AIAA SciTech Forum
January 3-7, 2022, San Diego, CA & Virtual
AIAA SCITECH 2022 Forum

10.2514/6.2022-0508

Surrogate Unsteady Aerodynamic Modeling with Autoencoders and Long-Short Term Memory Networks

Hüseyin Emre Tekaslan* and Yusuf Demiroğlu †
Istanbul Technical University, Maslak, Istanbul, 34469, Turkey

Melike Nikbay ‡
Istanbul Technical University, Astronautical Engineering Department, Maslak, Istanbul, 34469, Turkey

This paper presents preliminary results of an ongoing research in prediction of time-dependent flow fields by focusing on data-driven surrogate modeling using artificial neural networks for unsteady aerodynamic problems. The aim of this research is to model unsteady flow fields with learning in low-dimensional space and reconstruct with recurrent autoencoders. Within the scope of this paper, we separately share our findings in viscous unsteady flow field reconstruction of a 2D cylinder in a channel with a deep autoencoder and unsteady aerodynamic-acoustic time-series prediction of the supersonic NASA C25D aircraft with shallow long short-term memory networks. Satisfactory results are achieved in both unsteady applications, yet further improvements and validations are needed to be achieved to establish the desired surrogate unsteady aerodynamic modeling for supersonic aircraft maneuvers.

I. Introduction

Reducing travel time for air transport is one of the recent technical challenges in the aerospace industry. As advanced propulsion systems and aerodynamic designs enable efficient and affordable high speed flow, supersonic flight is becoming more attainable. Potential demand in both supersonic civil transportation and the business jet market is estimated to grow in the following decades [1, 2] . Researchers employ the current state-of-the-art high-fidelity numerical solvers that require a huge number of computational power to predict aerodynamic and acoustic characteristics of civil supersonic air vehicles [3, 4]. However, especially during design optimization processes, uncertainty quantification studies and modeling of unsteady dynamic motions, a huge number of samplings are required where the use of high-fidelity solvers is not feasible. As a remedy, reduced-order modeling methods such as artificial neural networks can be utilized to reduce the required time. Fast and accurate predictions of the aerodynamic and acoustic properties of a supersonic aircraft are crucial for the design studies of a supersonic civil transport [5–8]. A dataset that consists of a large number of sampling is required to establish models for unsteady dynamic motions for a supersonic aircraft. For the flight dynamic models to be used in the dynamic stability analysis, several unsteady analyses must be conducted with respect to different input signals. Here, to eliminate the computational burden resulting from unsteady analyses, a neural network can be trained to estimate the unsteady behavior of the aircraft in terms of aerodynamic properties and sonic boom loudness with the given input signals.

Artificial neural networks have emerged a new era in computer and computational sciences with a rapidly increasing data accumulation and computational power. Practical conundrums back then a few decades such as image, video & audio recognition and machine translation can now be resolved with an adequate size of data and neural networks. This outstanding data-driven metamodeling technique has been disseminated over time and become ubiquitous across different disciplines such as bioinformatics [9–11], geosciences [12–14], and aerospace [15–17]. There are previous studies exploiting neural network in the prediction of flow field while a comprehensive review of aerodynamic modeling with artificial neural network is presented in [18].

In this paper, we present preliminary results of an ongoing research in prediction of time-dependent flow fields. Two separate applications with convolutional autoencoders and recurrent neural networks are considered. The knowledge obtained with these applications in flow field reconstruction and time-series prediction will be combined in the future studies. Firstly, we respectively present mathematical backgrounds of convolutional autoencoders, recurrent neural

*Graduate Student, Research Scholar, AeroMDO Lab, Faculty of Aeronautics and Astronautics, AIAA Student Member
†Graduate Student, Research Scholar, AeroMDO Lab, Faculty of Aeronautics and Astronautics, AIAA Student Member
‡Professor, Faculty of Aeronautics and Astronautics, AIAA Associate Fellow

network, and aerodynamic & acoustic solution in sections II and III. In Section IV, applications of 2D unsteady viscous flow field reconstruction of a cylinder in a channel with a deep convolutional autoencoder and sequence prediction of aerodynamic & acoustic variables with a shallow Long-Short Term Memory (LSTM) network. Then, we briefly summarize the findings and further study in Section V.

II. Neural Networks

A. Convolutional Autoencoders

Convolutional autoencoders use encoder and a following decoder architecture to compress the given data and then decompress it with convolution and transpose convolution layers. This method is analogous to proper orthogonal decomposition (POD) where a dataset is shrunk by projecting it onto an optimal subspace and then reconstructed with POD coefficients to the original size. A convolution layer maps the input data by using learnable filters into a different low-dimensional hyperplane. A convolution function can shown as in Eq. (1).

$$f(\boldsymbol{x}) = \{\boldsymbol{k} \odot \bar{\mathbf{x}} + \boldsymbol{b} \mid \bar{\mathbf{x}} \subseteq \boldsymbol{x}\} \tag{1}$$

where $\boldsymbol{x}$ is the input, $\bar{\mathbf{x}}$ is a submatrix of input and $\boldsymbol{b}$ is the bias matrix. Additionally, $\boldsymbol{k}$ is the filter that strides all over the input and element-wise multiplied with the overlapping input to obtain feature maps. The input is squeezed based on filter size and stride because at each stride a convolutional operation yields a single value. On the contrary, a transpose convolution layer used in the decoder part takes a single element of a given input and results in a high-dimensional submatrix. Therefore, concatenated convolutional layers are used to obtain a low-dimensional representation of high-dimensional data. Convolution and transpose convolution operations are illustrated in Fig. 1.

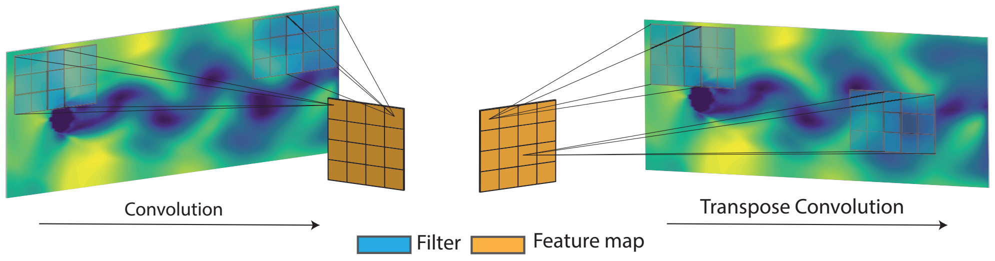


Fig. 1 Demonstration of convolution (left) and transpose convolution (right) layers.

B. Recurrent Neural Networks

Recurrent neural networks (RNN) are deep learning methods that are utilized in the prediction of a sequence of data. An RNN layer can be expressed as a function, $f : \mathcal{X} \to \mathcal{Y}$, with spatio-temporal input, $\mathcal{X} \in \mathbb{R}^{N_\mathcal{X}}$, and output $\mathcal{Y} \in \mathbb{R}^{N_\mathcal{Y}}$ domains with $\boldsymbol{N}_i = \{N_1^i \times N_2^i \times \ldots \times N_d^i \times N_{S_i} \mid i : \mathcal{X}, \mathcal{Y};\ d \in \mathbb{Z}^+;\ N_{S_i} \in \mathbb{Z}^+\}$ where d and N_S are the number of spatial dimensions and sequence length, respectively. Explicitly expressing the RNN layer as function of input, $\mathbf{x}_t$ at time t and the hidden state $\mathbf{h}_{t-1}$ with size N_h at time $t-1$,

$$f(\mathbf{x}_t, \mathbf{h}_{t-1}) = \mathbf{h}_t = g(\mathbf{W}_h \mathbf{h}_{t-1} + \mathbf{W}_x \mathbf{x}_t + \mathbf{b}_h) \tag{2}$$

where $g(.)$ is a nonlinear activation function, $\mathbf{W}_h$, $\mathbf{W}_x$, and $\mathbf{b}_h$ are learnable weight and bias matrices of a network. A hidden state is a matrix containing the short-term information dependencies and can be initialized with different schemes at time $t = 0$. Additionally, the relation between the input and output sequence lengths determines the type of RNN. The simplest usage of a recurrent network is to make one-to-one predictions without recurrence that means an input with a single time step passing through the network yields an output with a single time step; thus, $N_{S_\mathcal{X}} = N_{S_\mathcal{Y}}$. One-to-many, many-to-one, and many-to-many (seq2seq) type of RNNs can also be trained in case of $N_{S_\mathcal{X}} \neq N_{S_\mathcal{Y}}$. In the concept of a recurrent neural network, learnable parameters are optimized such that the network learns the conditional likelihood of a sequence.

$$L(\theta; \mathbf{y} \mid \mathbf{x}) = \prod_{t=1}^{N_{S_\mathcal{Y}}} p\left(\mathbf{y}_t \mid \mathbf{x}_{t-1}, \ldots, \mathbf{x}_1; \theta\right) \tag{3}$$

where $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_{S_X}}\} \in \mathcal{X}$ is an input sequence and $\boldsymbol{y}_t$ is the datum at time t and an element of $\boldsymbol{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{N_{S_y}}\} \in \mathcal{Y}$. The optimal network parameter set, $\theta = \{\mathbf{W}_h, \mathbf{W}_x, \mathbf{b}\}$, maximize the conditional likelihood of input and output sequences. So, basically RNNs are trained in a way that they determine what the output with the maximum likelihood should be based on the history of a sequence.

$$\theta = \arg\max L(\theta; \mathbf{y} \mid \mathbf{x}) \tag{4}$$

Furthermore, stochastic gradient descent algorithms are broadly used to find out the sought-after network parameters with a gradient computation algorithm referred to as backpropagation through time (BPTT) [19]. BPTT is a scheme to compute time-dependent gradients of an unfolded RNN architecture. For example, the derivative of the loss function, $\mathcal{L}_{\theta,t}$, with parameter set, θ, at time t with respect to weight $\mathbf{W}_h$ is calculated as in (5).

$$\frac{\partial \mathcal{L}_{\theta,t}}{\partial \mathbf{W}_h} = \sum_{k=1}^{t} \frac{\partial \mathcal{L}_{\theta,t}}{\partial \hat{\mathbf{y}}_t} \frac{\partial \hat{\mathbf{y}}_t}{\partial \mathbf{h}_t} \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_k} \frac{\partial \mathbf{h}_k}{\partial \mathbf{W}_h} \tag{5}$$

where $\hat{\mathbf{y}}_t$ is the prediction at time t. Moreover, BPTT has a drawback especially in the case of training neural networks with a long sequence of data. This scheme may bring about gradients either with unbounded upper limits or approaching zero known as exploding and vanishing gradients respectively. Gradient clipping was proposed in [20] as a remedy for exploding gradients whereas long short-term memory -an improved RNN cell- that alleviates both exploding and vanishing gradient problem was introduced in [21].

C. Long Short-Term Memory

Long short-term memory (LSTM) is an improved RNN cell that has a special gated additive structure that brings enhancements in exploding and vanishing gradient problems, unlike a vanilla RNN cell. In addition to a hidden state, an LSTM network passes cell state, $\mathbf{c}$, as well throughout the network. Unlike a hidden state, a cell state is a memory matrix that carries long-term information that latently exists in the sequence of data. An LSTM cell consists of 3 gates: forget gate, input gate, and output gate. Each of the gates performs an interpretable mathematical operation to a given input. To begin with, the forget gate, as its name reveals, determines what information within the time series to forget in the long run. It uses a sigmoid function to map its input to the range of $(0, 1)$. The information converged to 0 is forgotten where the information worth remembering is mapped closer to 1 and then the previous cell state is updated with a Hadamard product. Secondly, the input gate controls which new information should be let through to the cell state by using sigmoid and tangent hyperbolic functions. Afterward, it adds worthwhile information to the current cell state. Lastly, the next hidden state is formed by the output gate using the latest cell state, input, and the previous hidden state. An elaborated cell structure of the LSTM layer is depicted in Fig. 2. In this figure, each gate is encompassed with a dashed box to clarify the structure. Mathematical representations of gates are given with Eq. (6). We use f_t, i_t, and o_t to respectively refer to forget, input, and output functions with a stacked input $[\mathbf{h}_{t-1}, \mathbf{x}_t]$. Sigmoid function is given as $\sigma(.)$ whereas we designate tangent hyperbolic function as $\tanh(.)$. Additionally, parameter set $\theta = \{\mathbf{W}_f, \mathbf{W}_i, \mathbf{W}_o, \mathbf{b}_f, \mathbf{b}_i, \mathbf{b}_o\}$ involves learnable weights and biases of gates to be optimized. The arithmetic operation symbol, $\odot$, corresponds to Hadamard product. Lastly, new cell state that updates the current cell state is given as $\tilde{c}_t$.

$$\begin{aligned}
f_t &= \sigma(\mathbf{W}_f[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f) \\
i_t &= \sigma(\mathbf{W}_i[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i) \\
o_t &= \sigma(\mathbf{W}_o[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o) \\
\tilde{c}_t &= \tanh\left(\mathbf{W}_i[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i\right) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\
h_t &= o_t \odot \tanh c_t
\end{aligned} \tag{6}$$

In the scope of this paper, we aim to use LSTM networks to make many-to-many predictions for time series of scalar variables. Therefore, we provide the first few time steps to the LSTM network and it estimates the rest of the time steps. The main reason that we employ a many-to-many prediction framework is that the one-to-one scheme is not promising; besides, we experienced that the one-to-many method acquires a divergent characteristic by exponentially raising the error of each time step prediction. We could overcome this issue with a multi-time step input which bounds the error along the sequence. At this point, the sequence length, N_{sq}, that an input possesses comes into prominence and we

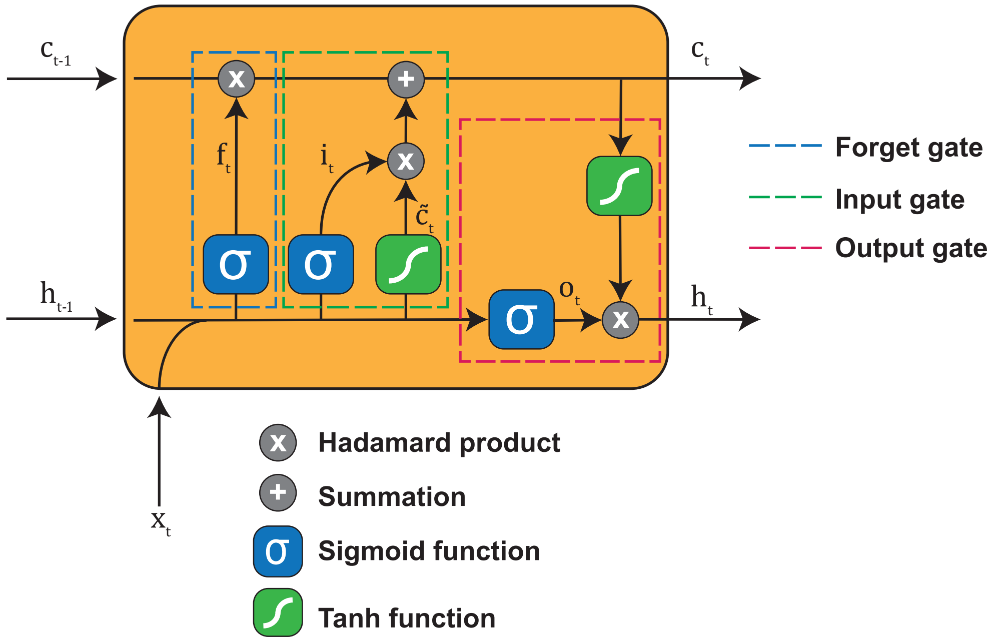
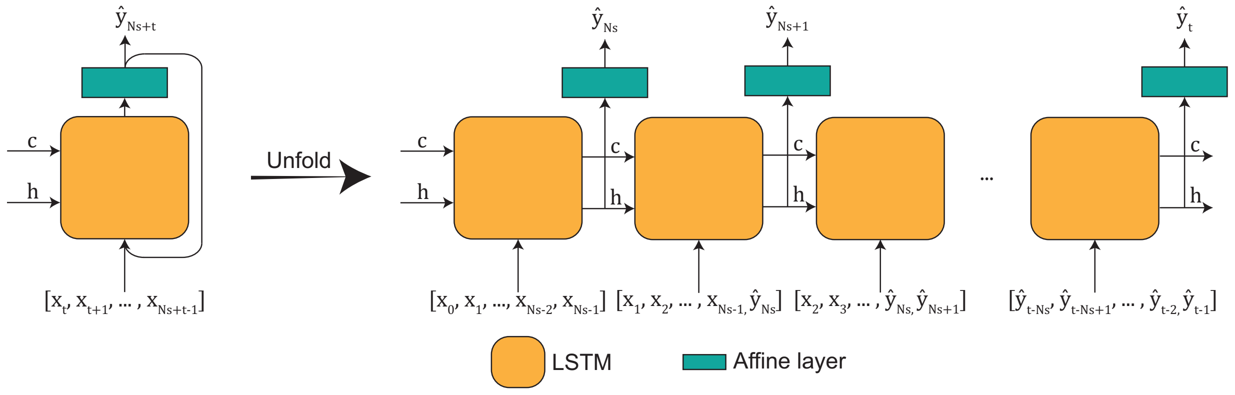


Fig. 2 An LSTM cell.

consider it as a hyperparameter that should be tuned to enhance the network prediction performance. The network architecture is given in Fig. 3. In this figure, the unfolded architecture accepts multi-time step input and produces a single-time step prediction in an iteration step. An affine layer is concatenated to the LSTM cell to linearly transform the hidden state to the prediction. Just after the prediction is yielded by the network, it is stacked with the previous input and fed again to the network; hence, the network recursively feeds itself.

D. Unsupervised Learning

Neural networks used in time series prediction are trained in a way that both the input and ground target belonge to a single dataset. This is referred to unsupervised learning in the literature [22, 23]. The training part of a many-to-many LSTM network may be quite tricky due to the tendency of the network to overfit, explode or vanish gradients even if it utilizes LSTM architecture, and a trivial optimization contingent upon a loss function and the network architecture. Different and efficacious strategies have been developed to cope with these hindrances. One approach to ease the training



$\hat{y}_{Ns+t}$
c
h
$[x_t, x_{t+1}, \dots, x_{Ns+t-1}]$
Unfold
$\hat{y}_{Ns}$
$\hat{y}_{Ns+1}$
$\hat{y}_t$
$[x_0, x_1, \dots, x_{Ns-2}, x_{Ns-1}]$
$[x_1, x_2, \dots, x_{Ns-1}, \hat{y}_{Ns}]$
$[x_2, x_3, \dots, \hat{y}_{Ns}, \hat{y}_{Ns+1}]$
$[\hat{y}_{t-Ns}, \hat{y}_{t-Ns+1}, \dots, \hat{y}_{t-2}, \hat{y}_{t-1}]$
...
LSTM
Affine layer

Fig. 3 Many-to-many LSTM network.

is to normalize or scale the data. Since LSTM uses gates with sigmoid function, a saturation of activation function which arises from high positive or low negative values may slow down the gradient-based learning [24]. Therefore, our network is trained with scaled fluctuations, $x_s^{\prime t}$, around the time average, μ, as in [22, 23] where each scalar datum satisfies $x_s^{\prime t} \in [0, 1] \subset \mathbb{R}$.

$$\begin{aligned} \mathbf{X}^{'} &= \{x_t^{'} = x^t - \mu \mid t = 1, 2, \dots N_{S_X}\} \\ x_{t,s}^{'} &= \frac{x_t^{'} - X_{\text{min}}^{'}}{X_{\text{max}}^{'} - X_{\text{min}}^{'}} \end{aligned} \tag{7}$$

Equation (7) represents the relations used in data scaling. Here, $\mathbf{x}_t^{'}$ corresponds to fluctuations around the time average at time t and $\mathbf{X}^{'} \in \mathbb{R}^{N_{S_X}}$ is the dataset including only fluctuations. *min* and *max* subscripts respectively emphasize the minimum and the maximum values of the dataset. We implement sliding window algorithm to generate ground truths for a given sequence length. Thus, we arrange input and targets such that $\mathbf{X}_s^{'} \in \mathbf{R}^{(N_{S_X} - N_{sq}) \times N_{sq}}$ and $\mathbf{Y}_s^{'} \in \mathbf{R}^{N_{S_X} - N_{sq}}$. Besides, input and target sets are globally shuffled once at the beginning of training to improve the performance of the learning process [25, 26] and split into 80% − 20% training and test data with a batch size of 32. Moreover, PyTorch -an open-source machine learning framework- [27] and Adam first-order gradient-based optimizer [28] are employed to train the network. All trainings are done with a single NVIDIA Quadro P4000 GPU with 8GB VRAM.

III. Aerodynamic and Acoustics Solution

Time-dependent prediction with the RNN approach was applied to the prediction of aerodynamic properties and sonic boom loudness of a maneuvering low-boom supersonic aircraft. To be able to calculate aerodynamic properties and sonic boom loudness, the flow field around the aircraft must be solved accurately with a high-fidelity solver. In the current study, we employed SU2 open-source multi-physics solver [29] developed by Stanford University ADL. We validated and used SU2 solver in our previous studies [7, 15] in the scope of low-boom supersonic research. Therefore, SU2 is employed in this study as well. Inviscid unsteady supersonic flow solution is used to calculate the near-field flow properties around the aircraft. Governing equations in this case are Euler equations, and are shown in Eq. (8) in the conservative form.

$$\frac{\partial Q}{\partial t} + \nabla \cdot \bar{F}_c(Q) = 0 \tag{8}$$

where Q is the conservative state vector and F_c is the convective flux vector. Q and F_c are given by

$$Q = \left\{ \begin{array}{c} \rho \\ \rho\bar{V} \\ \rho E \end{array} \right\} \quad \bar{F}_c = \left\{ \begin{array}{c} \rho\bar{V} \\ \rho\bar{V} \otimes \bar{V} + \bar{\bar{I}}p \\ \rho E\bar{V} + p\bar{V} \end{array} \right\} \tag{9}$$

In the equation system, ρ, E, p, $\bar{V}$, and $\bar{I}$ represent density, energy per mass, thermodynamic pressure, velocity vector and identity matrix, respectively. The finite volume method with a vertex-based data structure is used by SU2 to discretize equations in the flow domain. Jameson-Schmidt-Turkel's (JST) central scheme is selected for the computation of convective fluxes thanks to the faster convergence rate with the proper artificial dissipation coefficients. The convergence criterion is set to 10^{-10} for the RMS density. The special gradients for convective fluxes are calculated with weighted least squares. Since unstructured meshes can be used in SU2, a mixed-type grid structure is used in the supersonic flow solution for the low-boom aircraft. A two-level V-cycle multigrid method is used to increase the convergence rate of the solution. FGMRES algorithm is selected for the iterative solution of the matrix system.

With the accurate solution of near-field domain around the aircraft, typically 2 or 3 body lengths away, the sonic boom loudness on the ground level can be calculated by utilizing nonlinear acoustic propagation methods. A pressure signature taken on a straight line parallel to the x axis and placed 2 body lengths away from the aircraft with a specific azimuth angle is considered as an initial waveform signal for the wave propagation throughout the atmosphere. The Ray tube method is used for the domain of wave propagation in the subject of geometric acoustics. A ray tube geometry is assumed as a slender horn formed by the four different imaginary rays propagating from the aircraft. In this ray tube geometry domain, augmented Burger's equation [30] is solved numerically by the operator-splitting method. The equation is shown in Eq. (10) in a dimensionless form.

$$\frac{\partial P}{\partial \sigma} = P\frac{\partial P}{\partial \tau} + \frac{1}{\Gamma}\frac{\partial^2 P}{\partial \tau^2} + \sum_{\nu} C_\nu \frac{\frac{\partial^2}{\partial \tau^2}}{1 + \theta_\nu \frac{\partial}{\partial \tau}} P - \frac{\frac{\partial A}{\partial \sigma}}{2A} P + \frac{\frac{\partial(\rho_0 c_0)}{\partial \sigma}}{2\rho_0 c_0} P \tag{10}$$

In this equation, each term on the right side can be expressed as the nonlinear effect, thermoviscous attenuation, molecular relaxation, geometrical spreading, and stratification respectively. In this study, NASA sBOOM code [31] is employed for the wave propagation modeling. After the pressure signature on the ground is obtained, a Fast Fourier Transform can be applied to compute loudness perceived loudness. Stevens' Mark VII algorithm [32] is widely used to calculate perceived loudness from the pressure signature. An illustration of the overall sonic boom loudness prediction is presented in Fig. 4.

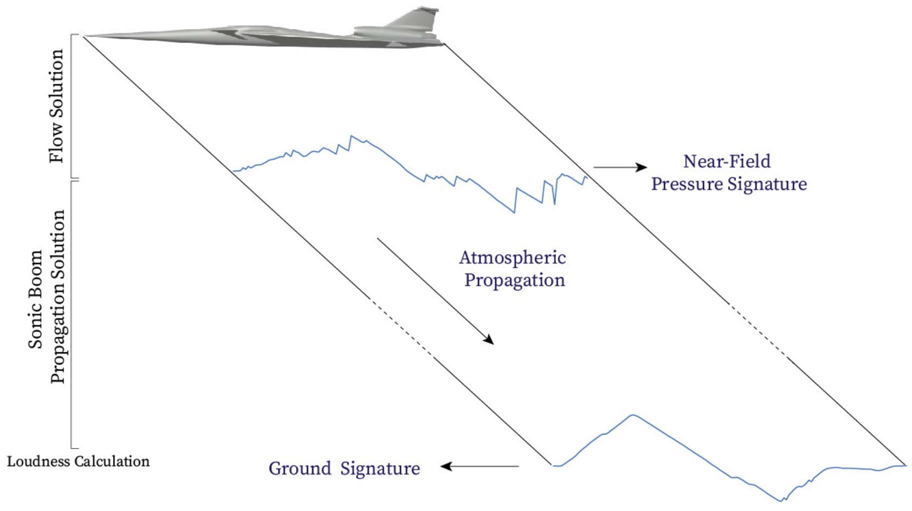


Fig. 4 Sonic loudness calculation process [7]

IV. Application

We present two unsteady aerodynamic applications in this section. The first one is flow field reconstruction of a 2D cylinder and the second one is for unsteady motion of a low-boom supersonic aircraft.

1. Flow Field Reconstruction of a 2D Cylinder

Flow field reconstruction with a convolutional autoencoder of a 2D cylinder in a channel is performed. The channel is discretized into roughly 13000 unstructured elements and unsteady Reynolds-Averaged-Navier-Stokes flow solutions are obtained for 256 time steps at Mach 0.2 and Reynolds number of 5000 with Spallart-Allmaras turbulence model. Since convolutional neural networks leverage locality of adjacent data, we sampled each flow field of all time steps to generate 128×128 structured flow fields and this enabled us to represent each solution with a matrix instead of a vector due to the unstructured grid. We built convolutional autoencoder with total 16 layers to downsample a given input, extract latent vectors, and reconstruct flow fields. The detailed architecture of the autoencoder is provided in Table 1.

Table 1 Convolutional encoder and decoder layers.

Encoder				Decoder			
Layer	Filter	Kernel Size	Stride	Layer	Filter	Kernel Size	Stride
Convolution	2	4×4	2×2	Transpose Convolution	32	2×2	1×1
Convolution	4	4×4	2×2	Transpose Convolution	16	2×2	1×1
Convolution	8	4×4	2×2	Transpose Convolution	8	4×4	2×2
Convolution	16	4×4	2×2	Transpose Convolution	4	4×4	2×2
Convolution	32	2×2	1×1	Transpose Convolution	2	4×4	2×2
Convolution	64	2×2	1×1	Transpose Convolution	1	4×4	2×2

Two affine layers are concatenated after encoder and before decoder part to respectively downsample and upsample a latent vector, $\mathbf{z} \in \mathbb{R}^{64}$. After each convolution layer, batch normalization is applied to improve the generalization capability of the network and speed up the training process [33]. At the end of the network, a sigmoid layer is utilized to map the prediction between 0 and 1. Thus, the total number of learnable parameters of approximately 1.1 million is obtained. The dataset is pre-processed as it is described in II.D. Moreover, we employ Adam optimizer and mean squared error loss function with a learning rate of 10^{-4} and train our autoencoder for 5000 epochs. We achieved 0.0035 mean-squared-error loss with the test data. Randomly selected reconstructed unsteady flow fields from the test dataset are compared with RANS solutions in Fig. 5. In the figure, filled contours show the reconstructed flow field by the autoencoder whereas the contour lines are belonged to RANS solutions. Results demonstrate that latent features that represent a given flow field in a low-dimensional space are extracted and then used to reconstruct the flow field well. Even though the estimated flow around the cylinder provides dominant flow properties, sudden jumps between adjacent data points cause a noisy estimation; hence, there is still room for improvement.

A. NASA C25D Low-Boom Aircraft Case

As explained in the previous sections, the RNN approach is applied for the unsteady motion of a low-boom supersonic aircraft. NASA Concept 25D (C25D) aircraft which was designed to be a low-boom demonstration aircraft [34] is selected and shown in Fig. (6). Since the C25D model was considered as a validation case in the 2nd AIAA Sonic Boom Prediction Workshop (SBPW), high-fidelity solution results with different grids are available in the literature from several research institutes and foundations [35]. Therefore, we conducted a validation study to ensure that convergence of our solution is sufficient.

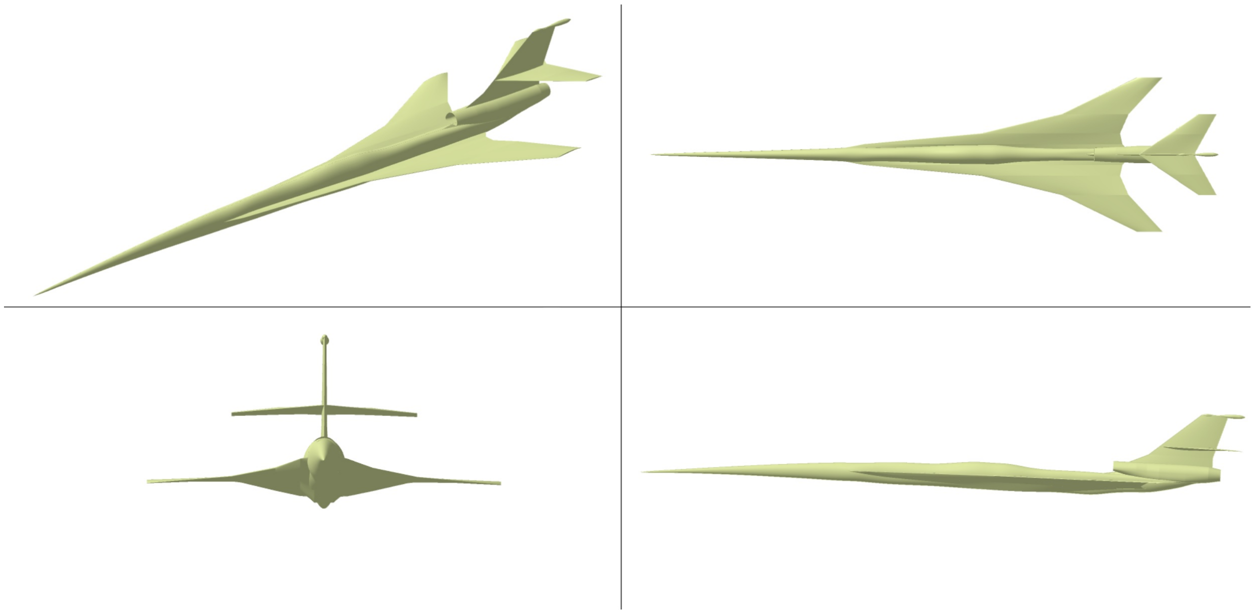


Fig. 6 Illustration of NASA C25D low-boom demonstrator aircraft

Outer-Mold-Line (OML) geometry is taken from the SBPW and a mixed-type grid is generated with 6.5 million unstructured and 12 million structured mesh elements. Then, a steady analysis is performed at 1.6 Mach number and 15760 m. An illustration of pressure coefficient distribution on the symmetry plane is provided in Fig. (7). The steady solution converged with 10^{-10} RMS density in 540 iterations and took 70 minutes on a work station with 36 cores.

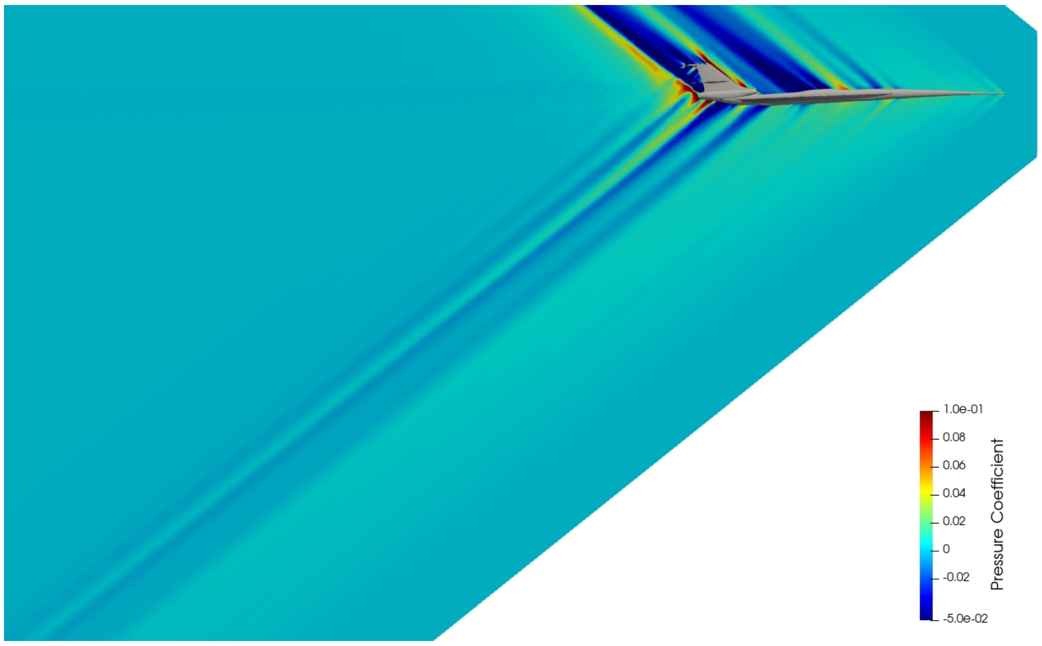
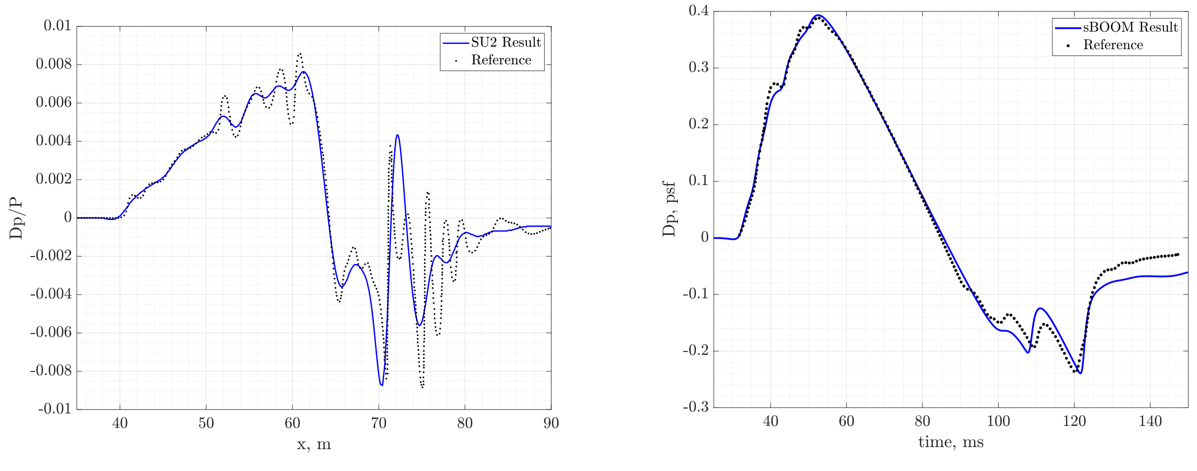


Fig. 7 Illustration of the pressure coefficient distribution on the symmetry plane

The near-field pressure signature is compared with SBPW INRIA results [36] where an adaptive unstructured grid with 26.9 million elements was used. Comparison of the near-field pressure signature taken from 1 body length away from the aircraft is illustrated in Fig. (8a). Then, by using the near-field pressure signature taken from 2.5 body length away from the aircraft was entered to sBOOM code to calculate ground signature which is compared with [35] in Fig (8b). Sonic boom analysis is conducted at 15670 m by using standard atmosphere profiles without wind effect and the perceived loudness is obtained as 78.475 dB.

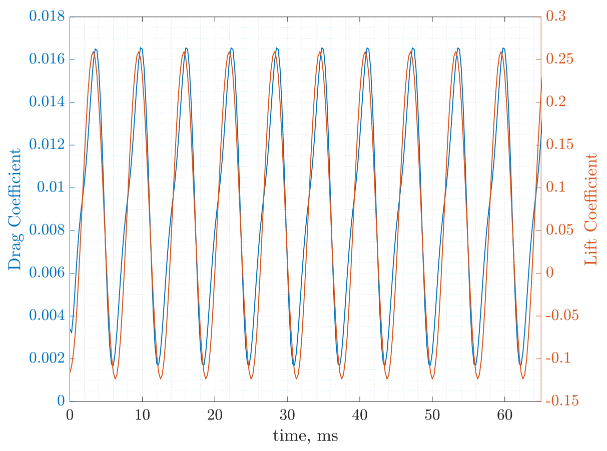


(a) Near-field pressure signatures, $R/L = 1$

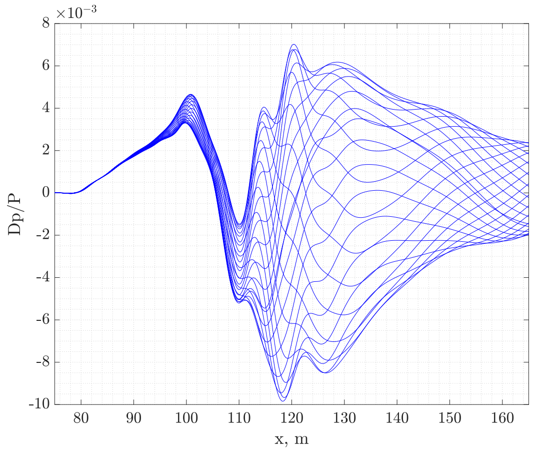
(b) Ground signatures

Fig. 8 Pressure signature comparison for the static case

With the same grid used in the static condition, unsteady analyses are performed in SU2 solver. The time step was set to 0.0025 seconds where the total physical solution is 0.6 seconds. SU2 allows a conducted unsteady periodic motion analysis by utilizing mesh rotation. We performed a pitch-oscillation motion with 106 rad/s angular frequency and 1 degree amplitude. The behaviors of the lift and drag coefficients during the oscillation motion are provided in Fig. (9a) and the near-field pressure signatures are shown in Fig. (9b) with respect to time.



(a) Time-dependent C_L and C_D, $R/L = 1$



(b) Near-field pressure signatures, 25 samples

Fig. 9 Time-dependent aerodynamic coefficients and pressure signatures

B. Time Series Prediction using LSTM Network

We investigate prediction of scalar unsteady aerodynamic and aeroacoustic variables using LSTM as a preliminary study of this ongoing research to gain intuition and insight with shallow recurrent networks. Three different time series are considered with an increasing nonlinearity which are lift coefficient, drag coefficient, and sonic boom loudness of the C25D aircraft. We aim to observe dynamic training characteristics that are closely associated with hyperparameters and capabilities of LSTM networks.

First, we consider prediction of aerodynamic coefficients. The sequence obtained from flow simulations includes 70 snapshots of lift and drag coefficient values, $X \in \mathbb{R}^{70\times2}$, and pre-processed as it is described in Section II.D. We implement a shallow recurrent neural network with concatenated LSTM and affine layers and the total number of learnable parameters is 1794. The hidden size of 16 is found sufficient to capture the time-dependent information. The output of an LSTM layer is flattened and fed into an affine layer where its weight and bias respectively are $\mathbf{W}_a \in \mathbb{R}^{N_h \cdot N_{sq} \times 2}$ and $\mathbf{b}_a \in \mathbb{R}^2$. We experimented 3 different sequence lengths, $N_{sq} = \{4, 8, 16\}$, and trained each case for 10000 epochs with a learning rate of 10^{-5} and L_1 loss function. A high number of epochs and a low learning rate are preferred to smooth out the loss reduction in each epoch and avoid oscillations that may cause overshoot. Our network takes an input sequence of aerodynamic coefficients with a sequence length N_{sq} and estimates the next pair of coefficients. The predictions that are made in each recursive iteration are appended to the input sequence which fed into the network omitting the first element. As a result, 0.0025 mean absolute error is achieved. Predictions of all drag and lift coefficients with respect to the angle of attack are given in Fig. 10. We conclude that even a single LSTM layer is capable of capturing the dynamics of the given unsteady aerodynamic coefficients. However, a set of priori data should be given to enable the network to predict accurately.

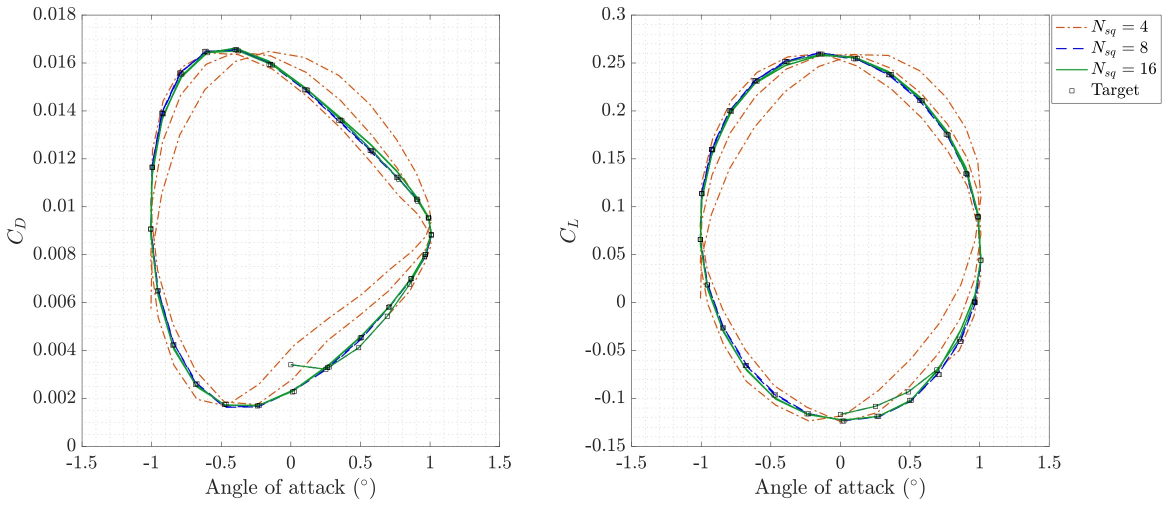


Fig. 10 Aerodynamics coefficient predictions.

The dynamic behavior of aerodynamic coefficients considered in this study is almost harmonic; thus, easier to predict when compared to time-dependent acoustic solutions. Although the obtained acoustic solutions show a harmonic behavior due to the nature of the given maneuver signal, they do not follow a certain path. In addition, this dataset comprises a longer sequence of a scalar variable with a total 155 time steps, $X \in \mathbb{R}^{155}$. Similarly, the same pre-process is applied to the sonic boom loudness dataset. The same LSTM network architecture and training algorithm are used. For this case, inputs with sequence lengths of 8, 16, and 32 are used. The results are shared in Fig. 11. In general, a shallow LSTM network can make time series predictions of sonic boom loudness well with a 0.025 mean absolute error. However, in case of lower sequence lengths, a remarkable phase difference occurs as the time passes. In addition to that none of the experimented networks can accurately predict where the amplitude becomes maximum and minimum even the complexity of the network is increased with additional nonlinear layers. The accuracy of the network at those points may be enhanced with a broader dataset or data augmentation.

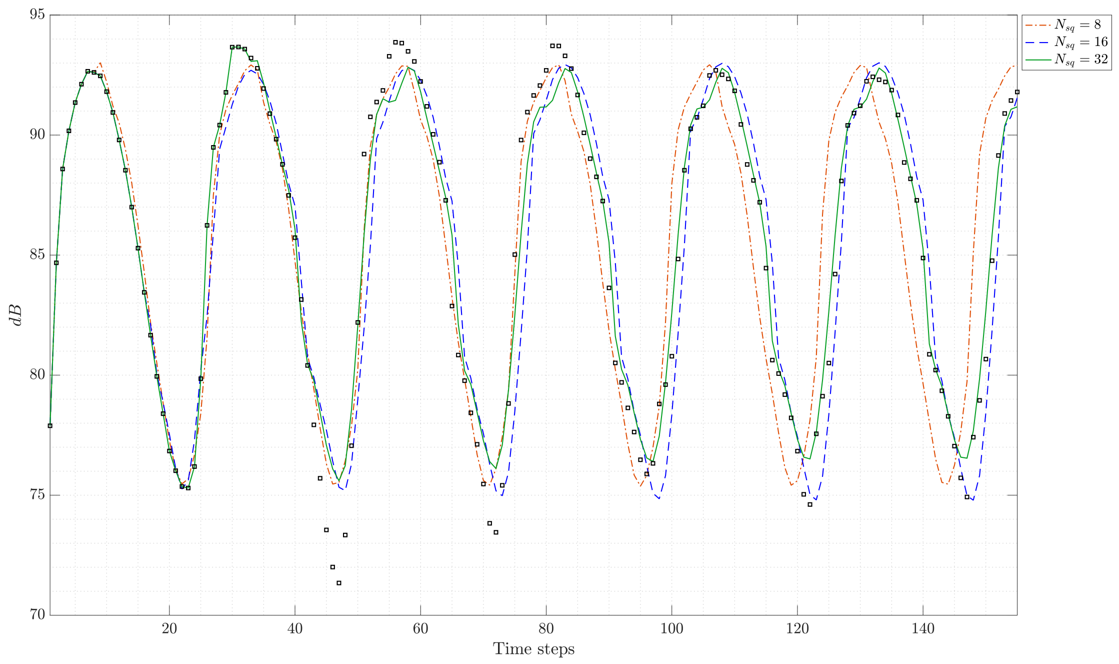


Fig. 11 Sonic boom loudness predictions.

V. Conclusions and Ongoing Work

This ongoing research is divided into three parts: flow field reconstruction with an autoencoder, time-series prediction of scalar variables with recurrent neural networks, and 2D unsteady flow field estimation with convolutional recurrent autoencoders. In this paper, the findings of the first two parts are presented. Firstly, we provide a basic mathematical background of implemented methods. Then, applications of 2D viscous flow field reconstruction of a cylinder in a channel and time-series prediction of aerodynamic and acoustic variables with an LSTM network are presented. In the first application, we build a convolutional autoencoder that has 16 layers and takes a scaled Mach field of a time step to represent it in a low-dimensional space. Then, the flow field is reconstructed by using the compressed representation and transpose convolution layers. In summary, the proposed network can extract the low-dimensional features and reconstruct the entire flow field well. However, improvements may be done in the decoder part in order to eliminate sudden small jumps in adjacent data points which brings about a noisy prediction. Moreover, a 2-layer LSTM network is used to estimate unsteady drag and lift coefficients as well as sonic boom loudness. Inputs with different sequence lengths are given to the network and expected to predict the rest of the time steps. It can be concluded that even a shallow network can learn the dynamics of a given time-series.

In the following work packages of this study, the obtained knowledge will be used to exploit convolutional recurrent autoencoder in single and multifidelity predictions of 2D flow fields. In the autoencoder part, instead of feeding the decoder with a latent vector, a recurrent layer will be placed to march the low-dimensional representation of a given flow field in time. The obtained set of feature vectors will be then upsampled to reconstruct the unsteady flow field.

Acknowledgments

All authors would like to express their gratitude to TUBITAK for the research grant provided under the 218M471 TUBITAK 1001 project titled as "Development of Multifidelity and Multidisciplinary Methodologies Integrating Sonic Boom, Aeroelasticity and Propulsion System for Supersonic Aircraft Design".

Last author would like to thank NASA Langley Research Center for distributing sBOOM sonic boom prediction code internationally for academic research.

References

[1] Burgos, E. R. F., Trani, A., Hinze, N., Marien, T., Geiselhart, K., Dollyhigh, S., and Seidel, J., "Global Demand Model to Estimate Supersonic Commercial Services," *AIAA AVIATION 2021 FORUM*, American Institute of Aeronautics and Astronautics, 2021. https://doi.org/10.2514/6.2021-3187, URL https://doi.org/10.2514/6.2021-3187.

[2] Mane, M., Jain, S., and Crossley, W., "Market Size and Design Requirements for Supersonic Passenger Transport Aircraft," *AIAA AVIATION 2021 FORUM*, American Institute of Aeronautics and Astronautics, 2021. https://doi.org/10.2514/6.2021-2442, URL https://doi.org/10.2514/6.2021-2442.

[3] Farhat, C., Maute, K., Argrow, B., and Nikbay, M., "Shape Optimization Methodology for Reducing the Sonic Boom Initial Pressure Rise," *AIAA Journal*, Vol. 45, No. 5, 2007, pp. 1007–1018. https://doi.org/10.2514/1.27607, URL https://doi.org/10.2514/1.27607.

[4] Maute, K., Farhat, C., Argrow, B., and Nikbay, M., "Sonic boom mitigation via shape optimization using an adjoint method and application to a supersonic fighter aircraft," *European Journal of Computational Mechanics*, Vol. 17, No. 1-2, 2008, pp. 217–243. https://doi.org/10.3166/remn.17.217-243.

[5] Alonso, J. J., and Colonno, M. R., "Multidisciplinary Optimization with Applications to Sonic-Boom Minimization," *Annual Review of Fluid Mechanics*, Vol. 44, No. 1, 2012, pp. 505–526. https://doi.org/10.1146/annurev-fluid-120710-101133, URL https://doi.org/10.1146/annurev-fluid-120710-101133.

[6] Li, W., and Geiselhart, K., "Multidisciplinary design optimization of low-boom supersonic aircraft with mission constraints," *AIAA Journal*, Vol. 59, No. 1, 2021, pp. 165–179. https://doi.org/10.2514/1.J059237.

[7] Demiroglu, Y., Yildiz, S., and Nikbay, M., "Multi-fidelity Sonic Boom Minimization of a Supersonic Aircraft by Parametric Wing Shape Design," *AIAA Scitech 2021 Forum*, American Institute of Aeronautics and Astronautics, 2021. https://doi.org/10.2514/6.2021-1009, URL https://doi.org/10.2514/6.2021-1009.

[8] Liebhardt, B., Lütjens, K., Ueno, A., and Ishikawa, H., "JAXA's S4 supersonic low-boom airliner – a collaborative study on aircraft design, sonic boom simulation, and market prospects," *Aiaa Aviation 2020 Forum*, Vol. 1 PartF, 2020, pp. 6–13. https://doi.org/10.2514/6.2020-2731.

[9] Min, S., Lee, B., and Yoon, S., "Deep Learning in Bioinformatics," , 2016.

[10] Spencer, M., Eickholt, J., and Cheng, J., "A Deep Learning Network Approach to italicab initio/italic Protein Secondary Structure Prediction," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, Vol. 12, No. 1, 2015, pp. 103–112. https://doi.org/10.1109/tcbb.2014.2343960, URL https://doi.org/10.1109/tcbb.2014.2343960.

[11] Cao, R., Bhattacharya, D., Hou, J., and Cheng, J., "DeepQA: improving the estimation of single protein model quality with deep belief networks," *BMC Bioinformatics*, Vol. 17, No. 1, 2016. https://doi.org/10.1186/s12859-016-1405-y, URL https://doi.org/10.1186/s12859-016-1405-y.

[12] Toms, B. A., Barnes, E. A., and Ebert-Uphoff, I., "Physically Interpretable Neural Networks for the Geosciences: Applications to Earth System Variability," *Journal of Advances in Modeling Earth Systems*, Vol. 12, No. 9, 2020. https://doi.org/10.1029/2019ms002002, URL https://doi.org/10.1029/2019ms002002.

[13] Mamalakis, A., Ebert-Uphoff, I., and Barnes, E. A., "Neural Network Attribution Methods for Problems in Geoscience: A Novel Synthetic Benchmark Dataset," , 2021.

[14] Lary, D. J., Alavi, A. H., Gandomi, A. H., and Walker, A. L., "Machine learning in geosciences and remote sensing," *Geoscience Frontiers*, Vol. 7, No. 1, 2016, pp. 3–10. https://doi.org/10.1016/j.gsf.2015.07.003, URL https://doi.org/10.1016/j.gsf.2015.07.003.

[15] Tekaslan, H. E., Imrak, R., and Nikbay, M., "Reliability Based Design Optimization of a Supersonic Engine Inlet," *AIAA Propulsion and Energy 2021 Forum*, American Institute of Aeronautics and Astronautics, 2021. https://doi.org/10.2514/6.2021-3541, URL https://doi.org/10.2514/6.2021-3541.

[16] Julian, K. D., and Kochenderfer, M. J., "Reachability Analysis for Neural Network Aircraft Collision Avoidance Systems," *Journal of Guidance, Control, and Dynamics*, Vol. 44, No. 6, 2021, pp. 1132–1142. https://doi.org/10.2514/1.g005233, URL https://doi.org/10.2514/1.g005233.

[17] Yu, J., and Hesthaven, J. S., "Flowfield Reconstruction Method Using Artificial Neural Network," *AIAA Journal*, Vol. 57, No. 2, 2019, pp. 482–498. https://doi.org/10.2514/1.j057108, URL https://doi.org/10.2514/1.j057108.

[18] Hu, L., Zhang, J., Xiang, Y., and Wang, W., "Neural Networks-Based Aerodynamic Data Modeling: A Comprehensive Review," *IEEE Access*, Vol. 8, 2020, pp. 90805–90823. https://doi.org/10.1109/ACCESS.2020.2993562.

[19] Werbos, P., "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, Vol. 78, No. 10, 1990, pp. 1550–1560. https://doi.org/10.1109/5.58337, URL https://doi.org/10.1109/5.58337.

[20] Pascanu, R., Mikolov, T., and Bengio, Y., "On the difficulty of training Recurrent Neural Networks," , 2012.

[21] Hochreiter, S., and Schmidhuber, J., "Long Short-Term Memory," Vol. 9, No. 8, 1997, pp. 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735, URL https://doi.org/10.1162/neco.1997.9.8.1735.

[22] Gonzalez, F. J., and Balajewicz, M., "Deep convolutional recurrent autoencoders for learning low-dimensional feature dynamics of fluid systems," , 2018.

[23] Bukka, S. R., Magee, A. R., and Jaiman, R. K., "Deep Convolutional Recurrent Autoencoders for Flow Field Prediction," , 2020.

[24] Goodfellow, I., Bengio, Y., and Courville, A., *Deep Learning*, MIT Press, 2016. http://www.deeplearningbook.org.

[25] Chung, J., "UberShuffle: Communication-efficient Data Shuffling for SGD via Coding Theory," 2017.

[26] Gürbüzbalaban, M., Ozdaglar, A., and Parrilo, P. A., "Why random reshuffling beats stochastic gradient descent," *Mathematical Programming*, Vol. 186, No. 1-2, 2019, pp. 49–84. https://doi.org/10.1007/s10107-019-01440-w, URL https://doi.org/10.1007/s10107-019-01440-w.

[27] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S., "PyTorch: An Imperative Style, High-Performance Deep Learning Library," *Advances in Neural Information Processing Systems 32*, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Curran Associates, Inc., 2019, pp. 8024–8035. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[28] Kingma, D. P., and Ba, J., "Adam: A Method for Stochastic Optimization," *ICLR (Poster)*, 2015. URL http://arxiv.org/abs/1412.6980.

[29] Economon, T. D., Palacios, F., Copeland, S. R., Lukaczyk, T. W., and Alonso, J. J., "SU2: An Open-Source Suite for Multiphysics Simulation and Design," *AIAA Journal*, Vol. 54, No. 3, 2016, pp. 828–846. https://doi.org/10.2514/1.j053813, URL https://doi.org/10.2514/1.j053813.

[30] CLEVELAND, R. O., "PROPAGATION OF SONIC BOOMS THROUGH A REAL, STRATIFIED ATMOSPHERE," Ph.D. thesis, THE UNIVERSITY OF TEXAS AT AUSTIN, 1995.

[31] Rallabhandi, S. K., "Advanced Sonic Boom Prediction Using the Augmented Burgers Equation," *Journal of Aircraft*, Vol. 48, No. 4, 2011, pp. 1245–1253. https://doi.org/10.2514/1.C031248.

[32] Stevens, S. S., "Perceived Level of Noise by Mark VII and Decibels (E)," *The Journal of the Acoustical Society of America*, Vol. 51, No. 2B, 1972, pp. 575–601. https://doi.org/10.1121/1.1912880.

[33] Ioffe, S., and Szegedy, C., "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," , 2015.

[34] Ordaz, I., Wintzer, M., and Rallabhandi, S. K., "Full-Carpet Design of a Low-Boom Demonstrator Concept," *33rd AIAA Applied Aerodynamics Conference*, American Institute of Aeronautics and Astronautics, 2015. https://doi.org/10.2514/6.2015-2261, URL https://doi.org/10.2514/6.2015-2261.

[35] Park, M. A., and Nemec, M., "Nearfield Summary and Statistical Analysis of the Second AIAA Sonic Boom Prediction Workshop," *Journal of Aircraft*, Vol. 56, No. 3, 2019, pp. 851–875. https://doi.org/10.2514/1.c034866, URL https://doi.org/10.2514/1.c034866.

[36] Loseille, A., Frazza, L., and Alauzet, F., "INRIA-GAMMA3 Contribution 2nd AIAA Sonic-Boom workshop," , 2017. URL https://lbpw-ftp.larc.nasa.gov/sbpw2/workshop/sbpw2-talks-nearfield/12-sbpw2-inria-loseille-frazza-alauzet.pdf.

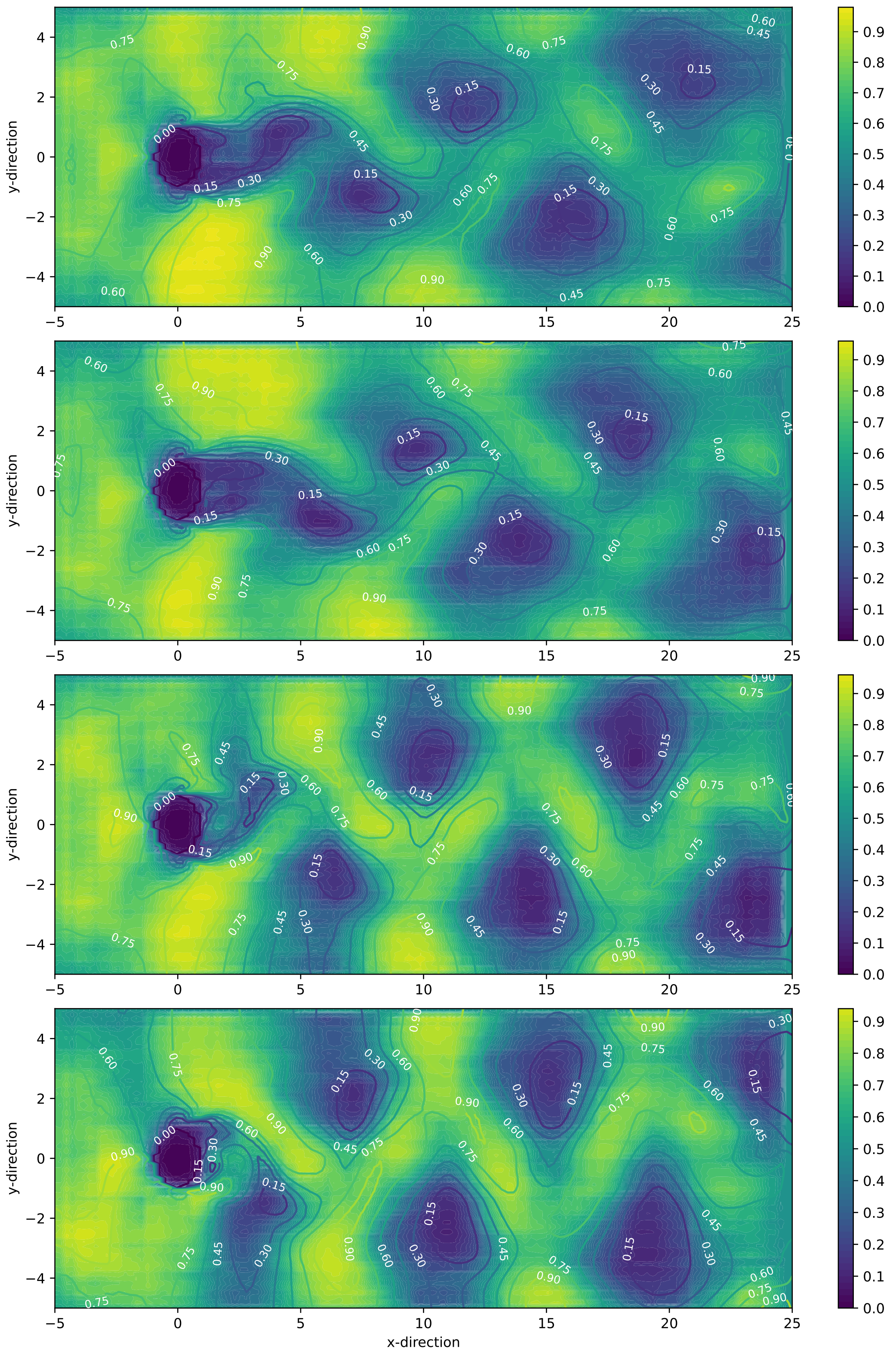


Fig. 5 Comparison of randomly selected reconstructed unsteady scaled Mach fields from the test dataset (filled contour) with RANS solutions (contour lines).