CS230

A deep multi-task learning approach for wall modeling of compressible turbulent boundary layers

Rahul Agrawal[*], **Tim J. Flint**[†], **Christopher T. Williams** [‡]
Center for Turbulence Research, Mechanical Engineering,
Stanford University
Code Repository: https://github.com/ct-williams/TCBLNet.git

Abstract

Skin-friction and heat transfer are the two primary quantities of engineering interest for high-speed flight vehicles. Direct numerical solution of the compressible Navier Stokes equations can be utilized to predict the heat flux and wall stress in these flight systems, but presently, these calculations are exceedingly intractable on even the largest supercomputers. Therefore, the current study seeks to evaluate the feasibility of combing a traditional high-speed flow physics solver with deep learning to obtain accurate predictions of shear stress and aerodynamic heating for aerospace vehicles at a significantly reduced computational expense. Both traditional feedforward networks, as well as deeper residual networks, are considered in the present study, with both architectures demonstrating promise for instantaneous prediction of heat flux and wall stress for coarse-grained calculations of turbulent hypersonic boundary layers.

1 Introduction

The formation of boundary layers, which can be characterized as near-wall regions dominated by viscous effects, directly leads to skin-friction drag, and in the case of hypersonic flight, kinetic energy dissipation in the boundary layer is responsible for aerothermodynamic heating. Moreover, in hypersonic flight, the flow over the aircraft often involves the formation of shock waves, as well as the onset of turbulence in the boundary layer, yielding a significant increase in heating, as well as corresponding fluctuations in temperature, velocity and pressure [2, 18]. With turbulent boundary layers developing on the fuselage of high-speed flight vehicles, efficient design of thermal protection systems and vehicle aerodynamics requires the accurate numerical simulation of hypersonic boundary layers at a feasible computational cost. Early attempts at reduced-order modeling of high-speed turbulent boundary layers were performed by [7, 8, 16], using the Reynolds-averaged Navier-Stokes equations (RANS), but only mixed success has been achieved with this methodology. Indeed, recent studies [1] have demonstrated that the RANS approach fails to accurately predict both heat transfer and skin friction for turbulent boundary layers in the high-Mach number regime, which is the focus of the current study. Recently, however, data-driven techniques, including deep learning, have been utilized [10, 12, 20] to model incompressible turbulent flows in which density, viscosity, and thermal conductivity variations are negligible. The pioneering study of [20] applied deep learning to the reduced-order technique of wall modeling [3, 4], advancing a coarsened Navier-Stokes solution in time with the trained neural network providing the flux boundary conditions. The current work seeks

[*]rahul29@stanford.edu
[†]timflint@stanford.edu
[‡]ctwilliams@stanford.edu

CS230: Deep Learning, Spring 2022, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

to extend the work of [20] by developing a deep-learned wall model for turbulent hypersonic boundary layers, for which the assumption of incompressibility is no longer valid. As such, the ultimate goal of the present study is to develop a neural network that can interpret a coarse/under-resolved numerical solution of a compressible boundary layer and output a corrected heat flux and wall stress, thereby reducing the computational cost relative to a fully-resolved direct numerical simulation.

2 Pre-Processing

The true labels, or the ground truth values of the heat flux and wall stress values for each boundary layer can be obtained directly from the DNS database described in Table 1. To mimic the numerical procedure involved in simulating coarse grained calculations of turbulent flows, we have filtered and downsampled the true solution. In this current work, the filtering procedure has been performed using a box filter with three vanishing moments [19]. Then, the filtered and downsampled data is accepted as an output of a coarse calculation on which one can train, develop, and evaluate a neural network. The workflow of pre-processing the boundary-layer data is as follows:

- By definition, a lower fidelity simulation does not resolve all of the smallest flow features, and instead seeks only to predict large-scale behavior. To generate a proxy for low-fidelity fields, we have undersampled (by up to 8 times in each spatial dimension) and then explicitly filtered the collected DNS fields. These fields will hereafter be referred to as LES (Large-eddy simulation) fields, owing to the fact that all fine-grained flow structures have since been eliminated. For reference, the computational cost associated with calculating a coarse-grained solution with this resolution would be 1000x less expensive relative to the fully-resolved direct numerical simulation. Further details on coarse-grained (LES) calculations can be found in [6].

- Selected quantities from the LES fields are then chosen as input to the neural network (examples include filtered velocity and temperature fields and their gradients in the three coordinate directions, and positional information for the relevant scalar fields). In total, then, the dimension of the input vector is presently 55. The two labels for each example are the true wall stress (τ_w) and heat flux (q_w) from the fully-resolved DNS.

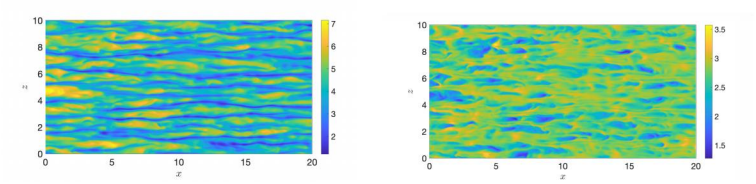
- Once this filtering procedure is completed, multiple LES fields (resulting from different DNS fields and extent of filtering) are concatenated and shuffled together to form multiple mini-batches. This shuffling ensures that each gradient descent step during training is informed by both coarse and relatively fine calculations at a variety of Reynolds and Mach numbers, accelerating the training process.

3 Dataset

A high-fidelity data set of direct numerical simulations (DNS) has been aggregated for high-speed turbulent boundary layers: these boundary layer calculations fully resolve the compressible flow physics, and therefore their heat flux and shear stress predictions can be considered as a ground truth. Moreover, the database is comprised of numerous large-scale simulations with varying physical parameters, such as wall temperature (hot or cold wall relative to overriding flow), flight velocity relative to the speed of sound (Mach number), and surrounding medium viscosity (Reynolds number). In this section, visual examples (contours, ref. to Fig.1) are provided for the raw DNS streamwise velocity and temperature fields. Both large and small scale physical features (streaks) are strikingly present in these fields. We have also provided an example of the filtered and coarse grained LES fields for comparison (ref. Fig.2). It is also apparent that these coarse grained flow fields lack most of the small scale structures in comparison to DNS fields. The final database of pre-processed fields contains 5400 mini-batches, with 256 examples per mini-batch. The labels for each mini-batch correspond to the wall stress and heat flux from the direct numerical simulation. For the present study, 55 features from the coarsened boundary layer fields were used as input to the neural networks, corresponding to the position, velocity, temperature, pressure, density, viscosity and conductivity at five near-wall locations in the coarsened boundary layer fields. While 3600 mini-batches are taken as training data, both the development and test data sets are each comprised of 900 mini-batches.

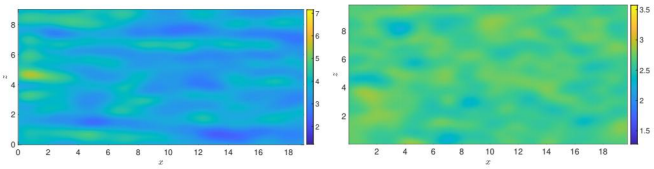
Case Name	Case Purpose	Re	Ma	T_w/T_{aw}
DNS_{01}	Training Set	1.5×10^5	5.84	0.25
DNS_{02}	Training Set	1.0×10^4	5.00	0.20
DNS_{03}	Training Set	2.0×10^4	5.00	0.60
DNS_{04}	Training Set	1.5×10^5	6.00	0.25
DNS_{05}	Training Set	1.0×10^4	7.00	0.20
DNS_{06}	Training Set	2.0×10^4	7.00	0.30
DNS_{07}	Training Set	4.0×10^4	7.00	0.30
DNS_{08}	Test/Dev Set	1.5×10^5	7.00	0.20
DNS_{09}	Test/Dev Set	3.0×10^4	11.00	0.30

Table 1: A summary of all the DNS databases used for preprocessing data, which was then fed to the neural network in this study. Note that $Re,\ Ma,\ T_w/T_{aw}$ are the Reynolds number, Mach number and the ratio of wall to adibatic-wall temperatures respectively. The test and dev sets were chosen to share the highest Reynolds and Mach numbers for providing the network unseen and challenging cases, and for fair comparisons w.r.t. existing physics-based models (ref. to the appendix for more details).



(a) (b)

Figure 1: Visualization of a near wall, wall parallel plane of a high fidelity simulation (a) velocity and (b) temperature fields.



(a) (b)

Figure 2: Visualization of the same wall parallel plane as Figure 1, for a filtered, low fidelity simulation (a) velocity and (b) temperature fields.

4 Technical Approach

The inherent symmetry between heat flux and shear stress in continuum compressible gas dynamics (in that both the quantities are first order derivatives of the primary flow variables like velocity, density and temperature) allows use of a multi-task deep learning architecture. Consequentially, feedforward and residual neural networks have been the primary architectures considered in this study. As we are interested in quantifying the error in this regression problem, the loss function for training the neural network is chosen to be the average of the Euclidean norm between predicted and true output wall stresses and heat fluxes across the training examples. Mathematically, for a set of m examples, this is equivalent to

$$\mathcal{J}(\tau_w, \widehat{\tau}_w, q_w, \widehat{q}_w) = \frac{1}{m}\sum_{i=1}^{m} |\tau^{(i)}_{w,DNS} - \widehat{\tau}^{(i)}_w|^2 + \frac{1}{m}\sum_{i=1}^{m} |q^{(i)}_{w,DNS} - \widehat{q}^{(i)}_w|^2 \tag{1}$$

where $\mathcal{J}$ is the cost function, $\tau_w,\ q_w$ are the wall stress and heat fluxes respectively. The subscript, $_{DNS}$, and the hat, $\widehat{\cdot}$, represent true and predicted label quantities respectively. It should be emphasized that here 'm' refers to the number of examples per mini-batch. The optimizer employed for the

current study is the Adam algorithm with weight decay, as implemented by default in PyTorch [13]:

$$W_t := W_{t-1} - \gamma m_t / \left(\sqrt{v_t} + \epsilon\right) \tag{2}$$

$$v_t := \left(\beta_2 v_{t-1} + (1-\beta_2) g_t^2\right) / (1-\beta_2^t)$$

$$m_t := \left(\beta_1 m_{t-1} + (1-\beta_1) g_t\right) (1-\beta_1^t)$$

where the subscript t refers to the iteration number, g is the gradient of the cost function with respect to W, λ is the weight decay value, γ is the learning rate, and ϵ is merely a small parameter to avoid division by zero, taken to be 1×10^{-8} for the current study. The hyperparameters of β_1 and β_2 were likewise held constant at 0.900 and 0.999, respectively. The first architecture considered in the present study is the standard feedforward neural network, consistent with the approach of [20]. Indeed, the present study has considered the exact architectures (see Table 4 in the Appendix) from [20], training those networks on the more challenging compressible boundary layer problem. Additionally, the present study has also considered deeper neural networks, for which the even-numbered layers are connected by skip connections. All networks considered in the present study have a linear output layer, whereas all other activation functions are taken to be Leaky Rectified Linear Units (ReLU). The code developed for the current study can be found using the GitHub link placed above. Finally, Table 3 in the Appendix contains the exact configurations in terms of neurons per layer, as well as the hyperparameters explored for the present study. Learning rate, number of training epochs, magnitude of weight decay, neurons per layer, and number of layers were all varied across the different neural networks for the present study.

5 Results

A summary of the results from the present study are included in Table 2, indicating the training, development, and test errors associated with each architecture and choice of hyperparameters. Let us define the error metric as

$$E(\xi) = \frac{\|\hat{\xi} - \xi\|}{\frac{1}{2}\left(\|\hat{\xi}\| + \|\xi\|\right)} \tag{3}$$

where $\hat{\xi}$ denotes the prediction of a quantity by the neural network while ξ denotes the corresponding label. Therefore, the error with respect to heat flux is defined as $E(q_w)$, which avoids singularities in the case of adiabatic boundary layers. The expression for wall stress error is an analogous expression, corresponding to $E(\tau_w)$. In comparing the relative errors of the neural networks in Table 2, which are arithmetically averaged across all relevant examples, it becomes clear that both the relatively shallow networks of [20], as well as the deeper networks with skip connections, perform well in predicting the heat flux and wall stress, given the right combination of hyperparameters. In particular, both NN_{09} and NN_{12} perform quite well across the training, development, and test sets, with errors for both heat flux and wall stress in the vicinity of 30 percent. The evolution of the cost functions vs. epoch during training is included in the Appendix for NN_{12}. Indeed, though the errors associated with NN_{09} are slightly improved relative to NN_{12}, whose was configuration was taken from [20], the computational expense for the latter model is dramatically smaller. As such, for a practical fluid mechanics calculation, NN_{12} would most likely be the preferable model. In comparison with the errors typical of traditional wall models (see the Appendix), most of the neural networks in the current study are performing quite well. Relative to traditional wall modeling techniques from the fluid mechanics literature, the networks trained in the present study obtain slightly higher errors for adiabatic cases, but for boundary layers with significant heat transfer, all of the deep learning models developed in the present study are competitive, with the traditional wall models' errors exceeding, in certain cases, 80 percent and 300 percent for heat flux and wall stress, respectively.

Moreover, the relatively consistent performance of the neural networks across training and dev/test sets is particularly admirable since the dev/test sets are comprised of Mach and Reynolds number combinations unseen in training. For example, the highest Mach number seen in training is 7, but profiles from a Mach 11 boundary layer simulation constitutes approximately half the training/dev examples. Therefore, most of the neural networks are generalizing quite well, particularly those which employed weight decay during training, indicating the regularizing effect is significant for the present problem. In evaluating the relative performance of the different models, it would also seem that a learning rate between 0.001 and 0.005 is optimal for the current data and choice of mini-batch

size, while the best performance is obtained when training for a duration between 20 and 30 epochs to minimize overfitting (plots provided in the appendix). Contours of heat flux and wall stress labels from the test set, juxtaposed with the corresponding predictions from the deep-learned models, are included in the Appendix. From inspection, the predicted contours of heat flux and wall stress bear appreciable resemblance to the respective DNS contours.

Network code	Training Error (τ_w)	Training Error (q_w)	Dev. Error (τ_w)	Dev. Error (q_w)	Test Error (τ_w)	Test Error (q_w)
NN_{01}	0.28	0.23	0.34	0.27	0.34	0.27
NN_{02}	0.43	0.58	0.52	0.71	0.52	0.71
NN_{03}	0.46	0.26	0.42	0.33	0.42	0.33
NN_{04}	0.68	0.23	0.78	0.30	0.78	0.31
NN_{05}	0.36	0.29	0.32	0.33	0.32	0.33
NN_{06}	0.27	0.26	0.30	0.34	0.30	0.35
NN_{07}	0.32	0.29	0.44	0.32	0.44	0.31
NN_{08}	0.33	0.25	0.38	0.39	0.39	0.40
NN_{09}	0.27	0.24	0.32	0.32	0.32	0.32
NN_{10}	0.38	0.23	0.34	0.32	0.34	0.32
NN_{11}	0.30	0.26	0.34	0.36	0.34	0.36
NN_{12}	0.26	0.25	0.30	0.32	0.30	0.32
NN_{13}	0.30	0.24	0.57	0.42	0.57	0.43

Table 2: A summary of the training, dev. and test set errors in the prediction of wall stress and heat fluxes for all the neural networks considered.

6 Conclusion/Future Work

In conclusion, the present study has identified a relatively shallow a feedforward network, NN_{12}, as well as much deep network with skip connections,NN_{09}, that accurately predict the heat flux and wall stress induced by a turbulent hypersonic boundary layer, given a coarse-grained solution field. Relative to traditional wall-modeling techniques, the deep-learned models of the current study perform quite well in the case of high Mach number boundary layers with significant heat transfer. Future work will primarily entail gathering more DNS data for hypersonic flows, especially from flow over more complex geometries that bear a stronger resemblance to realistic flight vehicle configurations. Furthermore, while the current study focuses on *a priori* analysis of neural networks on coarsened DNS calculations, future work will pursue *a posteriori* calculations in which the neural networks provide flux boundary conditions to advance wall-modeled solution in time, as was pursued in [20]. Likewise, pending the performance of the current architectures for more complicated flows, more sophisticated networks such as Neural ODE's [14, 5, 15, 11] may be considered, as well.

7 Contributions

R.A., T.J.F. and C.T.W. equally contributed to the initial conceptualization of this work. R.A. and T.J.F. worked on the pre-processing codes and arrangement of data for feeding into the neural net. C.T.W. was responsible for the development of the PyTorch code, and for the training and evaluation of the neural networks as part of hyperparameter tuning. Technical discussions were equally contributed to, by all of the team members, which led to the multiple iterations across choice of hyperparameters (and their tuning), size of the network, and interpretation of the results. This report, and the final project video were also contributed to, equally, by all of the team members.

8 Acknowledgements

The authors sincerely thank Dr. Mario Di Renzo of CERFACS for generously providing us with the DNS fields considered in the present study.

References

[1] T. T. Aiken, I. D. Boyd, L. Duan, and J. Huang. Assessment of reynolds averaged navier-stokes models for a hypersonic cold-wall turbulent boundary layer. In *AIAA SCITECH 2022 Forum*, page 0586, 2022.

[2] J. J. Bertin and R. M. Cummings. Critical hypersonic aerothermodynamic phenomena. *Annu. Rev. Fluid Mech.*, 38:129–157, 2006.

[3] S. T. Bose and P. Moin. A dynamic slip boundary condition for wall-modeled large-eddy simulation. *Phys. Fluids*, 26(1):015104, 2014.

[4] S. T. Bose and G. I. Park. Wall-modeled large-eddy simulation for complex turbulent flows. *Annu. Rev. Fluid Mech*, 50:535–561, 2018.

[5] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud. Neural ordinary differential equations, 2018.

[6] L. Fu, M. Karp, S. T. Bose, P. Moin, and J. Urzay. Shock-induced heating and transition to turbulence in a hypersonic boundary layer. *J. Fluid Mech*, 909, 2021.

[7] J. Fulton, J. Edwards, H. Hassan, J. McDaniel, C. Goyne, and R. Rockwell. Continued hybrid les/rans simulation of a hypersonic dual-mode scramjet combustor. In *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, page 117, 2013.

[8] C. Jordan, J. R. Edwards, and D. L. Stefanski. Evaluation of rans closure models using les datasets for hypersonic shock boundary layer interactions. In *AIAA Scitech 2021 Forum*, page 1666, 2021.

[9] S. Kawai and J. Larsson. Wall-modeling in large eddy simulation: Length scales, grid resolution, and accuracy. *Physics of Fluids*, 24(1):015105, 2012.

[10] H. Kim, J. Kim, S. Won, and C. Lee. Unsupervised deep learning for super-resolution reconstruction of turbulence. *Journal of Fluid Mechanics*, 910, 2021.

[11] K. Lee and E. J. Parish. Parameterized neural ordinary differential equations: applications to computational physics problems. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 477(2253):20210162, 2021.

[12] J. Ling, A. Kurzawski, and J. Templeton. Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics*, 807:155–166, 2016.

[13] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[14] M. Poli, S. Massaroli, A. Yamashita, H. Asama, J. Park, and S. Ermon. Torchdyn: Implicit models and neural numerical methods in pytorch. *NeurIPS*, 2021.

[15] C. J. G. Rojas, A. Dengel, and M. D. Ribeiro. Reduced-order model for fluid flows via neural ordinary differential equations, 2021.

[16] K. Sinha, M. Martin, and G. Candler. Assessment of temperature fluctuation models for rans simulations of hypersonic reacting flows. In *38th Aerospace Sciences Meeting and Exhibit*, page 537, 2000.

[17] A. Trettel and J. Larsson. Mean velocity scaling for compressible wall turbulence with heat transfer. *Physics of Fluids*, 28(2):026102, 2016.

[18] J. Urzay. Supersonic combustion in air-breathing propulsion systems for hypersonic flight. *Annual Review of Fluid Mechanics*, 50:593–627, 2018.

[19] O. V. Vasilyev, T. S. Lund, and P. Moin. A general class of commutative filters for LES in complex geometries. *Journal of Computational Physics*, 146(1):82–104, 1998.

[20] X. Yang, S. Zafar, J.-X. Wang, and H. Xiao. Predictive large-eddy-simulation wall modeling via physics-informed neural networks. *Physical Review Fluids*, 4(3):034602, 2019.

9 Appendix

9.1 Description of the Network Architecture

Network code	Neurons per Layer	Learning Rate (α)	No. epochs	Weight Decay ratio
NN_{01}	55, 40x6, 20x6, 10x8, 2	0.001	25	0.10
NN_{02}	55, 40x10, 20x10, 10x10	0.001	40	0.50
NN_{03}	55, 40x10, 20x10, 10x10	0.01	25	0.01
NN_{04}	55, 40x4, 20x4, 10x4, 2	0.01	25	0.01
NN_{05}	55, 80 x 4, 60 x 4, 40 x 4, 20 x 4, 10 x 4, 2	0.01	25	0.01
NN_{06}	55, 80 x 2, 60 x 2, 40 x 2, 20 x 2, 10 x 2, 2	0.001	35	0.10
NN_{07}	55, 100x2, 80x2, 40x2, 20x2, 10x2,2	0.01	25	0.10
NN_{08}	55, 100x2, 50x2, 25x2, 12x2, 2	0.01	20	0.01
NN_{09}	55, 120x2, 100x2, 80x2, 60x2, 40x2, 20x2, 12x2, 2	0.005	30	0.01
NN_{10}	55, 60x6, 30x4, 20x2, 10x4, 2	0.005	25	0.01

Table 3: A summary of the residual neural networks considered. For the configuration section, note that $X \times Y$ here refers to a layer with X neurons repeated Y times.

Network code	Neurons per Layer	Learning Rate (α)	No. epochs	Weight Decay ratio
NN_{11}	55,4,2,2,2	0.005	20	0.01
NN_{12}	55,6,4,3,3,2	0.005	20	0.01
NN_{13}	55,8,8,6,4,4,2	0.005	20	0.01

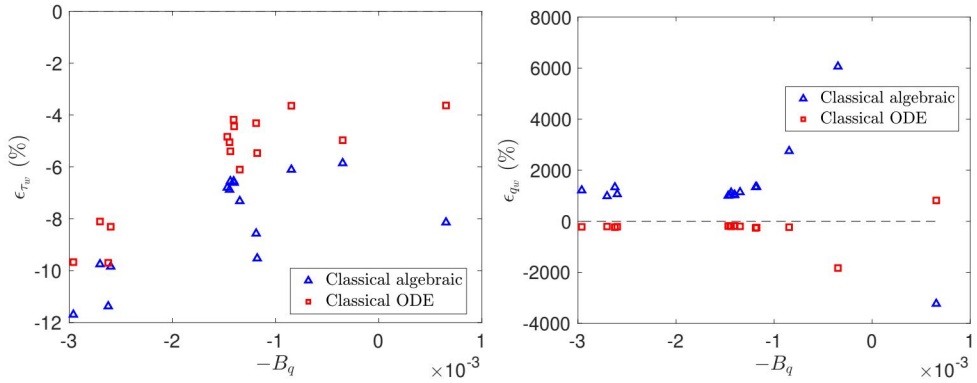
Table 4: Feedforward neural network architectures considered in the present study from [20]

9.2 Errors in the prediction of wall shear stress and heat flux from existing wall models

The existing physics based wall models [9, 17] for compressible flows have been shown to work accurately in simple cases, where the effects of compressibility were rather minimal [9]. However, in the case of a hypersonic flow, the errors in the correct prediction of the wall stress and heat fluxes are not negligible, and as a result, these models can not be used in ongoing research of such flows. Figs. 3, 6 present the errors (quantified as the relative L_2 norm of the difference between the modeled and true output values, normalized with the true values) from the existing models for both adiabatic (no heat transfer from/to the wall) and diabatic (allows heat transfer) boundary layer. It is very apparent that as $-B_q$, or the magnitude of wall heating increases, these existing state of the art physics inspired models perform very poorly. In this light, the results presented in this work are an improvement over the existing wall models, in that even for Mach 11 turbulent flow, with very significant wall heating, the deep-learning based model can perform up to reasonable accuracy.

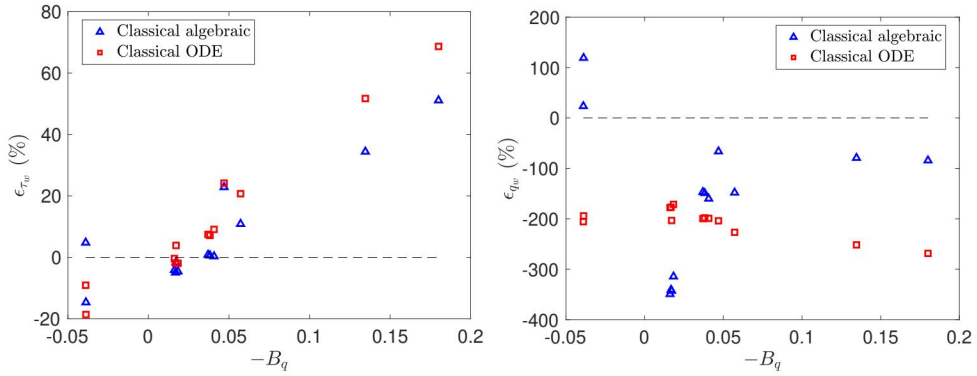
9.3 Training loss as a function of epoch

Cost function vs epoch number for the the neural network NN_{12} is provided in this section. Since the cost appears to have approached an asymptote, additional training was not pursued in order to avoid any over-fitting.



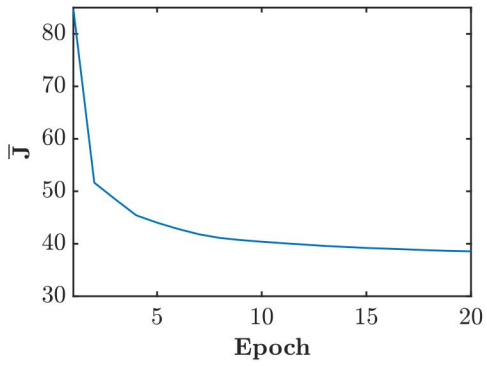
(a) (b)

Figure 3: Errors in the prediction of (a) wall shear stress and (b) heat flux for an adiabatic (does not allow heat transfer from/to the wall) boundary layer, as a function of the normalized (by the local wall quantities such as density, heat conductivity, temperature and velocity)-true heat flux. These figures have been modified from the ongoing Ph.D. thesis work of Kevin P. Griffin at CTR, Stanford.



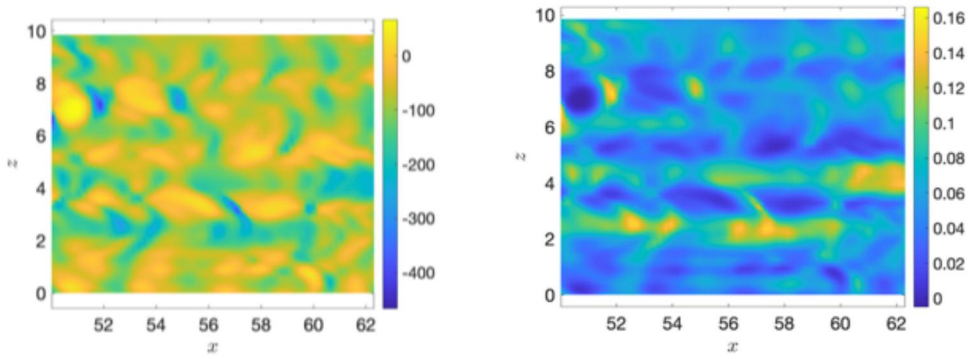
(a) (b)

Figure 4: Errors in the prediction of (a) wall shear stress and (b) heat flux for a diabatic (allows heat transfer) boundary layer, as a function of the normalized (by the local wall quantities such as density, heat conductivity, temperature and velocity)-true heat flux. These figures have been modified from the ongoing Ph.D. thesis work of Kevin P. Griffin at CTR, Stanford.



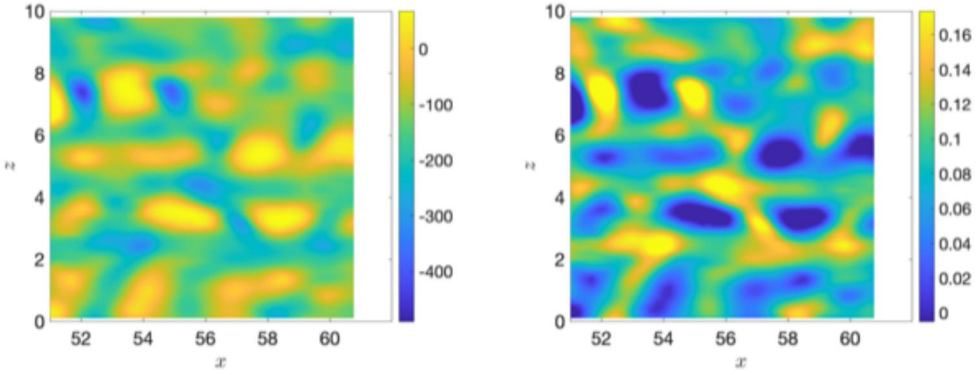
(a)

Figure 5: Cost function, averaged across batches, vs. epoch for NN_{12}



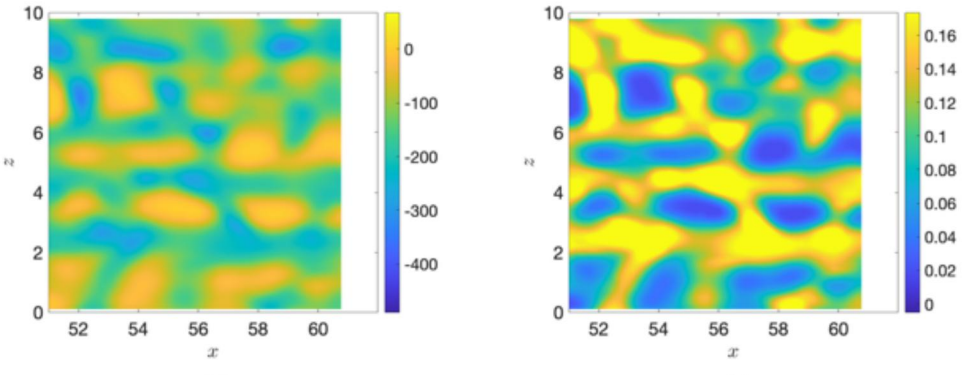
(a) (b)

Figure 6: Heat flux (a.) and wall stress (b.) from the direct numerical simulation results for the Mach 11 boundary layer.



(a) (b)

Figure 7: Heat flux (a.) and wall stress (b.) for the Mach 11 boundary layer, as predicted by the NN_{09}



(a) (b)

Figure 8: Heat flux (a.) and wall stress (b.) for the Mach 11 boundary layer, as predicted by the NN_{12}