neural networks on the first 200 points of the testing data set.

We have also computed two types of error norms to get a better idea of the quality of the trained neural networks. The L^2 Norm measures the error between all the testing data and predictions from the neural networks. This value is averaged using the total number of sample points on the testing data set consisting of 270,750 sample points. The L^∞ Norm gives the maximum difference between the testing data and predictions, which occurs at one single point. From Table 1 we can see the neural network with 4 hidden layers and 16 neurons in each hidden layer performs best. We also see there is a large value of L^∞ Norm for the model with 3 hidden layers and 16 neurons in each hidden layer, and that with 2 hidden layers and 32 neurons in each hidden layer. As most of the machine learning based models cannot achieve 100% correct predictions, these discrepancies are expected in our study. However, for scientific computing, such discrepancies can be detrimental to the stability of the numerical schemes, and need to be avoided. More study is needed to find how to decrease the error on predictions, such as incorporating the L^∞ norm of the error in the NN training, and/or using more sophisticated NN models, such as the vector cloud neural network (VCNN) [26], and the graph kernel network (GKN) [27]. Results will be reported in our future publications.

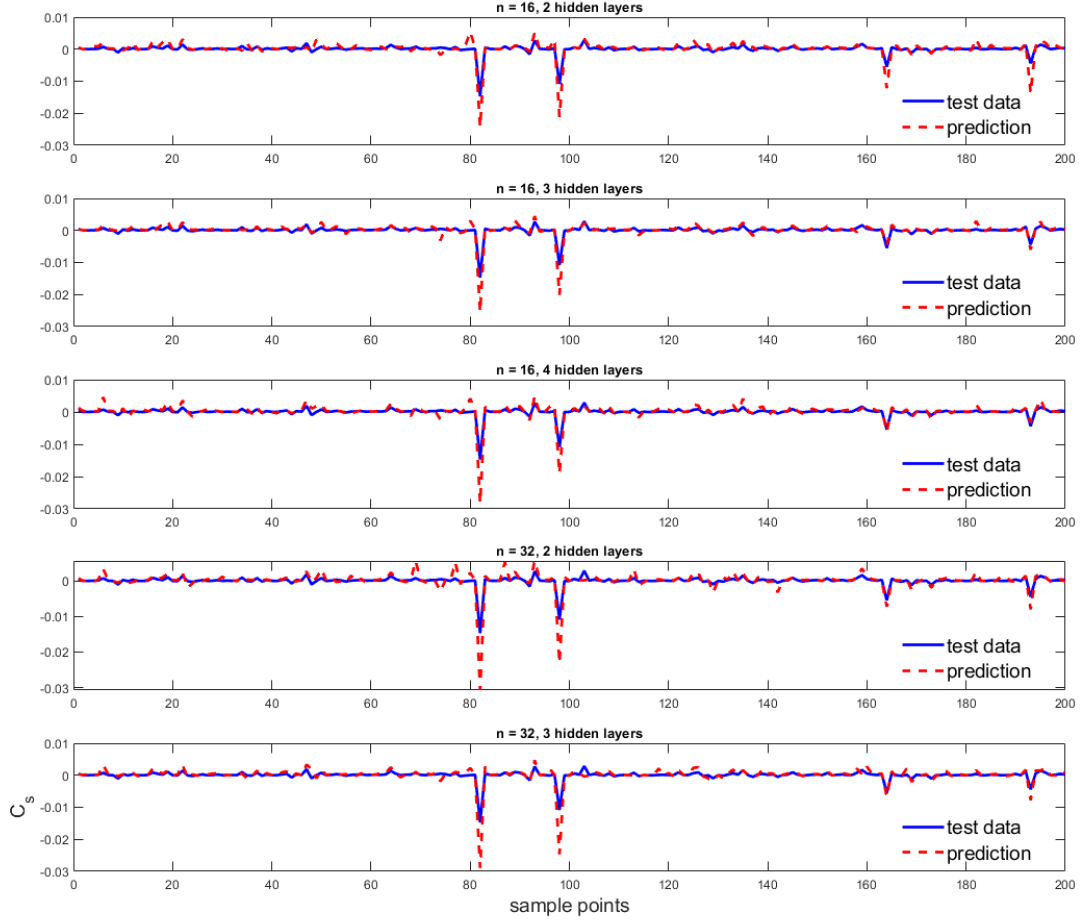


Fig. 8 Comparison between testing data and predictions from 5 different neural network setup on height level around 303 m

Table 1 Prediction errors of C_s in the data-driven SGS model with different neural network setups. Here “HL” stands for “Hidden Layer”.

Model Description	L^2 Norm	L^∞ Norm
2 HLs, 16 Neurons/HL	7.072×10^{-6}	0.3374
3 HLs, 16 Neurons/HL	2.838×10^{-5}	3.459
4 HLs, 16 Neurons/HL	6.194×10^{-6}	0.2123
2 HLs, 32 Neurons/HL	3.648×10^{-5}	5.920
3 HLs, 32 Neurons/HL	16.431×10^{-6}	0.4847

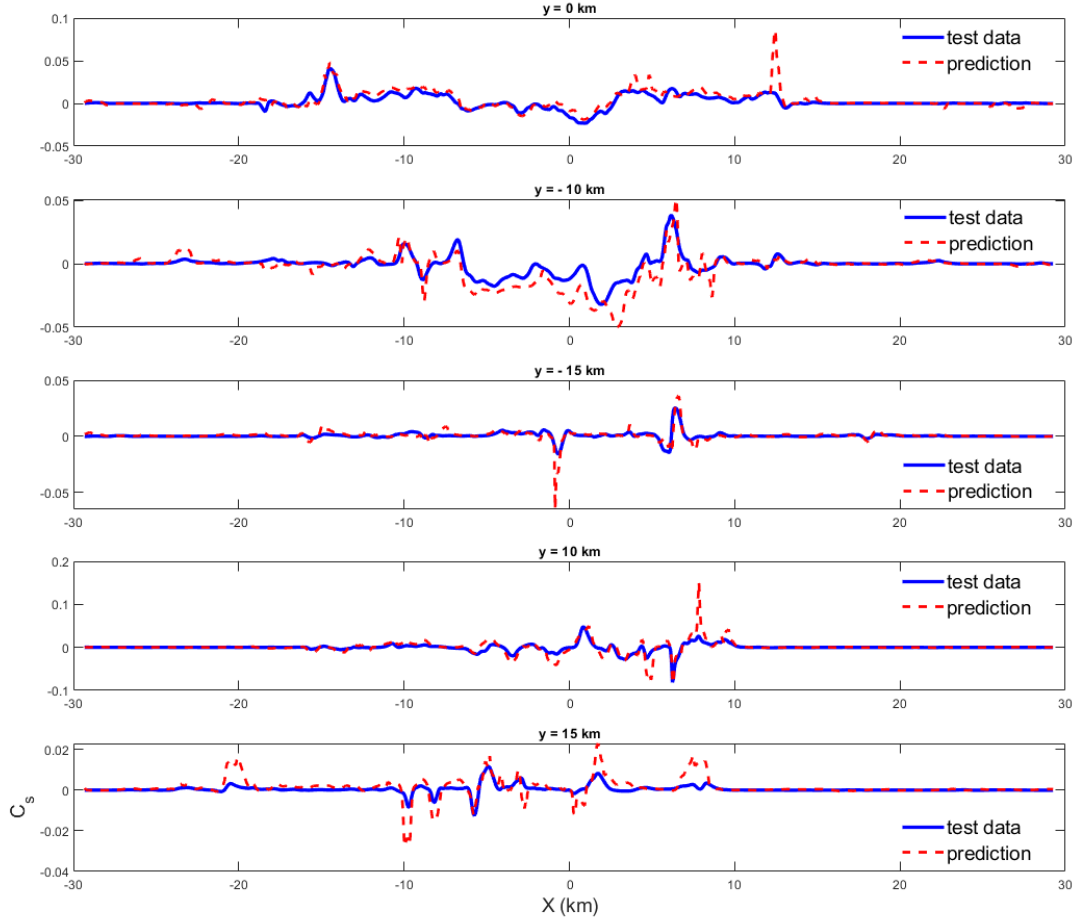


Fig. 9 Comparison between calculated C_s and predictions from neural network having 4 hidden layer each containing 16 neurons on different points in x-direction

Based on Fig. 8 and Table 1, we select the neural network with 4 hidden layers and each containing 16 neurons for testing on selected physical locations from Fig. 6. The comparison between the predicted values of C_s from neural network and calculated values of C_s are shown in Fig. 9. In this test five different locations were used, and each location has 950 sample points. The SGS model receives resolved kinetic energy and derivatives of u , v and w in three dimensions x , y and z as inputs, and gives the prediction of C_s on those sample points as the output. The prediction and test data can reasonably match with each other well, although there exist certain mismatch. In Table 2 the L^2 Norm and the L^∞ Norm of the errors are presented for different predictions. We observe that the L^∞ Norm of the error is small for every prediction from this small test set.

Table 2 Prediction errors of C_s in the data-driven SGS model at different locations of the hurricane vortex away from its center.

Test point on x-direction	L^2 Norm	L^∞ Norm
y = 0 km	2.097×10^{-4}	7.339×10^{-2}
y = - 10 km	2.333×10^{-4}	3.489×10^{-2}
y = - 15 km	1.056×10^{-4}	5.479×10^{-2}
y = 10 km	2.969×10^{-4}	1.249×10^{-1}
y = 15 km	1.201×10^{-4}	2.447×10^{-2}

V. Conclusions and Future Work

In our current work, structured kinetic energy backscatter (indicated by negative eddy viscosity) has been observed in a finely resolved (at about 60 m) hurricane boundary layer flow field filtered at meso-scale (1 km). A fully connected feed-forward NN is used to model the signed Smagorinsky model coefficient for the filtered flow fields. We find that although the variation pattern of the Smagorinsky model coefficient is complicated, the feed-forward NN can reasonably recognize the pattern. However, the maximum prediction error of the data-driven SGS model is not bounded. This can be detrimental to scientific computing if such models are used to provide SGS stresses. Therefore, more studies are needed to understand the error source, and reduce the model errors. As has been observed in [19], deep CNNs can improve the accuracy of SGS models if the model is trained with a sufficient amount of data. In the future work, we will test more advanced NNs, such as VCNN and GKN, to incorporate more physics into the data-driven SGS model to strengthen its accuracy and robustness.

Acknowledgments

The first author would like to acknowledge Dr. Stephen Guimond for the many discussions on hurricane boundary layer flow physics. The hardware used in the computational studies is part of the UMBC High Performance Computing Facility (HPCF). The facility is supported by the U.S. National Science Foundation through the MRI program (grant nos. CNS-0821258, CNS-1228778, and OAC-1726023) and the SCREMS program (grant no. DMS-0821311), with additional substantial support from the University of Maryland, Baltimore County (UMBC).

References

- [1] Yang, D., Meneveau, C., and Shen, L., “Large-eddy simulation of offshore wind farm,” *Physics of Fluids*, Vol. 26, No. 2, 2014, p. 025101. <https://doi.org/10.1063/1.4863096>.
- [2] Ren, H., Dudhia, J., and Li, H., “Large-eddy simulation of idealized hurricanes at different sea surface temperatures,” *Earth and Space Science Open Archive*, 2019, p. 10. <https://doi.org/10.1002/essoar.10501105.1>, URL <https://doi.org/10.1002/essoar.10501105.1>.
- [3] Sroka, S., and Guimond, S., “Organized kinetic energy backscatter in the hurricane boundary layer from radar measurements,” *Journal of Fluid Mechanics*, Vol. 924, No. A21, 2021.
- [4] Hasan, M. B., Guimond, S. R., Yu, M. L., Reddy, S., and Giraldo, F. X., “The Effects of Numerical Dissipation on Hurricane Rapid Intensification With Observational Heating,” *Journal of Advances in Modeling Earth Systems*, Vol. 14, No. 28, 2022.
- [5] Duraisamy, K., Iaccarino, G., and Xiao, H., “Turbulence Modeling in the Age of Data,” *Annual Review of Fluid Mechanics*, Vol. 51, No. 1, 2019, pp. 357–377. <https://doi.org/10.1146/annurev-fluid-010518-040547>, URL <https://www.annualreviews.org/doi/10.1146/annurev-fluid-010518-040547>.
- [6] Ling, J., Kurzawski, A., and Templeton, J., “Reynolds averaged turbulence modelling using deep neural networks with embedded invariance,” *Journal of Fluid Mechanics*, Vol. 807, 2016, p. 155–166. <https://doi.org/10.1017/jfm.2016.615>.
- [7] Srinivasan, P. A., Guastoni, L., Azizpour, H., Schlatter, P., and Vinuesa, R., “Predictions of turbulent shear flows using deep neural networks,” *Phys. Rev. Fluids*, Vol. 4, 2019, p. 054603. <https://doi.org/10.1103/PhysRevFluids.4.054603>, URL <https://link.aps.org/doi/10.1103/PhysRevFluids.4.054603>.

- [8] Milano, M., and Koumoutsakos, P., “Neural network modeling for near wall turbulent flow,” *Journal of Computational Physics*, Vol. 182, 2002, pp. 1–26.
- [9] Gamahara, M., and Hattori, Y., “Searching for turbulence models by artificial neural network,” *Physical Review Fluids*, Vol. 2, No. 5, 2017, 054604. <https://doi.org/10.1103/PhysRevFluids.2.054604>.
- [10] Lapeyre, C. J., Misdariis, A., Cazard, N., Veynante, D. P., and Poinso, T., “Training convolutional neural networks to estimate turbulent sub-grid scale reaction rates,” *Combustion and Flame*, 2019.
- [11] Beck, A., Flad, D., and Munz, C.-D., “Deep neural networks for data-driven LES closure models,” *Journal of Computational Physics*, Vol. 398, 2019, p. 108910. <https://doi.org/https://doi.org/10.1016/j.jcp.2019.108910>, URL <https://www.sciencedirect.com/science/article/pii/S0021999119306151>.
- [12] Maulik, R., San, O., Rasheed, A., and Vedula, P., “Subgrid modelling for two-dimensional turbulence using neural networks,” *Journal of Fluid Mechanics*, Vol. 858, 2019, p. 122–144. <https://doi.org/10.1017/jfm.2018.770>.
- [13] Pawar, S., San, O., and Rasheed, A., “A priori analysis on deep learning of subgrid-scale parameterizations for Kraichnan turbulence,” *Theor. Comput. Fluid Dyn.*, Vol. 34, 2020, p. 429–455.
- [14] Zhou, Z., He, G., Wang, S., and Jin, G., “Subgrid-scale model for large-eddy simulation of isotropic turbulent flows using an artificial neural network,” *Computers & Fluids*, Vol. 195, 2019, p. 104319. <https://doi.org/https://doi.org/10.1016/j.compfluid.2019.104319>.
- [15] Bolton, T., and Zanna, L., “Applications of Deep Learning to Ocean Data Inference and Subgrid Parameterization,” *Journal of Advances in Modeling Earth Systems*, Vol. 11, No. 1, 2019, pp. 376–399. <https://doi.org/https://doi.org/10.1029/2018MS001472>.
- [16] Zanna, L., and Bolton, T., “Data-Driven Equation Discovery of Ocean Mesoscale Closures,” *Geophysical Research Letters*, Vol. 47, No. 17, 2020, p. e2020GL088376. <https://doi.org/https://doi.org/10.1029/2020GL088376>, e2020GL088376 10.1029/2020GL088376.
- [17] Beck, A., and Kurz, M., “A perspective on machine learning methods in turbulence modeling,” *GAMM-Mitteilungen*, Vol. 44, No. 1, 2021, p. e202100002. <https://doi.org/https://doi.org/10.1002/gamm.202100002>.
- [18] Stoffer, R., van Leeuwen, C. M., Podareanu, D., Codreanu, V., Veerman, M. A., Janssens, M., Hartogensis, O. K., and van Heerwaarden, C. C., “Development of a large-eddy simulation subgrid model based on artificial neural networks: a case study of turbulent channel flow,” *Geoscientific Model Development*, Vol. 14, No. 6, 2021, pp. 3769–3788. <https://doi.org/10.5194/gmd-14-3769-2021>.
- [19] Guan, Y., Chattopadhyay, A., Subel, A., and Hassanzadeh, P., “Stable a posteriori LES of 2D turbulence using convolutional neural networks: Backscattering analysis and generalization to higher Re via transfer learning,” *Journal of Computational Physics*, Vol. 458, 2022, p. 111090. <https://doi.org/10.1016/j.jcp.2022.111090>.
- [20] “Bayesian regularization backpropagation - MATLAB,” , 2022. URL <https://www.mathworks.com/help/deeplearning/ref/trainbr.html>.
- [21] Skamarock, W. C., Klemp, J. B., Dudhia, J., Gill, O., Liu, Z., Berner, J., Wang, W., Powers, G., Duda, G., Barker, D. M., and Yu Huang, X., “A Description of the Advanced Research WRF Model Version 4,” *NCAR Technical Notes*, 2021. <https://doi.org/http://dx.doi.org/10.5065/1dfh-6p97>.
- [22] Rotunno, R., Chen, Y., Wang, W., Davis, C., Dudhia, J., and Holland, G. J., “Large-Eddy Simulation of an Idealized Tropical Cyclone,” *Bulletin of the American Meteorological Society*, Vol. 90, No. 12, 2009, pp. 1783 – 1788. <https://doi.org/10.1175/2009BAMS2884.1>, URL https://journals.ametsoc.org/view/journals/bams/90/12/2009bams2884_1.xml.
- [23] Piomelli, U., and Balaras, E., “Wall-Layer Models for Large-Eddy Simulations,” *Annual Review of Fluid Mechanics*, Vol. 34, No. 1, 2002, pp. 349–374. <https://doi.org/10.1146/annurev.fluid.34.082901.144919>.
- [24] Mackay, D. J. C., “Bayesian Interpolation,” *Neural Computation*, Vol. 4, 1992, pp. 415–447.
- [25] Dan Foresee, F., and Hagan, M., “Gauss-Newton approximation to Bayesian learning,” *Proceedings of International Conference on Neural Networks (ICNN’97)*, Vol. 3, 1997, pp. 1930–1935 vol.3. <https://doi.org/10.1109/ICNN.1997.614194>.
- [26] Zhou, X.-H., Han, J., and Xiao, H., “Frame-independent vector-cloud neural network for nonlocal constitutive modeling on arbitrary grids,” *Computer Methods in Applied Mechanics and Engineering*, Vol. 388, 2022, p. 114211. <https://doi.org/https://doi.org/10.1016/j.cma.2021.114211>.

- [27] Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A., “Neural Operator: Graph Kernel Network for Partial Differential Equations,” , 2020. <https://doi.org/10.48550/ARXIV.2003.03485>, URL <https://arxiv.org/abs/2003.03485>.