Data-driven prediction of unsteady flow over a circular cylinder using deep learning

Sangseung Lee[1] and Donghyun You[1],†

[1]Department of Mechanical Engineering, Pohang University of Science and Technology,
77 Cheongam-ro, Nam-gu, Pohang, Gyeongbuk 37673, Republic of Korea

Unsteady flow fields over a circular cylinder are used for training and then prediction using four different deep learning networks: generative adversarial networks with and without consideration of conservation laws; and convolutional neural networks with and without consideration of conservation laws. Flow fields at future occasions are predicted based on information on flow fields at previous occasions. Predictions of deep learning networks are made for flow fields at Reynolds numbers that were not used during training. Physical loss functions are proposed to explicitly provide information on conservation of mass and momentum to deep learning networks. An adversarial training is applied to extract features of flow dynamics in an unsupervised manner. Effects of the proposed physical loss functions and adversarial training on predicted results are analysed. Captured and missed flow physics from predictions are also analysed. Predicted flow fields using deep learning networks are in good agreement with flow fields computed by numerical simulations.

Key words: vortex shedding, computational methods

1. Introduction

Observation of fluid flow in nature, laboratory experiments and numerical simulations has provided evidence of the existence of flow features and certain, but often complex, rules. For example, in nature, Kelvin–Helmholtz waves in clouds (Dalin *et al.* 2010), von Kármán vortices in ocean flow around an island (Berger & Wille 1972) and the swirling great red spot on Jupiter (Marcus 1988) are flow structures that can be classified as a certain type of vortical motion produced by a distinct combination of boundary conditions and initial conditions for the governing first principles. Similar observations have also been reported in laboratory experiments and numerical simulations (Freymuth 1966; Ruderich & Fernholz 1986; Babucke, Kloker & Rist 2008; Wu & Moin 2009). The existence of distinct and dominant flow features has also been widely investigated by reduced-order models (ROMs) using mathematical decomposition techniques such as the proper orthogonal decomposition (POD) method (Sirovich 1987), the dynamic mode decomposition (DMD) method (Schmid 2010) and the Koopman operator method (Mezić 2013; Morton *et al.* 2018).

† Email address for correspondence: dhyou@postech.ac.kr

Owing to the existence of distinct or dominant flow features, animals such as insects, birds and fish are reported to be able to control their body movements in order to adapt their fluid dynamic environment and so improve their aero- or hydrodynamic performance and efficiency (Wu 2011; Yonehara *et al.* 2016). This suggests the possibility that they empirically learn to generate dominant fluid motions as well as the nonlinear correlation of fluid motions and are able to estimate future flow based on flow experienced in their environments. Such observations in nature motivate us to investigate the feasibility of predicting unsteady fluid motions by learning flow features using neural networks.

Attempts to apply neural networks to problems of fluid flow have been recently made by Tracey, Duraisamy & Alonso (2015), Zhang & Duraisamy (2015) and Singh, Medida & Duraisamy (2017), who utilized shallow neural networks for turbulence modelling for Reynolds-averaged Navier–Stokes (RANS) simulations. Ling, Kurzawski & Templeton (2016) employed deep neural networks to better model the Reynolds stress anisotropy tensor for RANS simulations. Guo, Li & Iorio (2016) employed a convolutional neural network (CNN) to predict steady flow fields around bluff objects and reported reasonable prediction of steady flow fields with significantly reduced computational cost than that required for numerical simulations. Similarly, Miyanawala & Jaiman (2017, 2018) and Mao *et al.* (2018) employed CNNs to predict aerodynamic forces on bluff bodies, also with notably reduced computational costs. Those previous studies showed the high potential of deep learning techniques for enhancing simulation accuracy and reducing computational cost.

Predicting unsteady flow fields using deep learning involves extracting both spatial and temporal features of input flow field data, which could be considered to be learning videos. Video modelling enables prediction of a future frame of a video based on information from previous video frames by learning the spatial and temporal features of the video. Although deep learning techniques have been reported to generate high-quality real-world-like images in image modelling areas (Denton, Chintala & Fergus 2015; Radford, Metz & Chintala 2015; van den Oord *et al.* 2016a; van den Oord, Kalchbrenner & Kavukcuoglu 2016b), it is known that, for video modelling, deep learning techniques have presented difficulties in generating high-quality prediction due to blurriness caused by complexity in the spatial and temporal features in a video (Ranzato *et al.* 2014; Mathieu, Couprie & LeCun 2015; Srivastava, Mansimov & Salakhudinov 2015; Xingjian *et al.* 2015).

Mathieu *et al.* (2015) proposed a video modelling architecture that utilizes a generative adversarial network (GAN) (Goodfellow *et al.* 2014), which combines a fully convolutional generator model and a discriminator model. The GAN was capable of generating future video frames from input frames at previous times. The generator model generates images and the discriminator model is employed to discriminate the generated images from real (ground truth) images. A GAN is adversarially trained so the generator network is trained to fool the discriminator network, and the discriminator network is trained not to be fooled by the generator network. The Nash equilibrium in the two-pronged adversarial training leads the network to extract underlying low-dimensional features in an unsupervised manner and, in consequence, good quality images can be generated. The most notable advantage of using the GAN is that, once it is trained, the network is able to generate predictions in a larger domain. This leads to a memory-efficient training of videos because the network can predict a frame with a larger size than that in training. A recurrent neural network (RNN) based architecture lends itself to learning the temporal correlation among encoded information in the past and thereby predicting future frames. It is also worth

noting that, in the present study, the application of RNNs proposed by Srivastava *et al.* (2015) and by Xingjian *et al.* (2015) has been attempted. However, it has been found that the methods are practical only for low-resolution frames since the number of weight parameters for the RNNs increases as a function of the square of the resolution of a frame. Ranzato *et al.* (2014) proposed a recurrent convolutional neural network (rCNN), which is also able to predict a frame with a larger size than that in training. However, Mathieu *et al.* (2015) reported that the GAN improves the capability for predicting future frames on a video dataset of human actions (Soomro, Zamir & Shah 2012) compared to the rCNN, the predictions of which are more static for unsteady motions.

Prediction of unsteady flow fields using deep learning could offer new opportunities for real-time control and guidance of aero- or hydro-vehicles, fast weather forecasting, etc. As the first step towards prediction of unsteady flow fields using deep learning, the present study is an attempt to predict rather simple but canonical unsteady vortex shedding over a circular cylinder using four different deep learning networks: GANs with and without consideration of conservation laws and CNNs with and without consideration of conservation laws. Consideration of conservation laws is realized as a form of loss function. The aim of the present study is to predict unsteady flow fields at Reynolds numbers that were not utilized in the learning process. This differs from the aim of ROMs, which is to discover and understand low-dimensional representation of flow fields at certain Reynolds numbers by learning them (Liberge & Hamdouni 2010; Bagheri 2013).

The paper is organized as follows: the method for constructing flow field datasets and deep learning methods are explained in §§ 2 and 3, respectively. The results obtained using the present deep learning networks are discussed in § 4, followed by concluding remarks in § 5.

2. Construction of flow field datasets

2.1. *Numerical simulations*

Numerical simulations of flow over a circular cylinder at Reynolds number $Re_D = U_\infty D/\nu = 150$, 300, 400, 500, 1000, 3000 and 3900, where U_∞, D and ν are the free-stream velocity, cylinder diameter and kinematic viscosity, respectively, are conducted by solving the incompressible Navier–Stokes equations as follows:

$$\frac{\partial u_i}{\partial t} + \frac{\partial u_i u_j}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu\frac{\partial^2 u_i}{\partial x_j \partial x_j} \tag{2.1}$$

and

$$\frac{\partial u_i}{\partial x_i} = 0, \tag{2.2}$$

where u_i, p and ρ are the velocity, pressure and density, respectively. Velocity components and the pressure are non-dimensionalized by U_∞ and ρU_∞^2, respectively. A fully implicit fractional-step method is employed for time integration, where all terms in the Navier–Stokes equations are integrated using the Crank–Nicolson method. Second-order central-difference schemes are employed for spatial discretization and the kinetic energy is conserved by treating face variables as arithmetic means of neighbouring cells (You, Ham & Moin 2008). The computational domain consists
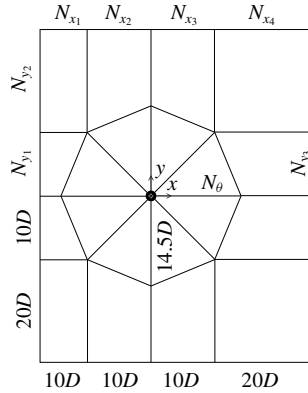
FIGURE 1. The computational domain for numerical simulations. N denotes the number of mesh points, where $N_{x_1} = 20$, $N_{x_2} = 30$, $N_{x_3} = 50$, $N_{x_4} = 50$ $N_{y_1} = 30$, $N_{y_2} = 30$, $N_{y_3} = 80$ and $N_\theta = 150$. The domain size and the number of mesh points in the spanwise direction are $6D$ (πD for flow at $Re_D \geqslant 1000$) and 96, respectively.

of a block structured H-grid with an O-grid around the cylinder (figure 1). The computational domain sizes are $50D$ and $60D$ in the streamwise and the cross-flow directions, respectively, where D is the cylinder diameter. In the spanwise direction, $6D$ is used for flow at Reynolds numbers less than 1000, while πD is used otherwise. The computational time-step size $\Delta t U_\infty / D$ of 0.005 is used for all simulations. The domain size, number of grid points and time-step sizes are determined from an extensive sensitivity study.

2.2. *Datasets*

Flow fields in different vortex shedding regimes are calculated for training and testing deep learning networks. The following flow regimes and Reynolds numbers are considered: two-dimensional vortex shedding regime ($Re_D = 150$), three-dimensional wake transition regime ($Re_D = 300$, 400 and 500) and shear-layer transition regime ($Re_D = 1000$, 3000 and 3900). Simulation results of flow over a cylinder at each Reynolds number are collected with a time-step interval of $\delta t = 20 \Delta t U_\infty / D = 0.1$. Flow variables $u_1/U_\infty (= u/U_\infty)$, $u_2/U_\infty (= v/U_\infty)$, $u_3/U_\infty (= w/U_\infty)$ and $p/\rho U_\infty^2$ at each time step in a square domain of $-1.5D < x < 5.5D$, $-3.5D < y < 3.5D$, $z = 0D$ ($7D \times 7D$ sized domain) are interpolated into a uniform grid with 250×250 cells for all Reynolds number cases. Thus, a dataset at each Reynolds number consists of flow fields with the size of 250×250 (grid cells) $\times 4$ (flow variables).

The calculated datasets of flow fields are divided into training and test datasets, so that flow fields at Reynolds numbers in the training dataset is not included in the test dataset. Flow fields in the training dataset are randomly subsampled in time and space into five consecutive flow fields on a $0.896D \times 0.896D$ domain with 32×32 grid cells (see figure 2). The subsampled flow fields contain diverse types of flow, such as free-stream flow, wake flow, boundary layer flow or separating flow. Therefore, deep learning networks are allowed to learn diverse types of flow. The first four consecutive sets of flow fields are used as an input ($\mathcal{I}$), while the following set of flow fields is a ground truth flow field ($\mathcal{G}(\mathcal{I})$). The pair of input and ground truth flow fields form a training sample. In the present study, a total of 500 000 training samples are
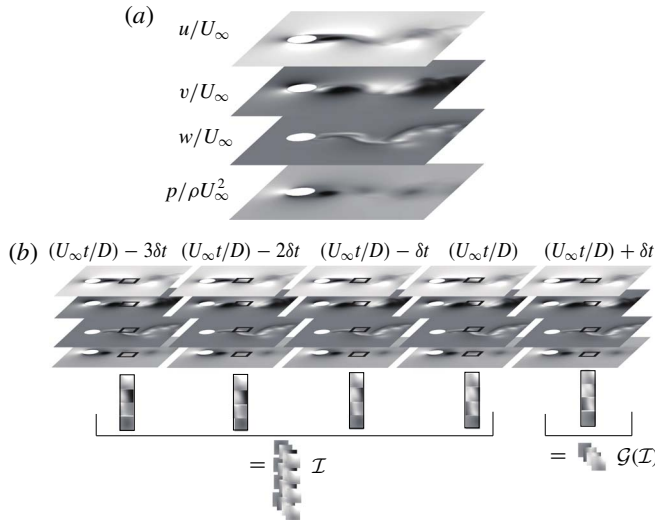
FIGURE 2. (*a*) Instantaneous fields of flow variables u/U_∞, v/U_∞, w/U_∞ and $p/\rho U_\infty^2$ on a $7D \times 7D$ domain with 250×250 grid cells. (*b*) The procedure of subsampling five consecutive flow fields to the input ($\mathcal{I}$) and the ground truth ($\mathcal{G}(\mathcal{I})$) on a $0.896D \times 0.896D$ domain with 32×32 grid cells.

employed for training deep learning networks. The predictive performance of networks is evaluated on a test dataset, which is composed of interpolated flow fields from numerical simulations on a $7D \times 7D$ domain with 250×250 grid cells.

3. Deep learning methodology

3.1. *Overall procedure of deep learning*

A deep learning network learns a nonlinear mapping of an input tensor and an output tensor. The nonlinear mapping is composed of a sequence of tensor operations and nonlinear activations of weight parameters. The objective of deep learning is to learn appropriate weight parameters that form the most accurate nonlinear mapping of the input tensor and the output tensor that minimizes a loss function. A loss function evaluates the difference between the estimated output tensor and the ground truth output tensor (the desired output tensor). Therefore, deep learning is an optimization procedure for determining weight parameters that minimize a loss function. A deep learning network is trained with the following steps.

(1) A network estimates an output tensor from a given input through the current state of weight parameters, which is known as feed forward.
(2) A loss (scalar value) is evaluated by a loss function of the difference between the estimated output tensor and the ground truth output tensor.
(3) Gradients of the loss with respect to each weight parameter are calculated through the chain rule of partial derivatives starting from the output tensor, which is known as back propagation.
(4) The weight parameters are gradually updated in the negative direction of the gradients of the loss with respect to each weight parameter.
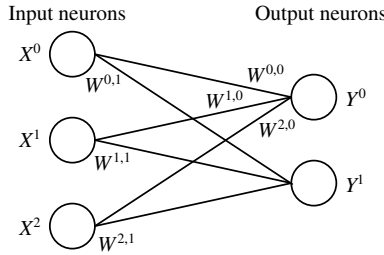
FIGURE 3. Illustration of a fully connected layer.

(5) Steps 1 to 4 are repeated until weight parameters (deep learning network) are sufficiently updated.

The present study utilizes two different layers that contain weight parameters: fully connected layers and convolution layers. An illustration of a fully connected layer is shown in figure 3. Weight parameters of a fully connected layer are stored in connections (W) between layers of input (X) and output (Y) neurons, where neurons are elementary units in a fully connected layer. Information inside input neurons is passed to output neurons through a matrix multiplication of the weight parameter matrix and the vector of input neurons as follows:

$$Y^i = \sum_j W^{j,i} X^j + \text{bias}, \tag{3.1}$$

where a bias is a constant, which is also a parameter to be learned. An output neuron of a fully connected layer collects information from all input neurons with respective weight parameters. This provides strength to learn a complex mapping of input and output neurons. However, as the number of weight parameters is determined as the multiplication of the number of input and output neurons, where the number of neurons is generally in the order of hundreds or thousands, the number of weight parameters easily becomes more than sufficient. As a result, abundant use of fully connected layers leads to inefficient learning. For this reason, fully connected layers are typically used as a classifier, which collects information and classifies labels, after extracting features using convolution layers.

An illustration of a convolution layer is shown in figure 4. Weight parameters (W) of a convolution layer are stored in kernels between input (X) and output (Y) feature maps, where feature maps are elementary units in a convolution layer. To maintain the shape of the input after convolution operations, zeros are padded around input feature maps. The convolution operation with padding is applied to input feature maps using kernels as follows:

$$Y_{i,j}^n = \left[\sum_k \sum_{c=0}^{F_y-1} \sum_{r=0}^{F_x-1} W_{r,c}^{n,k} \underbrace{X_{i+r,j+c}^k}_{\text{Pad included}} \right] + \text{bias}, \tag{3.2}$$

where $F_x \times F_y$ is the size of kernels. Weight parameters inside kernels are updated to extract important spatial features inside input feature maps, so an output feature map contains an encoded feature from input feature maps. Updates of weight parameters

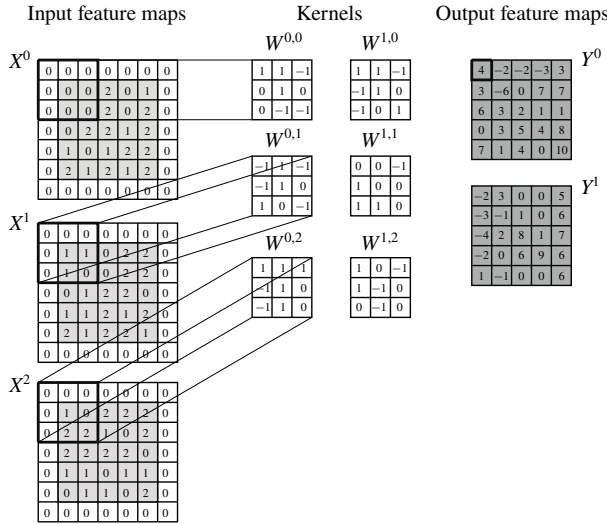Input feature maps   Kernels   Output feature maps

FIGURE 4. Illustration of a convolution layer.

could be affected by padding as output values near boundaries of an output feature map are calculated using parts of weight parameters of kernels, whereas values far from boundaries are calculated using all weight parameters of kernels. However, without padding, the output shape of a feature map of a convolution layer is reduced, which indicates loss of information. Therefore, padding enables a CNN to minimize the loss of information and to be deep by maintaining the shape of feature maps, but as a trade-off it could affect updates of weight parameters.

Convolution layers contain significantly fewer parameters to update, compared to fully connected layers, which enables efficient learning. Therefore, convolution layers are typically used for feature extraction.

After each fully connected layer or convolution layer, a nonlinear activation function is usually applied to the output neurons or feature maps to provide nonlinearity to a deep learning network. The hyperbolic tangent function ($f(x) = \tanh(x)$), the sigmoid function ($f(x) = 1/(1 + \exp(-x))$) and the rectified linear unit (ReLU) activation function ($f(x) = \max(0, x)$) (Krizhevsky, Sutskever & Hinton 2012) are examples of typically applied activation functions. In the present study, these three functions are employed as activation functions (see § 3.2 for details).

A max pooling layer is also utilized in the present study, which does not contain weight parameters but applies a max filter to non-overlapping subregions of a feature map (see figure 5). A max pooling layer can be connected to an output feature map of a convolution layer to extract important features.

3.2. *Configurations of deep learning networks*

Deep learning networks employed in the present study consist of a generator model that accepts four consecutive sets of flow fields as an input. Each input set of flow fields is composed of flow variables of $\{u/U_\infty, v/U_\infty, w/U_\infty, p/\rho U_\infty^2\}$, to take advantage of learning correlated physical phenomena among flow variables. The number of consecutive input flow fields is determined by a parameter study. A high

Input feature map Output feature map
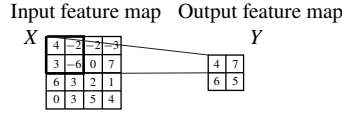
FIGURE 5. Illustration of a 2×2 max pooling layer.

Generator model $(G_3 \rightarrow G_2 \rightarrow G_1 \rightarrow G_0)$

Generative CNN	Numbers of feature maps	Kernel sizes
G_3	16, 128, 256, 128, 4	3×3, 3×3, 3×3, 3×3
G_2	20, 128, 256, 128, 4	5×5, 3×3, 3×3, 5×5
G_1	20, 128, 256, 512, 256, 128, 4	5×5, 3×3, 3×3, 3×3, 3×3, 5×5
G_0	20, 128, 256, 512, 256, 128, 4	7×7, 5×5, 5×5, 5×5, 5×5, 7×7

TABLE 1. Configuration of the generator model in GANs and multi-scale CNNs (see figure 6 for connections).

number of input flow fields increases memory usage and therefore the learning time. A low number might cause a shortage of input information for the networks. Three cases with $m = 2$, 4 and 6 are trained and tested for unsteady flow fields. No significant benefit in the prediction is found with m beyond 4. The flow variables are scaled using a linear function to guarantee that all values are in -1 to 1. This scaling supports the usage of the ReLU activation function by providing nonlinearity to networks and the hyperbolic tangent activation function by bounding predicted values. Original values of the flow variables are retrieved by an inverse of the linear scaling. The generator model utilized in this study is composed of a set of multi-scale generative CNNs $\{G_0, G_1, G_2, G_3\}$ to learn multi-range spatial dependences of flow structures (see table 1 and figure 6). Details of the study for determining network parameters such as numbers of layers and feature maps are summarized in § C.1.

During training, a generative CNN G_k generates flow field predictions $(G_k(\mathcal{I}))$ on the $0.896D \times 0.896D$ domain with resolution of $32/2^k \times 32/2^k$ through padded convolution layers. G_k is fed with four consecutive sets of flow fields on the domain with $32/2^k \times 32/2^k$ resolution $(\mathcal{I}_k)$, which are bilinearly interpolated from the original input sets of flow fields with 32×32 resolution $(\mathcal{I})$, and a set of upscaled flow fields, which is obtained by $R_{k+1} \circ G_{k+1}(\mathcal{I})$ (see figure 6). $R_{k+1} \circ ()$ is an upscale operator that bilinearly interpolates a flow field on a domain with resolution of $32/2^{k+1} \times 32/2^{k+1}$ to a domain with resolution of $32/2^k \times 32/2^k$. Note that domain sizes for $32/2^k \times 32/2^k$ and 32×32 resolution are identical to $0.896D \times 0.896D$, where the size of the corresponding convolution kernel ranges from 3 to 7 (see table 1). Consequently, G_k is able to learn larger spatial dependences of flow fields than G_{k-1} by sacrificing resolution. As a result, a multi-scale CNN-based generator model enables the learning and prediction of flow fields with multi-scale flow phenomena. The last layer of feature maps in each multi-scale CNN is activated with the hyperbolic tangent function to bound the output values, while other feature maps are activated with the ReLU function to provide nonlinearity to networks.

Let $\mathcal{G}_k(\mathcal{I})$ be ground truth flow fields with resized resolution of $32/2^k \times 32/2^k$. The discriminator model consists of a set of discriminative networks $\{D_0, D_1, D_2, D_3\}$ with convolution layers and fully connected layers (see table 2 and figure 7).
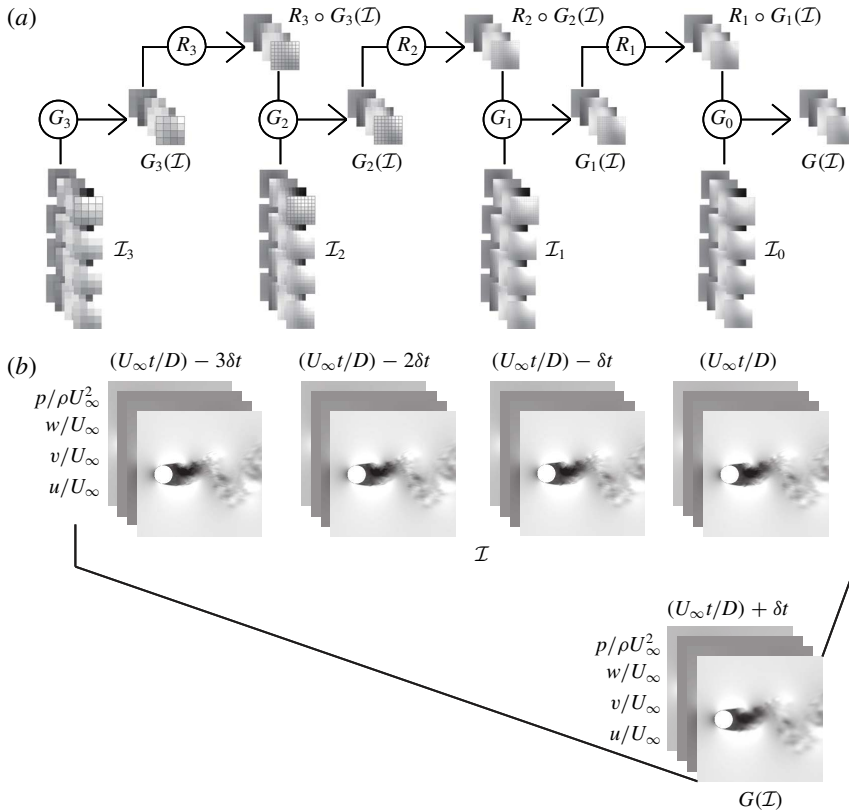
FIGURE 6. (*a*) Schematic diagram of generator models. $\mathcal{I}$ is the set of input flow fields (see figure 2) and $\mathcal{I}_k$ denotes interpolated input flow fields on an identical domain with $1/(2^k \times 2^k)$ coarser grid resolution. G_k indicates a generative CNN which is fed with input $\mathcal{I}_k$, while $G_k(\mathcal{I})$ indicates the set of predicted flow fields from the generative CNN G_k. $R_k \circ ()$ indicates the rescale operator, which upscales the grid size twice in both directions. (*b*) Example of input flow fields and the corresponding prediction of the flow field on a test data.

A discriminative network D_k is fed with inputs of predicted flow fields from the generative CNN ($G_k(\mathcal{I})$) and ground truth flow fields ($\mathcal{G}_k(\mathcal{I})$). Convolution layers of a discriminative network extract low-dimensional features or representations of predicted flow fields and ground truth flow fields through convolution operations. Then 2×2 max pooling, which extracts the maximum values from each equally divided 2×2 sized grid on a feature map, is added after convolution layers to pool the most important features. The max pooling layer outputs feature maps with resolution of $32/2^{k+1} \times 32/2^{k+1}$. The pooled features are connected to fully connected layers. Fully connected layers compare pooled features to classify ground truth flow fields into class 1 and predicted flow fields into class 0. The output of each discriminative network is a single continuous scalar between 0 and 1, where an output value larger than a threshold (0.5) is classified into class 1 and an output value smaller than the threshold is classified into class 0. Output neurons of the last fully connected layer of each discriminative network D_k are activated using the sigmoid function to bound

Discriminator model (convolution layers $\rightarrow$ max pooling layer $\rightarrow$ fully connected layers)			
D_0	D_1	D_2	D_3
Convolution layers (top row: numbers of feature maps, bottom row: kernel sizes)			
4,128,256,512,128	4,128,256,256	4,64,128,128	4,64
7×7, 7×7, 5×5, 5×5	5×5, 5×5, 5×5	3×3, 3×3, 3×3	3×3
2×2 max pooling layer			
Fully connected layers (neuron numbers)			
$16 \times 16 \times 128$,1024,512,1	$8 \times 8 \times 256$,1024,512,1	$4 \times 4 \times 128$,1024,512,1	$2 \times 2 \times 64$,512,256,1

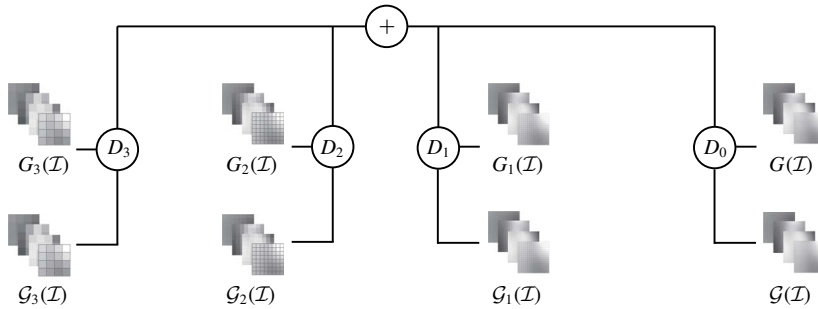TABLE 2. Configuration of the discriminator model inside the GAN.

FIGURE 7. Schematic diagram of the discriminator model: D_k indicates the discriminative network which is fed with $G_k(\mathcal{I})$ and $\mathcal{G}_k(\mathcal{I})$, $G_k(\mathcal{I})$ indicates the set of predicted flow fields from the generative CNN G_k, while $\mathcal{G}_k(\mathcal{I})$ indicates the set of ground truth flow fields.

the output values within 0 to 1, while other output neurons, including feature maps of convolution layers, are activated with the ReLU activation function.

Note that the number of neurons in the first layer of fully connected layers (see table 2) is a function of the square of the subsampled input resolution (32×32); as a result, parameters to learn are increased in the order of the square of the subsampled input resolution. Training could be inefficient or nearly impossible in a larger input domain size with the equivalent resolution (for example, 250×250 resolution on the domain size of $7D \times 7D$) due to the fully connected layer in the discriminator model depending on computing hardware. On the other hand, parameters in the generator model (fully convolutional architecture with padded convolutions) do not depend on the size and resolution of the subsampled inputs. This enables the generator model to predict flow fields in a larger domain size ($7D \times 7D$ domain with 250×250 resolution) compared to the subsampled input domain size ($0.896D \times 0.896D$ domain with 32×32 resolution).

The generator model is trained with the Adam optimizer, which is known to efficiently train a network, particularly in regression problems (Kingma & Ba 2014). This optimizer computes individual learning rates, which are updated during training, for different weight parameters in a network. The maximum learning rate of the parameters in the generator model is limited to 4×10^{-5}. However, the Adam

optimizer is reported to perform worse than a gradient descent method with a constant learning rate for a classification problem using CNNs (Wilson *et al.* 2017). As the discriminator model performs classification using CNNs, the discriminator model is trained with the gradient descent method along with a constant learning rate of 0.02. The same optimization method and learning rate have also been utilized in the discriminator model by Mathieu *et al.* (2015). Networks are trained up to 6×10^5 iterations with a batch size of 8. Training of networks is observed to be sufficiently converged without overfitting, as shown in figure 21 in § C.1.

3.3. *Conservation principles*

Let Ω be an arbitrary open, bounded and connected domain in $\mathbb{R}^3$, $\partial\Omega$ be a surface of which an outward unit normal vector can be defined as $\hat{n} = (n^1, n^2, n^3)$. Also let $\rho(t, \boldsymbol{x})$ be the density, $\boldsymbol{u}(t, \boldsymbol{x}) = (u_1, u_2, u_3)$ be the velocity vector, $p(t, \boldsymbol{x})$ be the pressure and $\tau(t, \boldsymbol{x})$ be the shear stress tensor ($\tau_{ij} = \rho\nu(\partial u_j/\partial x_i)$) of ground truth flow fields as a function of time t and space $\boldsymbol{x} \in \mathbb{R}^3$. Then conservation laws for mass and momentum can be written as follows:

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega} \rho \, \mathrm{d}V = - \int_{\partial\Omega} \rho u_j n^j \, \mathrm{d}S \tag{3.3}$$

and

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega} \rho u_i \, \mathrm{d}V = - \int_{\partial\Omega} (\rho u_i) u_j n^j \, \mathrm{d}S - \int_{\partial\Omega} (p\delta_{ij}) n^j \, \mathrm{d}S + \int_{\partial\Omega} \tau_{ji} n^j \, \mathrm{d}S, \tag{3.4}$$

where δ_{ij} is the Kronecker delta. The present study utilizes subsets of three-dimensional data (two-dimensional slices). Therefore, the domain Ω becomes a surface in $\mathbb{R}^2$ and the surface $\partial\Omega$ becomes a line in $\mathbb{R}^1$. Exact mass and momentum conservation cannot be calculated because derivatives in the spanwise direction are not available in two-dimensional slice data. Instead, conservation principles of mass and momentum in a flow field predicted by deep learning are considered in a form that compares the difference between predicted and ground truth flow fields in a two-dimensional space ($\mathbb{R}^2$).

Extension of the present deep learning methods to three-dimensional volume flow fields is algorithmically straightforward. However, the increase of the required memory space and the operation counts is significant, making the methods impractical. For example, the memory space and the operation counts for $32 \times 32 \times 32$ sized volume flow fields are estimated to be increased by two-orders of magnitude compared to those required for the 32×32 two-dimensional flow fields.

3.4. *Loss functions*

For a given set of input and ground truth flow fields, the generator model predicts flow fields that minimize a total loss function, which is a combination of specific loss functions as follows:

$$\mathcal{L}_{generator} = \frac{1}{N\lambda_{\sum}} \sum_{k=0}^{N-1} \{\lambda_{l2}\mathcal{L}_2^k + \lambda_{gdl}\mathcal{L}_{gdl}^k + \lambda_{phy}(\mathcal{L}_c^k + \mathcal{L}_{mom}^k) + \lambda_{adv}\mathcal{L}_{adv}^{G,k}\}, \tag{3.5}$$

where $N(= 4)$ is the number of scales of the multi-scale CNN and $\lambda_{\sum} = \lambda_{l2} + \lambda_{gdl} + \lambda_{phy} + \lambda_{adv}$. Contributions of each loss function can be controlled by tuning coefficients λ_{l2}, λ_{gdl}, λ_{phy}, and λ_{adv}.

Function $\mathcal{L}_2^k$ minimizes the difference between predicted flow fields and ground truth flow fields (see (A 1)), while $\mathcal{L}_{gdl}^k$ is applied to sharpen flow fields by directly penalizing gradient differences between predicted flow fields and ground truth flow fields (see (A 2)). Loss functions $\mathcal{L}_2^k$ and $\mathcal{L}_{gdl}^k$ provide prior information to networks that predicted that flow fields should resemble ground truth flow fields. These loss functions support networks to learn fluid dynamics that resemble the flow field, by extracting features in a supervised manner.

Function $\mathcal{L}_c$ enables networks to learn mass conservation by minimizing the total absolute sum of differences of mass fluxes in each cell in an x–y plane as defined in (A 3). Function $\mathcal{L}_{mom}$ enables networks to learn momentum conservation by minimizing the total absolute sum of differences of momentum fluxes due to convection, pressure gradient and shear stress in each cell in an x–y plane as defined in (A 4). Loss functions $\mathcal{L}_c$ and $\mathcal{L}_{mom}$, which are denoted physical loss functions, provide explicit prior information on physical conservation laws to networks, and support networks to extract features including physical conservation laws in a supervised manner. Consideration of conservation of kinetic energy can also be realized using a loss function, but it is not included in the present study since the stability of flow fields predicted by the present networks are not affected by the conservation of kinetic energy.

Function $\mathcal{L}_{adv}^G$ is a loss function with the purpose of deluding the discriminator model into classifying generated flow fields as ground truth flow fields (see (A 5)). The loss function $\mathcal{L}_{adv}^G$ provides knowledge in a concealed manner that features of the predicted and the ground truth flow fields should be indistinguishable. This loss function supports networks to extract features of underlying fluid dynamics in an unsupervised manner.

The loss function of the discriminator model is defined as follows:

$$\mathcal{L}_{discriminator} = \frac{1}{N} \sum_{k=0}^{N-1} [L_{bce}(D_k(\mathcal{G}_k(\mathcal{I})), 1) + L_{bce}(D_k(G_k(\mathcal{I})), 0)], \tag{3.6}$$

where L_{bce} is the binary cross-entropy loss function defined as

$$L_{bce}(a, b) = -b \log(a) - (1 - b) \log(1 - a), \tag{3.7}$$

for scalar values a and b between 0 and 1. Function $\mathcal{L}_{discriminator}$ is minimized so that the discriminator model appropriately classifies ground truth flow fields into class 1 and predicted flow fields into class 0. The discriminator model learns flow fields in a low-dimensional feature space.

4. Results

4.1. *Comparison of deep learning networks*

Four deep learning networks with different combinations of coefficients for loss functions are discussed in the present section. Case A employs a GAN with physical loss functions ($\lambda_{l2} = \lambda_{gdl} = 1.0$, $\lambda_{phy} = 1.0$ and $\lambda_{adv} = 0.1$); Case B employs a GAN without physical loss functions ($\lambda_{l2} = \lambda_{gdl} = 1.0$, $\lambda_{phy} = 0$ and $\lambda_{adv} = 0.1$); Case C employs a multi-scale CNN with physical loss functions ($\lambda_{l2} = \lambda_{gdl} = 1.0$, $\lambda_{phy} = 1.0$ and $\lambda_{adv} = 0$); Case D employs a multi-scale CNN without physical loss functions ($\lambda_{l2} = \lambda_{gdl} = 1.0$, $\lambda_{phy} = 0$ and $\lambda_{adv} = 0$). See §§ C.2 and C.3 for the determination of the weight parameters λ_{adv} and λ_{phy}, respectively. All deep learning cases (Cases A–D)
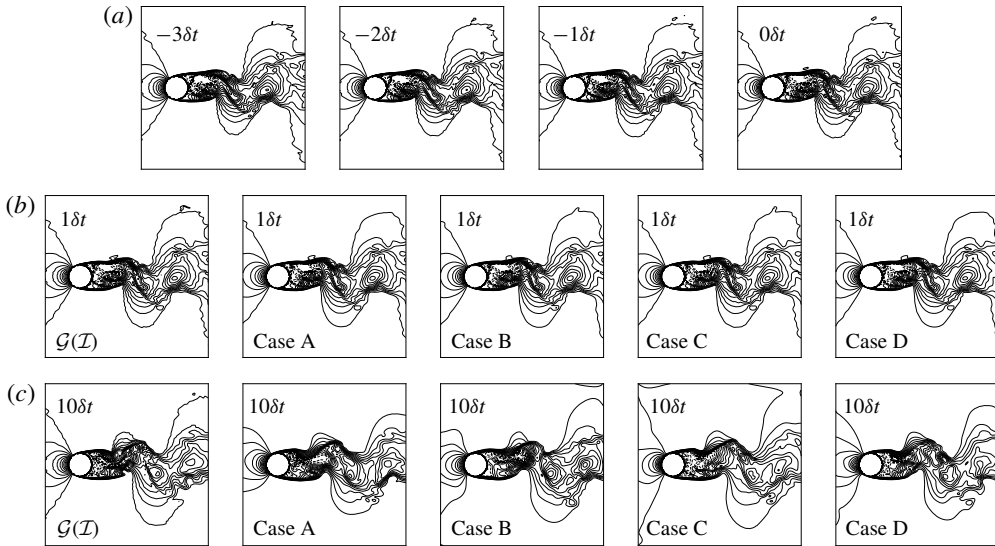
FIGURE 8. Comparison of the streamwise velocity (u/U_∞) at $Re_D = 3900$ predicted in Cases A–D. (*a*) Input set; (*b*) after a single prediction step ($1\delta t$), and (*c*) nine more recursive prediction steps ($10\delta t$). 20 contour levels from -0.5 to 1.0 are shown. Solid lines and dashed lines indicate positive and negative contour levels, respectively.

are trained with flow fields at $Re_D = 300$ and 500, which are in the three-dimensional wake transition regime, and tested on flow fields at $Re_D = 150$ (the two-dimensional vortex shedding regime), 400 (the same flow regime with training) and 3900 (the shear-layer transition regime).

Predicted flow fields at $Re_D = 3900$ from Cases A–D are shown in figure 8. Flow fields after time steps larger than δt are predicted recursively by utilizing flow fields predicted at prior time steps as parts of the input. Flow fields predicted after a single time step ($1\delta t$) are found to agree well with ground truth flow fields for all deep learning cases, even though the trained networks have not seen such small-scale flow structures at a higher Reynolds number. Note that the time-step size for network prediction δt corresponds to 20 times the simulation time-step size. Differences between the predicted and the ground truth flow fields increase as the number of recursive steps increases because errors from the previous predictions are accumulated to the next time-step prediction. Particularly, dissipation of small-scale flow structures in the wake region is observed, while large-scale vortical motions characterizing Kármán vortex shedding are well predicted.

Local distributions of errors for the streamwise velocity after a single time step for four deep learning cases are compared in figure 9, while global errors such as L_2, L_∞, L_c and L_{mom} as a function of the recursive time step are compared in figure 10. See appendix B for definitions of errors. All networks show that the maximum errors are located in accelerating boundary layers on the cylinder wall or in the braid region in the wake. Steep velocity gradients captured with relatively coarse resolution in the deep learning prediction are considered as the cause for relatively high errors in accelerating boundary layers. Magnitudes of the maximum errors at $Re_D = 400$ are found to be smaller (see figure 9*b*) than those at $Re_D = 150$ (figure 9*a*) and
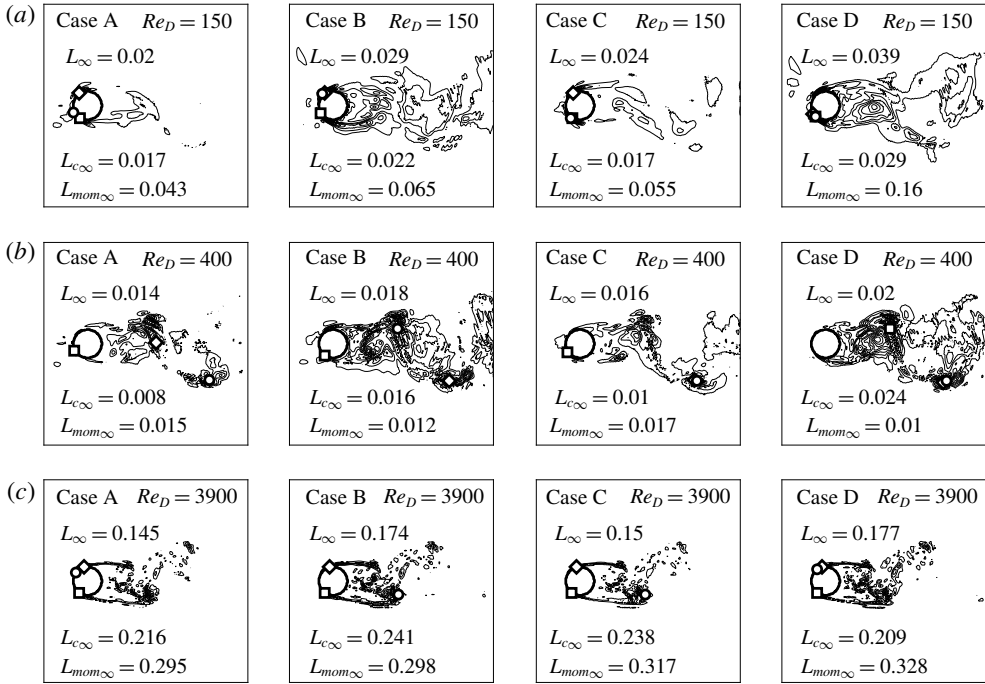
FIGURE 9. Local distributions of errors for u/U_∞ after a single prediction step at (a) $Re_D = 150$ (10 contour levels from 0 to 0.04), (b) $Re_D = 400$ (10 contour levels from 0 to 0.04) and (c) $Re_D = 3900$ (10 contour levels from 0.0 to 0.33). Locations of L_∞, $L_{c\infty}$ (maximum error in mass conservation) and $L_{mom\infty}$ (maximum error in momentum conservation) are indicated by $\bigcirc$, $\diamondsuit$ and $\square$, respectively.

3900 (figure 9c). This result implies that a network performs best in predicting flow fields in a regime that has been utilized during training, while the network shows relatively large errors in predicting flow fields in the flow regime with higher complexity.

Interestingly, unlike errors at $1\delta t$, as the recursive prediction step advances, errors at $Re_D = 150$ are observed to increase more slowly than those at $Re_D = 400$ (see figure 10). This implies that deep learning networks are capable of effectively learning large-scale or mainly two-dimensional vortex shedding physics from flow in three-dimensional wake transition regimes ($Re_D = 300$ and 500), thereby accurately predicting two-dimensional vortex shedding at $Re_D = 150$, flow fields of which are not included in the training dataset.

As also shown in figure 10, the multi-scale CNN with physical loss functions (Case C) shows reduction of L_c and L_{mom} errors, during recursive prediction steps, compared to the multi-scale CNN without physical loss functions (Case D), indicating the advantage of the incorporation of physical loss functions in improving the conservation of mass and momentum. At the same time, however, L_2 and L_∞ errors at $Re_D = 400$ and 3900 are found to increase in Cases C and D. Case A, which employs the GAN with physical loss functions, shows similar error trends to Case C but with smaller magnitudes of the L_∞ error at $Re_D = 150$.

On the other hand, the GAN without physical loss functions (Case B) shows smaller L_2 and L_{mom} errors for all three Reynolds number cases than those in Case D which
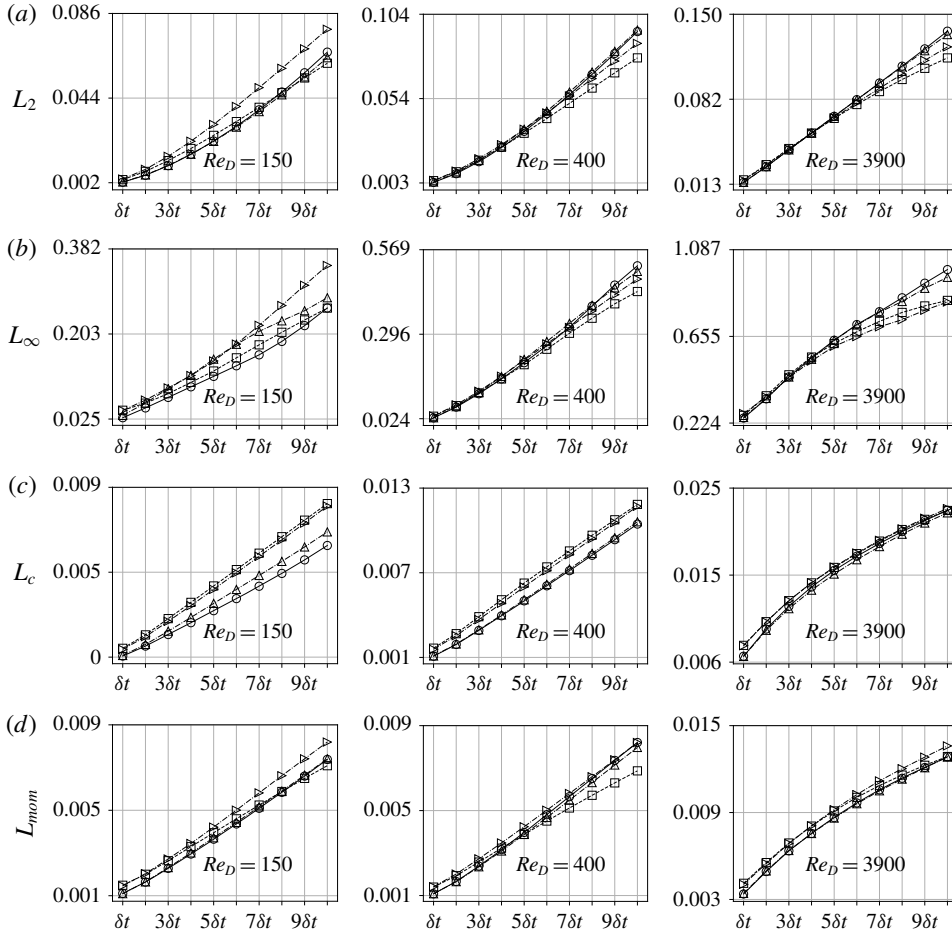
FIGURE 10. Comparisons of (*a*) L_2, (*b*) L_∞, (*c*) L_c and (*d*) L_{mom} errors for Cases A–D. See appendix B for definitions of the errors. The time-step interval between flow fields is $\delta t = 20\Delta t U_\infty/D = 0.1$. ○ and solid line: Case A; □ and dashed line: Case B; △ and dash-dotted line: Case C; ▷ and dotted line: Case D.

employs the multi-scale CNN without physical loss functions. The L_∞ errors in Case B at $Re_D = 150$ and 400 are also significantly smaller than those in Case D. These results imply that GANs (with and without physical loss functions, Cases A and B) and the multi-scale CNN with physical loss functions (Case C) are more capable of extracting features related to unsteady vortex shedding physics over a circular cylinder than the multi-scale CNN without physical loss functions (Case D). The GAN without physical loss function (Case B) is found to consistently reduce errors associated with resemblance (L_2 and L_∞) while error behaviours associated with conservation loss functions are rather inconsistent. Effects of physical loss functions on reduction of conservation errors are identifiable for networks with physical loss functions (Cases A and C).

Vortical structures at each Reynolds number predicted by the present four deep learning networks appear to be similar to each other after a single prediction step

as shown in figure 11(*a*). However, all deep learning cases have difficulties in learning the production of small-scale vortical structures. At $10\delta t$, small-scale vortical structures, which are not present in the ground truth flow field, are found to be generated inside shed large-scale vortices at $Re_D = 150$, while many small-scale vortices are missed in the wake at $Re_D = 3900$ (figure 11*b*). This observation implies that a network has difficulty in predicting flow fields, especially in recursive predictions as errors from previous predictions are accumulated, in flow regimes which are different from the regime for training.

After a few recursive prediction steps, Case D, where the multi-scale CNN without physical loss functions is applied, shows unphysical vortical structures near the front stagnation point, which are not present in flow fields predicted by other cases at the three considered Reynolds numbers (figure 11*b*). The effect of the inaccurate prediction in Case D on errors also appears in figure 10, where magnitudes are larger than those in Cases A, B and C.

All deep learning cases are found to be capable of predicting future flow fields, particularly in single-step predictions. However, networks with additional consideration of physics in either a supervised or an unsupervised manner (Cases A–C) are recommended for predicting further future flow fields with many recursive steps. Especially, the GAN without physical loss functions (Case B) is found to be the best among the considered networks for minimizing L_2 and L_∞ errors (see figure 10) while also satisfying the conservation of mass and momentum favorably.

4.2. *Analysis on captured and missed flow physics*

Discussion in the present section is focused on the GAN without physical loss functions (Case B), which is trained with flow fields at $Re_D = 300$ and 500 (the three-dimensional wake transition regime) and tested on flow fields at $Re_D = 150$ (the two-dimensional vortex shedding regime), 400 (the same flow regime with training) and 3900 (the shear-layer transition regime), in order to assess what flow characteristics the network captures or misses.

Contour plots of the spanwise vorticity calculated using ground truth velocity fields and velocity fields predicted by the GAN are compared in figure 12 for three Reynolds numbers at $1\delta t$ and $10\delta t$. First of all, laminar flow at the frontal face of the cylinder as well as the separated laminar shear layers including lengthening of the shear layers and detachment from the wall are observed to be well captured in all three Reynolds number cases. Convection (downstream translation) and diffusion of overall large-scale vortical structures in the wake are also well predicted at both $1\delta t$ and $10\delta t$. However, as also mentioned in the previous section, prediction results show differences in the generation and dissipation of small-scale vortices. After a number of recursive prediction steps, along with the non-zero spanwise velocity, unexpected smaller scale vortices than those present in the ground truth flow field are generated at $Re_D = 150$, in which Reynolds number regime, downstream vortical structures are expected to be laminar and two-dimensional. Generation of smaller scale vortical structures than those in ground truth flow fields after a few recursive predictions is also noticed in the GAN prediction at $Re_D = 400$. On the other hand, it is found that the GAN fails to accurately predict small-scale vortical structures inside large-scale vortices at $Re_D = 3900$. It is thought that the present results imply that the GAN is not fully trained for predicting production and dissipation of small-scale vortices. The lack of flow information along the spanwise direction is considered as a major cause for this failure. Due to the reason mentioned in § 3.3, the spanwise information in
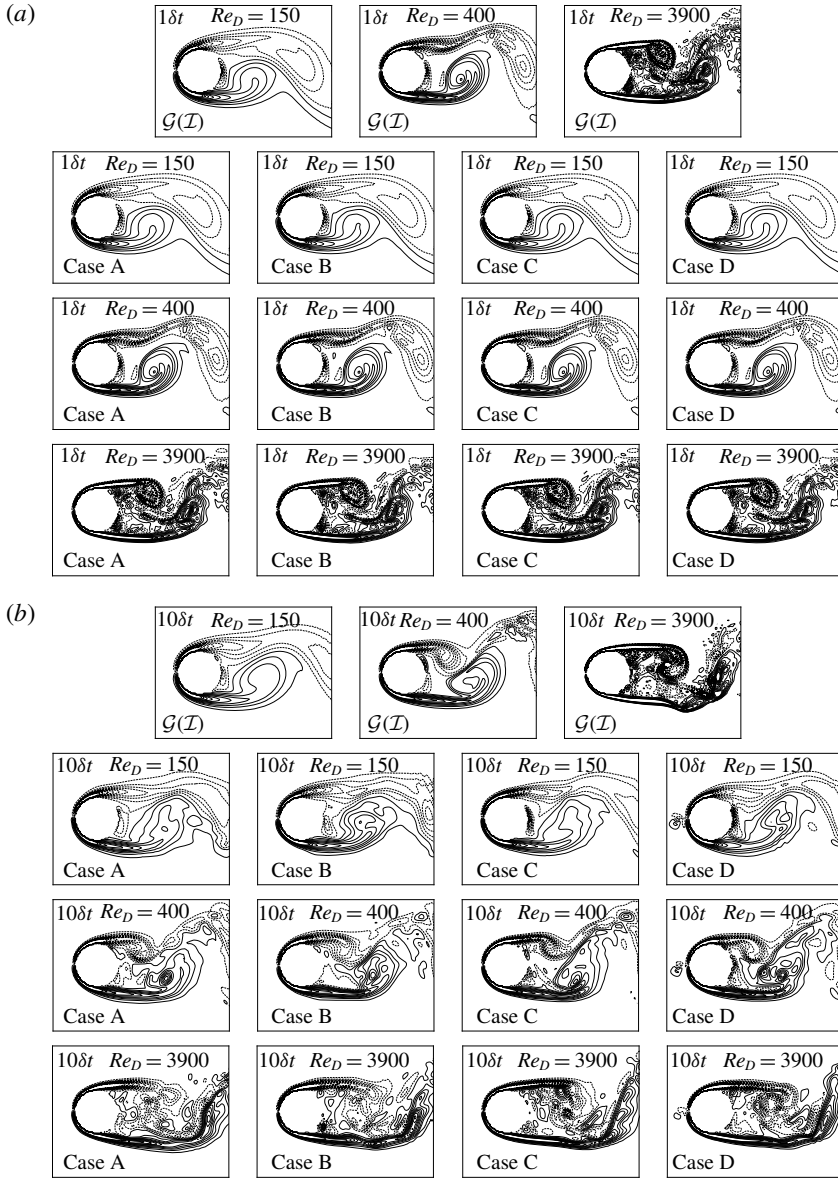
FIGURE 11. Comparison of the spanwise vorticity after (*a*) a single prediction step ($1\delta t$) and (*b*) nine more recursive prediction steps ($10\delta t$). 20 contour levels from -10.0 to 10.0 are shown. Solid lines and dashed lines indicate positive and negative contour levels, respectively.

the present training dataset includes only the spanwise velocity on a two-dimensional sliced domain, and therefore misses variation of flow variables along the spanwise direction.

The lack of spanwise information on flow variables seems to lead the network to miss the mechanism for generation of small-scale vortices, which can be formulated
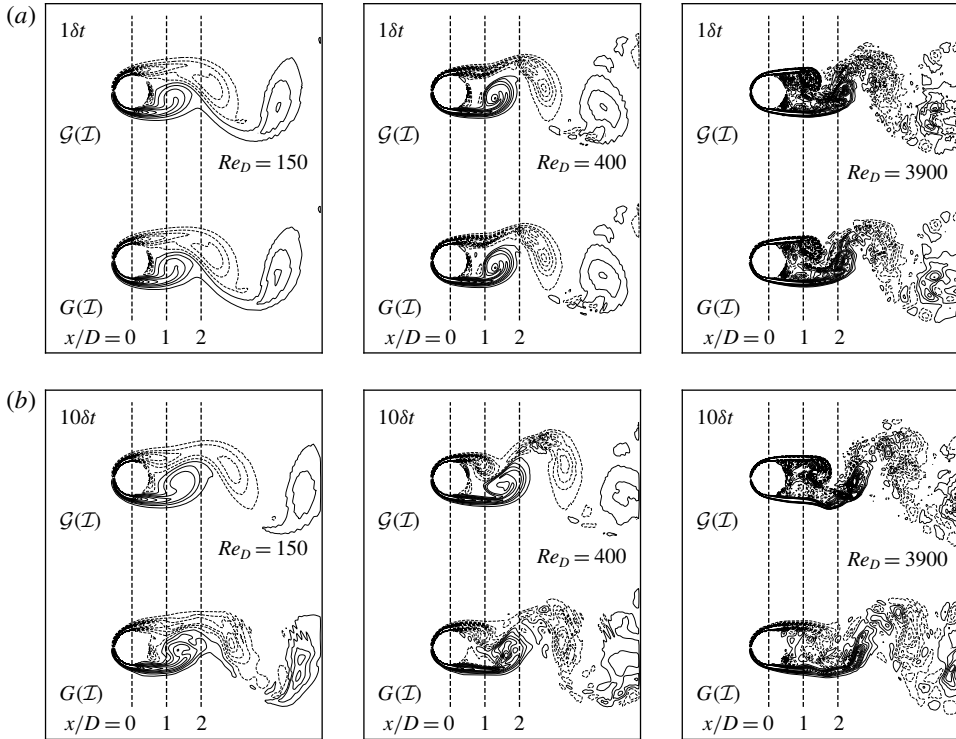
FIGURE 12. Contours of the spanwise vorticity calculated using ground truth velocity fields ($\mathcal{G}(\mathcal{I})$) and velocity fields predicted by the GAN ($G(\mathcal{I})$) after (*a*) a single prediction step ($1\delta t$) and (*b*) nine more recursive prediction steps ($10\delta t$) at $Re_D = 150$, 400 and 3900. 20 contour levels from -10.0 to 10.0 are shown. Solid lines and dashed lines indicate positive and negative contour levels, respectively.

as the vortex stretching term in the spanwise vorticity (ω_z) equation. The stretching term $\omega_z(\partial w/\partial z)$, which is associated with the generation of small-scale vortices, is missed in the present training. On the other hand, convection and diffusion of the spanwise vorticity are dominated by $u(\partial \omega_z/\partial x) + v(\partial \omega_z/\partial y)$ and $(1/Re_D)(\partial^2 \omega_z/\partial x \partial x + \partial^2 \omega_z/\partial y \partial y)$, which can be rather easily trained using the given flow field data.

Convection and diffusion phenomena in flow around a cylinder are investigated more quantitatively in the development of the velocity deficit. Profiles of the streamwise velocity from ground truth flow fields ($\bigcirc$) and flow fields predicted by the GAN (solid lines) at three streamwise locations, $x/D = 0$, 1.0 and 2.0, are compared in figure 13. Velocity profiles at $x/D = 0$ show no identifiable differences between ground truth and GAN flow fields at both $1\delta t$ and $10\delta t$ at all Reynolds numbers ($Re_D = 150$, 400 and 3900). This is because flow at $x/D = 0$ is laminar two-dimensional boundary layer flow, the characteristics of which are rather easily trained by the network. Noticeable differences in the velocity deficit are observed in the comparison at $10\delta t$ in the wake region, $x/D = 2.0$, at $Re_D = 3900$, where small-scale oscillatory motions are not accurately captured by the GAN. Recursively predicted velocity deficits at $Re_D = 150$ and 400 are in good agreement with the
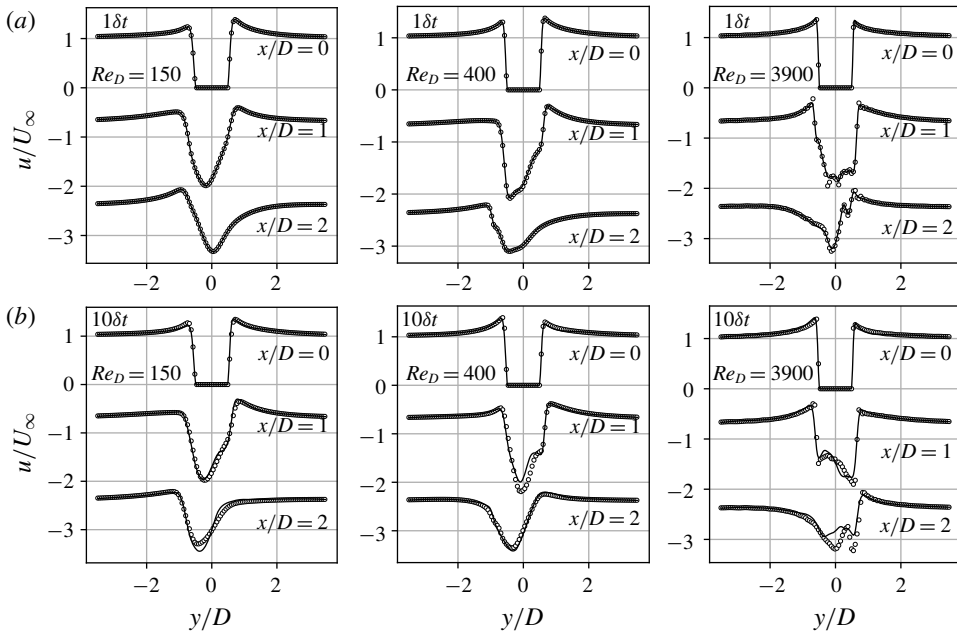
FIGURE 13. Profiles of the streamwise velocity at three streamwise locations after (*a*) a single prediction step (1δt) and (*b*) nine more recursive prediction steps (10δt) at $Re_D =$ 150, 400 and 3900. Circles indicate ground truth results and solid lines indicate results predicted by the GAN. Profiles at $x/D = 1.0$ and 2.0 are shifted by -1.7 and -3.4 in the vertical axis, respectively.

ground truth velocity deficit in terms of the peak, width and shape at both streamwise locations.

Plots of the power spectral density (PSD) of the streamwise velocity along the vertical axis (y) in the wake region at $x/D = 2.0$ are shown in figure 14 to evaluate the wavenumber content of wake flow. At $Re_D = 150$ and 400, PSDs produced by the GAN show good agreement with ground truth results in the single-step prediction (1δt), and are found to be still close to ground truth PSDs with marginal deviations in the middle- to high-wavenumber contents ($k > 10$) after nine recursive predictions. On the other hand, PSDs produced by the GAN at $Re_D = 3900$ at both 1δt and 10δt show deviations from ground truth PSDs, especially for high-wavenumber contents, again indicating the difficulty in learning the mechanism for production of small-scale vortices (high wavenumbers).

4.3. *Training with additional data*

The GAN without physical loss functions is trained with additional flow field data at Reynolds numbers of 1000 and 3000, in order to investigate the effect of small-scale contents in training data on the prediction of small-scale vortical motions in flow in the shear-layer transition regime ($Re_D = 3900$). Local distributions of errors for the streamwise velocity after a single time step for the GAN and the GAN with additional flow field data are compared in figure 15. Magnitudes of maximum errors, especially the mass and momentum errors, are significantly reduced by training the network with
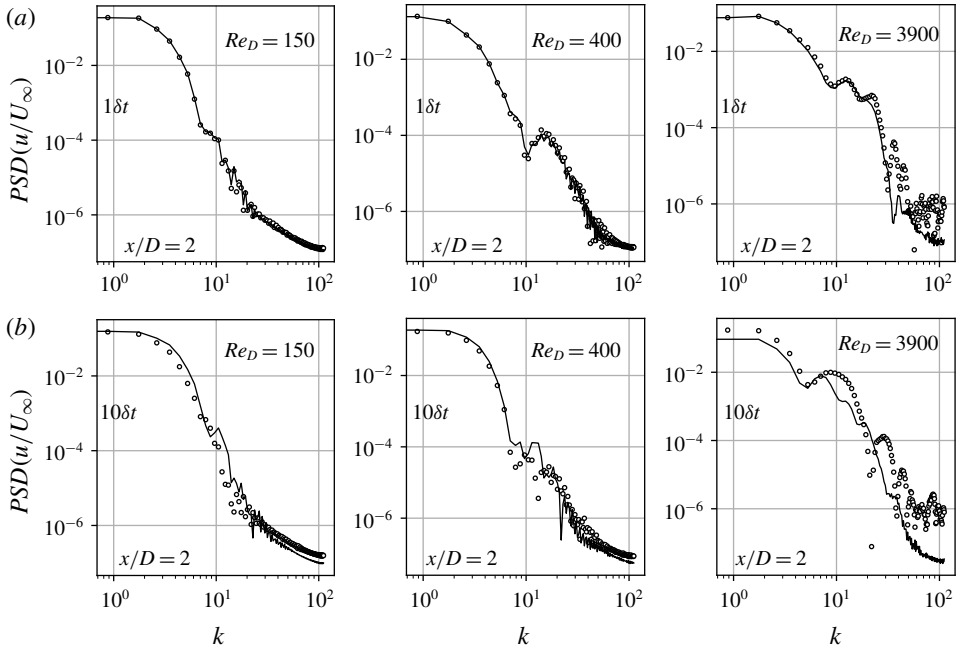
FIGURE 14. Power spectral density of the streamwise velocity at $x/D = 2.0$ after (*a*) a single prediction step ($1\delta t$) and (*b*) nine more recursive prediction steps ($10\delta t$) at $Re_D =$ 150, 400 and 3900. Circles indicate ground truth results and solid lines indicate results predicted by the GAN.
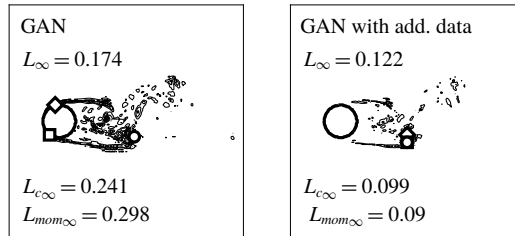

FIGURE 15. Local distributions of errors for u/U_∞ after a single prediction step at $Re_D =$ 3900 (10 contour levels from 0.0 to 0.33). Locations of L_∞, $L_{c\infty}$ (maximum error in mass conservation) and $L_{mom\infty}$ (maximum error in momentum conservation) are indicated by $\bigcirc$, $\diamondsuit$ and $\square$, respectively.

flow fields in the same flow regime as that to be predicted. Nevertheless, maximum errors are still larger than those at low Reynolds numbers (see figure 9*a*,*b*). The lack of spanwise information in the input is considered to be the remaining cause for the errors.

Contours of the spanwise vorticity calculated by ground truth flow fields, flow fields predicted by the GAN trained with data at $Re_D = 300$ and 500 and flow fields predicted by the GAN trained with additional data at $Re_D = 1000$ and 3000 are
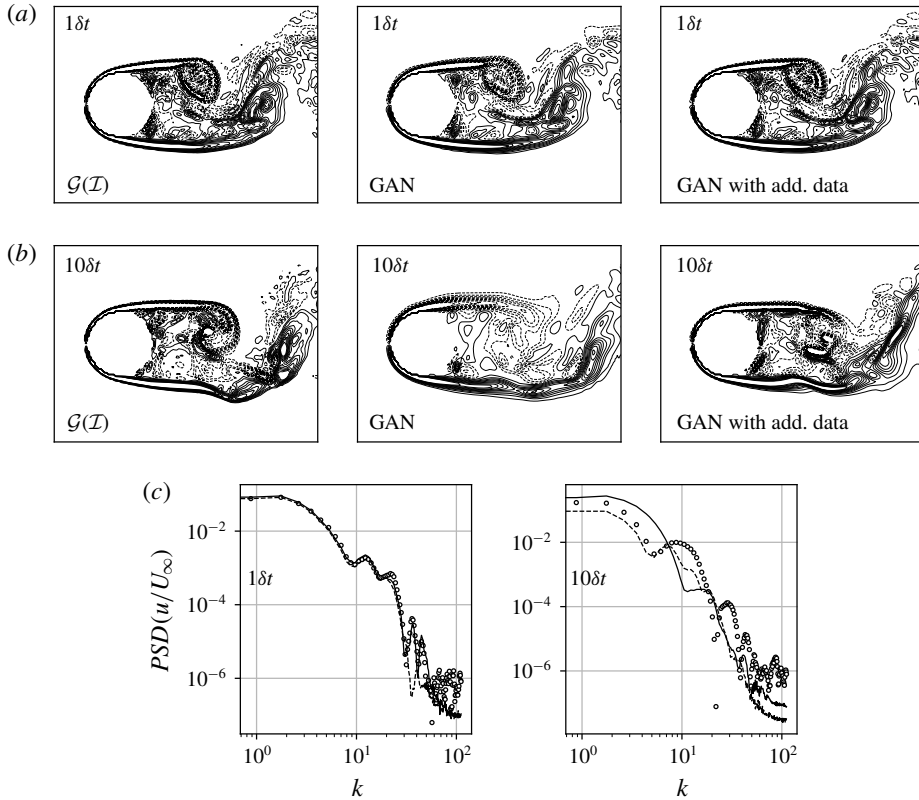
FIGURE 16. Contour plots of the spanwise vorticity calculated using ground truth velocity fields and velocity fields predicted by the GANs at $Re_D = 3900$ after (*a*) $1\delta t$ and (*b*) $10\delta t$. Solid lines and dashed lines denote positive and negative contour levels, respectively. (*c*) Plots of the power spectral density at $1\delta t$ and $10\delta t$. 20 contour levels from -10.0 to 10.0 are shown. Circles indicate ground truth result, while the dashed line and the solid line correspond to predicted results using the GAN and the GAN with additional data, respectively.

compared in figure 16(*a*,*b*). Training with additional data in the same flow regime is found to clearly improve the prediction of small-scale motions after a single prediction step ($1\delta t$). The spanwise vorticity predicted by the GAN trained with additional data is found to agree much better with the ground truth vorticity than that predicted by the GAN trained with flow fields only at $Re_D = 300$ and 500 after nine more recursive prediction steps ($10\delta t$) as shown in figure 16(*b*). However, as discussed in the previous section (§ 4.2), the GAN trained with additional data also suffers from a lack of production of small-scale vortical structures. PSDs produced by the GAN trained for $Re_D = 300$ and 500 and the GAN trained with additional data are close to the ground truth PSD at $1\delta t$, while the GAN trained with additional data better predicts small-scale high-wavenumber contents. Differences among predicted and ground truth PSDs become larger at $10\delta t$, where reduced small-scale high-wavenumber contents are clearly observable for both GANs (figure 16*c*).
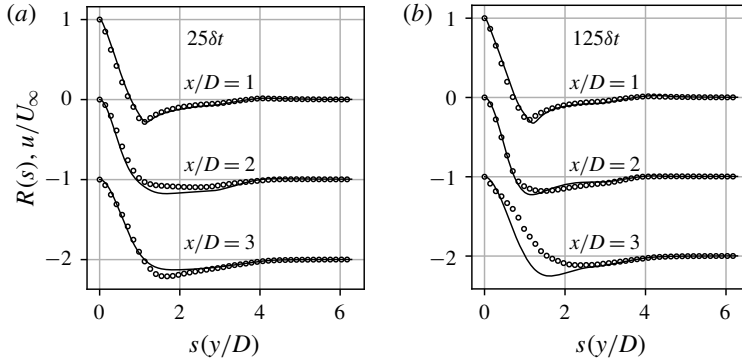
FIGURE 17. Two-point correlations of the streamwise velocity at three downstream locations ($x/D = 1.0$, 2.0 and 3.0) at $Re_D = 3900$. Circles indicate ground truth results and solid lines indicate predicted results by the GAN. Two-point correlations at $x/D = 2.0$ and 3.0 are shifted by -1 and -2 along the vertical axis, respectively.

4.4. *Training with a large time-step interval*

To investigate the potential of using a GAN in practical applications, where predicting large-scale flow motions is important, the GAN without physical loss functions is trained with a large time-step interval of $25\delta t = 500\Delta t U_\infty/D = 2.5$. This time-step interval is 25 times larger than the previous deep learning time-step interval and 500 times larger than the simulation time-step interval. Figure 17 shows plots of two-point correlations of the streamwise velocity along the y direction, which provide information on the large-scale fluid motions at three downstream wake locations at $Re_D = 3900$. After a single step with $25\delta t$, it is found that two-point correlations predicted by the GAN are in good agreement with correlations of the ground truth flow field. After four additional recursive large steps ($125\delta t$), however, small deviations of correlations from ground truth results are observed in the downstream wake region ($x/D = 3.0$). Note that $125\delta t$ corresponds to 2500 time steps of the numerical simulation conducted for the ground truth flow field.

Contour plots of the streamwise velocity predicted by the GAN at $Re_D = 3900$ are shown in figure 18 (see figures 28–30 in appendix E for contour plots of the other flow variables). Flow fields at $50\delta t$, $75\delta t$, $100\delta t$ and $125\delta t$ are recursively predicted. As shown in figure 18, large-scale oscillations of the streamwise velocity behind the cylinder are well predicted, while small-scale flow structures are found to be rather rapidly dissipated compared to those in ground truth flow fields. This may be partly due to the dynamics of small-scale flow structures, the time scales (τ) of which are smaller than the training interval size ($t = nD/U_\infty\delta t$, where n is an integer), and are disregarded from input information. The time scale of a small-scale flow structure can be approximated as

$$\tau \sim \left(\frac{\nu}{\epsilon}\right)^{1/2} \tag{4.1}$$

according to Tennekes & Lumley (1972), where ν is the kinematic viscosity and ϵ is the dissipation rate per unit mass that is approximated as

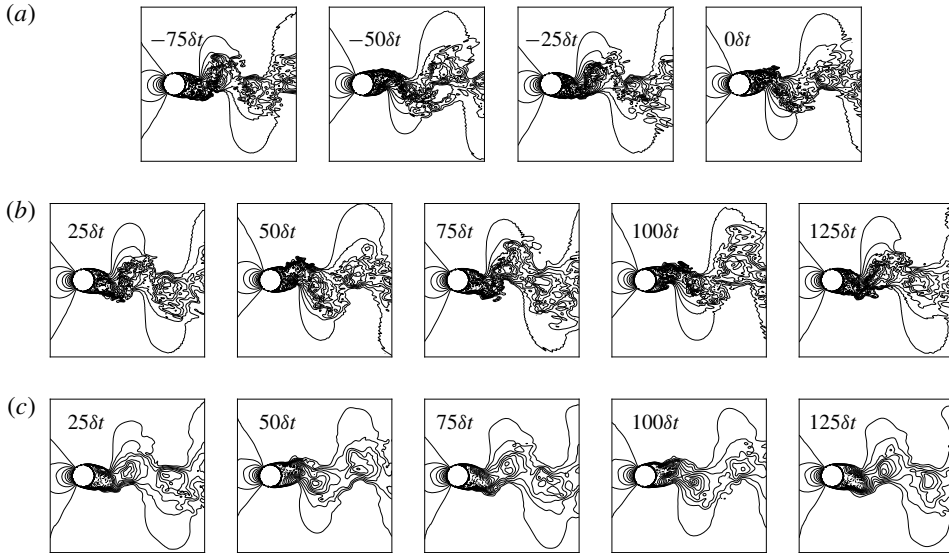$$\epsilon \sim \frac{u^3}{l} \sim \frac{U_\infty^3}{D}, \tag{4.2}$$

FIGURE 18. Contour plots of the streamwise velocity (u/U_∞) at $Re_D = 3900$ after $25\delta t$, $50\delta t$, $75\delta t$, $100\delta t$ and $125\delta t$, where $1\delta t = 20\Delta t U_\infty/D = 0.1$. Flow fields at $50\delta t$, $75\delta t$, $100\delta t$ and $125\delta t$ are recursively predicted (utilizing flow fields predicted from prior time steps as part of the input). (*a*) Input set, (*b*) ground truth flow fields and (*c*) flow fields predicted by the GAN. 20 contour levels from -0.5 to 1.0 are shown. Solid lines and dashed lines indicate positive and negative contour levels, respectively.

where u is the velocity scale and l is the length scale of a large-scale flow motion. The ratio of the time-scale for a small-scale flow structure to the training interval size can be derived as follows:

$$\frac{\tau}{t} \sim \frac{\tau U_\infty}{nD\delta t} \sim \frac{1}{n\delta t} \left(\frac{U_\infty D}{\nu} \right)^{-1/2} = \frac{1}{n\delta t \sqrt{Re_D}}. \tag{4.3}$$

The ratio of the time scale for a small-scale flow structure to the training interval size decreases as the Reynolds number and the integer n increase. Therefore, small-scale flow structures are reasonably well captured by the network trained with a small training-step interval (see figures 24–27), while it is found that small-scale flow structures predicted by the network trained with a large training-step interval of $25\delta t$, rapidly disappear in the wake (see figures 18, 28–30).

Regardless of the rapid loss of small-scale flow structures in the wake, flow fields predicted after a single large prediction-step interval of $25\delta t$ exhibit lower errors compared to flow fields recursively predicted at 25 small prediction steps of $25 \times 1\delta t$ (see table 3). The reduction of errors implies that predicting with a network trained with a large time-step interval enables the network to focus more on energetic large-scale flow motions by disregarding small-scale flow motions.

5. Conclusion

Unsteady flow fields around a circular cylinder at Reynolds numbers that were not informed during training were predicted using deep learning techniques. Datasets of

Variable	Time-step interval	Number of recursive steps	L_∞	L_2
u/U_∞	δt	25	1.74 ± 0.28	0.062 ± 0.002
	$25\delta t$	1	0.93 ± 0.19	0.025 ± 0.001
v/U_∞	δt	25	1.72 ± 0.39	0.064 ± 0.003
	$25\delta t$	1	0.95 ± 0.15	0.032 ± 0.002
w/U_∞	δt	25	0.91 ± 0.19	0.030 ± 0.003
	$25\delta t$	1	0.74 ± 0.11	0.015 ± 0.001
$p/\rho U_\infty^2$	δt	25	0.94 ± 0.17	0.040 ± 0.004
	$25\delta t$	1	0.56 ± 0.11	0.012 ± 0.001

TABLE 3. Comparison of errors for each flow variable at $Re_D = 3900$ from predictions obtained after 25 small time-step intervals of $1\delta t$ and after a single large time-step interval of $25\delta t$. $1\delta t = 20\Delta t U_\infty/D = 0.1$. Errors are composed of the mean and standard deviations determined by 32 independent prediction results.

flow fields have been constructed using numerical simulations in three different flow regimes: a two-dimensional laminar vortex shedding regime, a three-dimensional wake transition regime and a shear-layer transition regime. The present deep learning techniques are found to well predict convection and diffusion of large-scale vortical structures, while the mechanism for production of small-scale vortical structures is difficult to account for. Depending on the training scheme, the present deep learning techniques are found also to be capable of successfully predicting large-scale flow motions with large time-step interval sizes, which can be two to three orders of magnitude larger than the time-step interval size for the conventional unsteady numerical simulations. Predictions using the present deep learning networks can be conducted with significantly lower computational cost than numerical simulations regardless of the Reynolds number. A wall-clock time of 0.3 s is required for a time-step advance using a single graphic processing unit (NVIDIA Titan Xp).

Four deep learning networks, GANs with and without physical loss functions and multi-scale CNNs with and without physical loss functions, have been trained and compared for their predictive performance. The physical loss functions proposed in the present study inform the networks with, explicitly, the conservation of mass and momentum. Adversarial training in the GAN allows the deep learning network to extract various flow features in an unsupervised manner. All four deep learning techniques are shown to be capable of predicting flow fields in the immediate future. However, for long-term prediction using a recursive technique, which employs the predicted flow fields as part of the input dataset, GANs and the multi-scale CNN with physical loss functions are shown to be better predictors than the multi-scale CNN without physical loss functions. It has been found that the GAN without physical loss functions is the best at achieving a good resemblance to the ground truth flow field during recursive predictions. Especially, GAN-based networks take advantage of unsupervised training, so they can be applied to problems where underlying physics is unknown *a priori*. The present deep learning methods are expected to be useful in many practical applications, such as real-time flow control and guidance of aero- or hydro-vehicles, fast weather forecasting, etc., where fast prediction of energetic large-scale flow motions is important.

Physical interpretability of deep learning techniques is still an open problem and, therefore, further research is necessary to enhance our understanding of the underlying

mechanisms of deep learning networks for prediction of fluid flow, especially in transitional and turbulent flow regimes.

Appendix A. Loss functions

Function $\mathcal{L}_2^k$ minimizes the difference between the predicted and the ground truth flow fields as follows:

$$\mathcal{L}_2^k = \|G_k(\mathcal{I}) - \mathcal{G}_k(\mathcal{I})\|_2^2. \tag{A 1}$$

Also, $\mathcal{L}_{gdl}^k$ is a second-order central-difference version of the gradient difference loss function proposed by Mathieu *et al.* (2015), which is applied to sharpen flow fields by directly penalizing gradient differences between the predicted and the ground truth flow fields as follows:

$$\mathcal{L}_{gdl}^k = \sum_i \sum_j \left| \left| \frac{(\mathcal{G}_k(\mathcal{I})_{(i+1,j)} - \mathcal{G}_k(\mathcal{I})_{(i-1,j)})}{2} \right| - \left| \frac{(G_k(\mathcal{I})_{(i+1,j)} - G_k(\mathcal{I})_{(i-1,j)})}{2} \right| \right| $$
$$+ \sum_i \sum_j \left| \left| \frac{(\mathcal{G}_k(\mathcal{I})_{(i,j+1)} - \mathcal{G}_k(\mathcal{I})_{(i,j-1)})}{2} \right| - \left| \frac{(G_k(\mathcal{I})_{(i,j+1)} - G_k(\mathcal{I})_{(i,j-1)})}{2} \right| \right|, \tag{A 2}$$

where the subscript (i, j) indicates grid indices in the discretized flow domain, and n_x and n_y indicate the number of grid cells in the x and y directions, respectively.

Let u^k, v^k, w^k and p^k be non-dimensionalized flow variables retrieved from ground truth flow fields ($\mathcal{G}_k(\mathcal{I})$) and $\widetilde{u}^k$, $\widetilde{v}^k$, $\widetilde{w}^k$ and $\widetilde{p}^k$ be non-dimensionalized flow variables retrieved from predicted flow fields ($G_k(\mathcal{I})$). Flow variables on the right, left, top and bottom cell surfaces are calculated by the arithmetic mean between two neighbouring cells as $\phi_r = \frac{1}{2}(\phi_{(i,j)} + \phi_{(i+1,j)})$, $\phi_l = \frac{1}{2}(\phi_{(i,j)} + \phi_{(i-1,j)})$, $\phi_t = \frac{1}{2}(\phi_{(i,j)} + \phi_{(i,j+1)})$ and $\phi_b = \frac{1}{2}(\phi_{(i,j)} + \phi_{(i,j-1)})$ for a variable ϕ which is a function of the grid index (i, j). Function $\mathcal{L}_c$ enables networks to learn mass conservation by minimizing the total absolute sum of mass flux differences in each cell in an x–y plane as follows:

$$\left. \begin{aligned} \Delta\text{Con.}_{(i,j)}^k &= |(u_r^k - u_l^k) - (\widetilde{u}_r^k - \widetilde{u}_l^k)| + |(v_t^k - v_b^k) - (\widetilde{v}_t^k - \widetilde{v}_b^k)|, \\ \mathcal{L}_c^k &= \sum_i \sum_j \Delta\text{Con.}_{(i,j)}^k. \end{aligned} \right\} \tag{A 3}$$

Function $\mathcal{L}_{mom}$ enables networks to learn momentum conservation by minimizing the total absolute sum of differences of momentum fluxes due to convection, pressure

gradient and shear stress in each cell in an *x*–*y* plane as follows:

$$
\begin{aligned}
\Delta\text{Mom.}_{(i,j)}^k = {} & |((u_r^k)^2 - (u_l^k)^2) - ((\widetilde{u}_r^k)^2 - (\widetilde{u}_l^k)^2)| + |(u_r^k v_r^k - u_l^k v_l^k) - (\widetilde{u}_r^k \widetilde{v}_r^k - \widetilde{u}_l^k \widetilde{v}_l^k)| \\
& + |((v_t^k)^2 - (v_b^k)^2) - ((\widetilde{v}_t^k)^2 - (\widetilde{v}_b^k)^2)| + |(v_t^k u_t^k - v_b^k u_b^k) - (\widetilde{v}_t^k \widetilde{u}_t^k - \widetilde{v}_b^k \widetilde{u}_b^k)| \\
& + |(p_r^k - p_l^k) - (\widetilde{p}_r^k - \widetilde{p}_l^k)| + |(p_t^k - p_b^k) - (\widetilde{p}_t^k - \widetilde{p}_b^k)| \\
& + \frac{1}{Re_D} \left\{ \left| \left(\frac{v_{(i+1,j)}^k - 2v_{(i,j)}^k + v_{(i-1,j)}^k}{\Delta_x} \right) - \left(\frac{\widetilde{v}_{(i+1,j)}^k - 2\widetilde{v}_{(i,j)}^k + \widetilde{v}_{(i-1,j)}^k}{\Delta_x} \right) \right| \right. \\
& + \left| \left(\frac{v_{(i,j+1)}^k - 2v_{(i,j)}^k + v_{(i,j-1)}^k}{\Delta_y} \right) - \left(\frac{\widetilde{v}_{(i,j+1)}^k - 2\widetilde{v}_{(i,j)}^k + \widetilde{v}_{(i,j-1)}^k}{\Delta_y} \right) \right| \\
& + \left| \left(\frac{u_{(i,j+1)}^k - 2u_{(i,j)}^k + u_{(i,j-1)}^k}{\Delta_y} \right) - \left(\frac{\widetilde{u}_{(i,j+1)}^k - 2\widetilde{u}_{(i,j)}^k + \widetilde{u}_{(i,j-1)}^k}{\Delta_y} \right) \right| \\
& \left. + \left| \left(\frac{u_{(i+1,j)}^k - 2u_{(i,j)}^k + u_{(i-1,j)}^k}{\Delta_x} \right) - \left(\frac{\widetilde{u}_{(i+1,j)}^k - 2\widetilde{u}_{(i,j)}^k + \widetilde{u}_{(i-1,j)}^k}{\Delta_x} \right) \right| \right\}, \\
\mathcal{L}_{mom}^k = {} & \sum_i \sum_j \Delta\text{Mom.}_{(i,j)}^k,
\end{aligned}
\tag{A4}
$$

where Δ_x and Δ_y are grid spacings in the *x* and *y* directions, respectively.

Function $\mathcal{L}_{adv}^G$ is a loss function with the purpose of deluding the discriminator model to classify generated flow fields as class 1 as follows:

$$
\mathcal{L}_{adv}^G = L_{bce}(D_k(G_k(\mathcal{I})), 1).
\tag{A5}
$$

Appendix B. Error functions

Let *u*, *v*, *w* and *p* be non-dimensionalized flow variables retrieved from ground truth flow fields and $\widetilde{u}$, $\widetilde{v}$, $\widetilde{w}$ and $\widetilde{p}$ be non-dimensionalized flow variables retrieved from predicted flow fields. Error functions are defined as follows:

$$
\begin{aligned}
L_2 = {} & \left(\frac{1}{4n_x n_y} \sum_i \sum_j \{ (u_{(i,j)} - \widetilde{u}_{(i,j)})^2 + (v_{(i,j)} - \widetilde{v}_{(i,j)})^2 \right. \\
& \left. + (w_{(i,j)} - \widetilde{w}_{(i,j)})^2 + (p_{(i,j)} - \widetilde{p}_{(i,j)})^2 \} \right)^{1/2},
\end{aligned}
\tag{B1}
$$

$$
\begin{aligned}
L_\infty = {} & \frac{1}{4} \left(\max_{i,j} |u_{(i,j)} - \widetilde{u}_{(i,j)}| + \max_{i,j} |v_{(i,j)} - \widetilde{v}_{(i,j)}| \right. \\
& \left. + \max_{i,j} |w_{(i,j)} - \widetilde{w}_{(i,j)}| + \max_{i,j} |p_{(i,j)} - \widetilde{p}_{(i,j)}| \right),
\end{aligned}
\tag{B2}
$$

$$
L_c = \frac{1}{n_x n_y} \sum_i \sum_j \Delta\text{Con.}_{(i,j)},
\tag{B3}
$$

$$
L_{mom} = \frac{1}{n_x n_y} \sum_i \sum_j \Delta\text{Mom.}_{(i,j)},
\tag{B4}
$$

where $\Delta\text{Con.}_{(i,j)}$ and $\Delta\text{Mom.}_{(i,j)}$ are defined in equations (A3) and (A4), respectively.

GM_{16}		
Generative CNN	Numbers of feature maps	Kernel sizes
G_3	16 $\mathcal{N}_1$ $\mathcal{N}_1$ 4	$3 \times 3,\ 3 \times 3,\ 3 \times 3$
G_2	20 $\mathcal{N}_1$ $\mathcal{N}_1$ 4	$5 \times 5,\ 3 \times 3,\ 5 \times 5$
G_1	20 $\mathcal{N}_1$ $\mathcal{N}_2$ $\mathcal{N}_2$ $\mathcal{N}_1$ 4	$5 \times 5,\ 3 \times 3,\ 3 \times 3,\ 3 \times 3,\ 5 \times 5$
G_0	20 $\mathcal{N}_1$ $\mathcal{N}_2$ $\mathcal{N}_2$ $\mathcal{N}_1$ 4	$7 \times 7,\ 5 \times 5,\ 5 \times 5,\ 5 \times 5,\ 7 \times 7$

GM_{18}		
Generative CNN	Numbers of feature maps	Kernel sizes
G_3	16 $\mathcal{N}_1$ $\mathcal{N}_2$ $\mathcal{N}_1$ 4	$3 \times 3,\ 3 \times 3,\ 3 \times 3,\ 3 \times 3$
G_2	20 $\mathcal{N}_1$ $\mathcal{N}_2$ $\mathcal{N}_1$ 4	$5 \times 5,\ 3 \times 3,\ 3 \times 3,\ 5 \times 5$
G_1	20 $\mathcal{N}_1$ $\mathcal{N}_2$ $\mathcal{N}_2$ $\mathcal{N}_1$ 4	$5 \times 5,\ 3 \times 3,\ 3 \times 3,\ 3 \times 3,\ 5 \times 5$
G_0	20 $\mathcal{N}_1$ $\mathcal{N}_2$ $\mathcal{N}_2$ $\mathcal{N}_1$ 4	$7 \times 7,\ 5 \times 5,\ 5 \times 5,\ 5 \times 5,\ 7 \times 7$

GM_{20}		
Generative CNN	Numbers of feature maps	Kernel sizes
G_3	16 $\mathcal{N}_1$ $\mathcal{N}_2$ $\mathcal{N}_1$ 4	$3 \times 3,\ 3 \times 3,\ 3 \times 3,\ 3 \times 3$
G_2	20 $\mathcal{N}_1$ $\mathcal{N}_2$ $\mathcal{N}_1$ 4	$5 \times 5,\ 3 \times 3,\ 3 \times 3,\ 5 \times 5$
G_1	20 $\mathcal{N}_1$ $\mathcal{N}_2$ $\mathcal{N}_3$ $\mathcal{N}_2$ $\mathcal{N}_1$ 4	$5 \times 5,\ 3 \times 3,\ 3 \times 3,\ 3 \times 3,\ 3 \times 3,\ 5 \times 5$
G_0	20 $\mathcal{N}_1$ $\mathcal{N}_2$ $\mathcal{N}_3$ $\mathcal{N}_2$ $\mathcal{N}_1$ 4	$7 \times 7,\ 5 \times 5,\ 5 \times 5,\ 5 \times 5,\ 5 \times 5,\ 7 \times 7$

Number sets		
N_{32}	N_{64}	N_{128}
$\mathcal{N}_1 = 32,\ \mathcal{N}_2 = 64,\ \mathcal{N}_3 = 128,$	$\mathcal{N}_1 = 64,\ \mathcal{N}_2 = 128,\ \mathcal{N}_3 = 256$	$\mathcal{N}_1 = 128,\ \mathcal{N}_2 = 256,\ \mathcal{N}_3 = 512$

TABLE 4. Configurations (GM_{16}, GM_{18} and GM_{20}) and number sets (N_{32}, N_{64} and N_{128}) of generator models used in the parameter study.

The present loss functions and error functions for conservation of mass and momentum are not identical to the original forms of conservation laws, but are formulated using the triangle inequality. Therefore, the minimization of the present physical loss functions satisfies conservation of mass and momentum more strictly. In fact, smaller errors are calculated using the original forms of conservation laws, while the errors behave similarly to L_c and L_{mom} as a function of δt.

Appendix C. Parameter study

C.1. *Effects of numbers of layers and feature maps*

Errors as a function of the number of convolution layers of the generator model are calculated by training three generator models with configurations of GM_{16}, GM_{18} and GM_{20} with the number set of N_{128} (see table 4 for these configurations), while errors as a function of the number of feature maps of the generator model in multi-scale CNNs are calculated by training the generator model with number sets N_{32}, N_{64} and N_{128} with the configuration of GM_{20}. All networks are trained with flow fields at $Re_D = 300$ and 500. Magnitudes of errors in configurations considered in the present study are found not to be reduced monotonically with the increase of numbers of layers and feature maps. The configuration with the largest number of convolution layers (GM_{20}) tends to show smaller L_2 and L_∞ errors, while showing L_c and L_{mom}
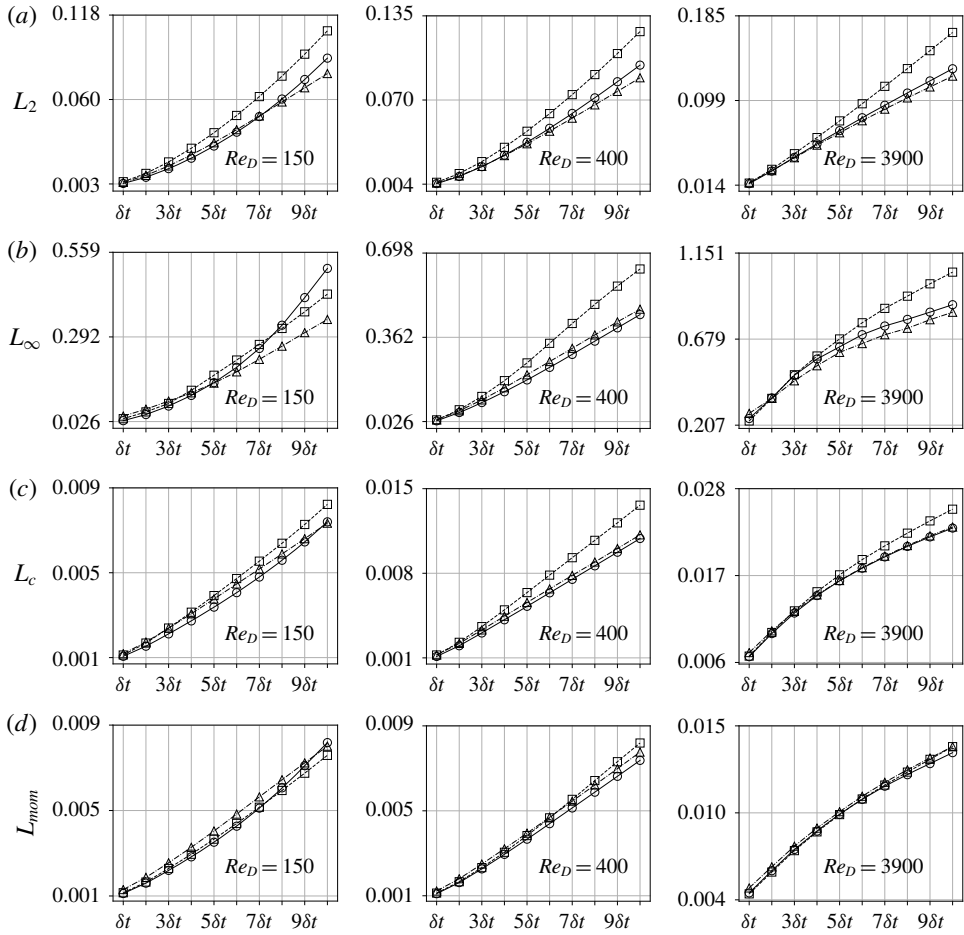
FIGURE 19. Configuration dependence of the generator model. (*a*) L_2, (*b*) L_∞, (*c*) L_c and (*d*) L_{mom} errors from the multi-scale CNN without physical loss functions as a function of recursive prediction steps δt, where $\delta t = 20\Delta t U_\infty / D = 0.1$. Symbol $\bigcirc$ and solid line denote errors from GM_{16}; $\square$ and dashed line denote errors from GM_{18}; $\triangle$ and dash-dotted line denote errors from GM_{20}.

errors of magnitudes similar to or smaller than those in configurations with smaller numbers of convolution layers (GM_{16} and GM_{18}) (figure 19).

The generator model with the largest number set N_{128} tends to show smaller errors (except for the L_{mom} error at $Re_D = 150$) on recursive prediction steps compared to smaller number set models (N_{32} and N_{64}) (figure 20). Therefore, the present study utilizes generator models with the configuration of GM_{20} and with the number set of N_{128}.

Figure 21 shows variations of L_2, L_∞, L_c and L_{mom} errors as a function of training iteration number for the multi-scale CNN without physical loss functions. All errors are found to converge without overfitting.
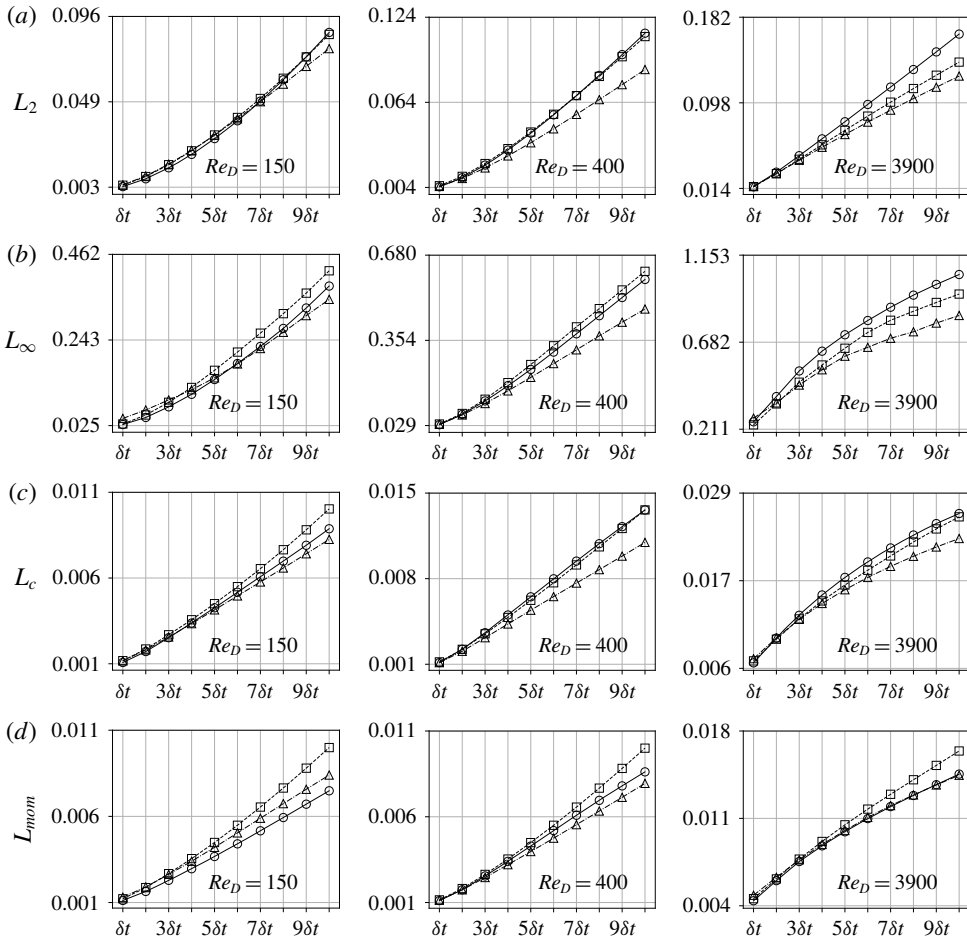
FIGURE 20. Number set dependence of (*a*) L_2, (*b*) L_∞, (*c*) L_c and (*d*) L_{mom} errors from the multi-scale CNN without physical loss functions as a function of recursive prediction steps δt, where $\delta t = 20\Delta t U_\infty/D = 0.1$. Symbol $\bigcirc$ and solid line denote errors from N_{32}; $\square$ and dashed line denote errors from N_{64}; $\triangle$ and dash-dotted line denote errors from N_{128}.
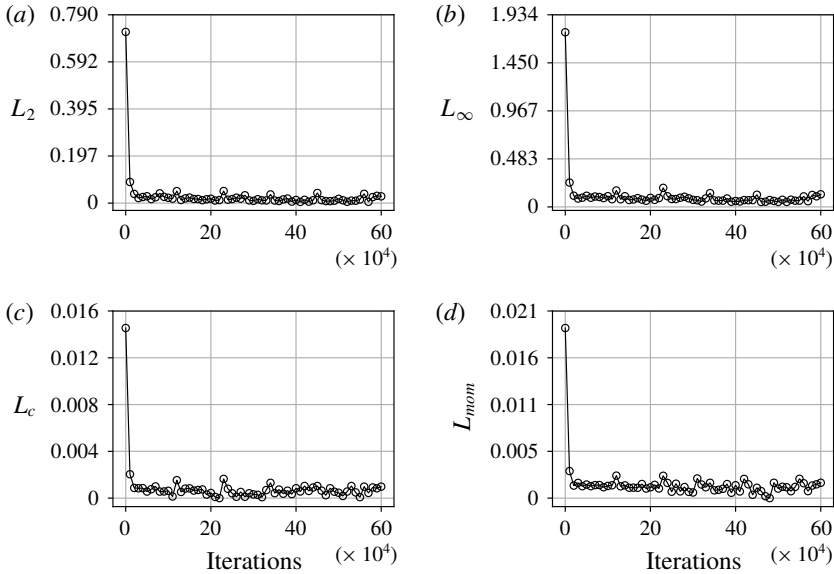
FIGURE 21. Errors from the multi-scale CNN without physical loss functions as a function of the number of training iterations. The network is trained with flow fields at $Re_D = 300$ and 500. The errors are evaluated for flow predictions at $Re_D = 400$.

C.2. *Effects of λ_{adv}*

Errors L_2, L_∞, L_c and L_{mom} from the GAN without physical loss functions using different adversarial training coefficients ($\lambda_{adv} = 0$, 0.05, 0.10, 0.15) are compared in figure 22. For the present parameter study, λ_{l2} and λ_{gdl} are fixed to 1 and λ_{phy} is fixed to 0. The GAN is trained with flow fields at $Re_D = 300$ and 500 and tested on flow fields at $Re_D = 150$, 400 and 3900. The value of λ_{adv} of 0.10 is selected for the present analysis in the results section because that case shows small L_∞ errors at all Reynolds numbers and the smallest L_2, L_c and L_{mom} errors at $Re_D = 3900$.

C.3. *Effects of λ_{phy}*

Errors L_2, L_∞, L_c, and L_{mom} from the multi-scale CNN with physical loss functions using different coefficients ($\lambda_{phy} = 0$, 0.10, 0.50, 1.00) are compared in figure 23. The values of λ_{l2} and λ_{gdl} are fixed to 1 and λ_{adv} is fixed to 0. The multi-scale CNN is trained with flow fields at $Re_D = 300$ and 500 and tested on flow fields at $Re_D = 150$, 400 and 3900. The value of λ_{phy} of 1.00 has been selected for the analysis in the results section because it shows relatively small L_c and L_{mom} errors at all Reynolds numbers (see figure 23).
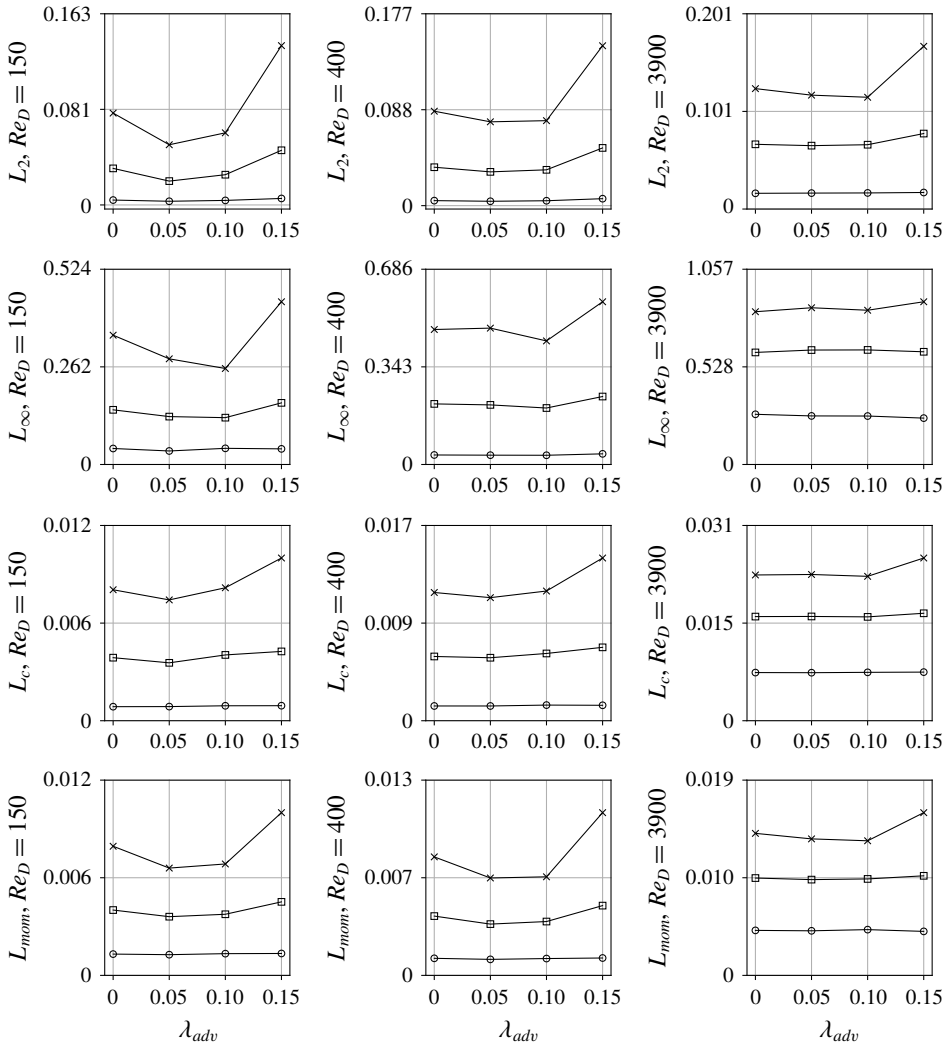
FIGURE 22. Errors as a function of λ_{adv}: $\bigcirc$, $\square$, and $\times$ indicate errors after $1\delta t$, $4\delta t$ and $10\delta t$, respectively.
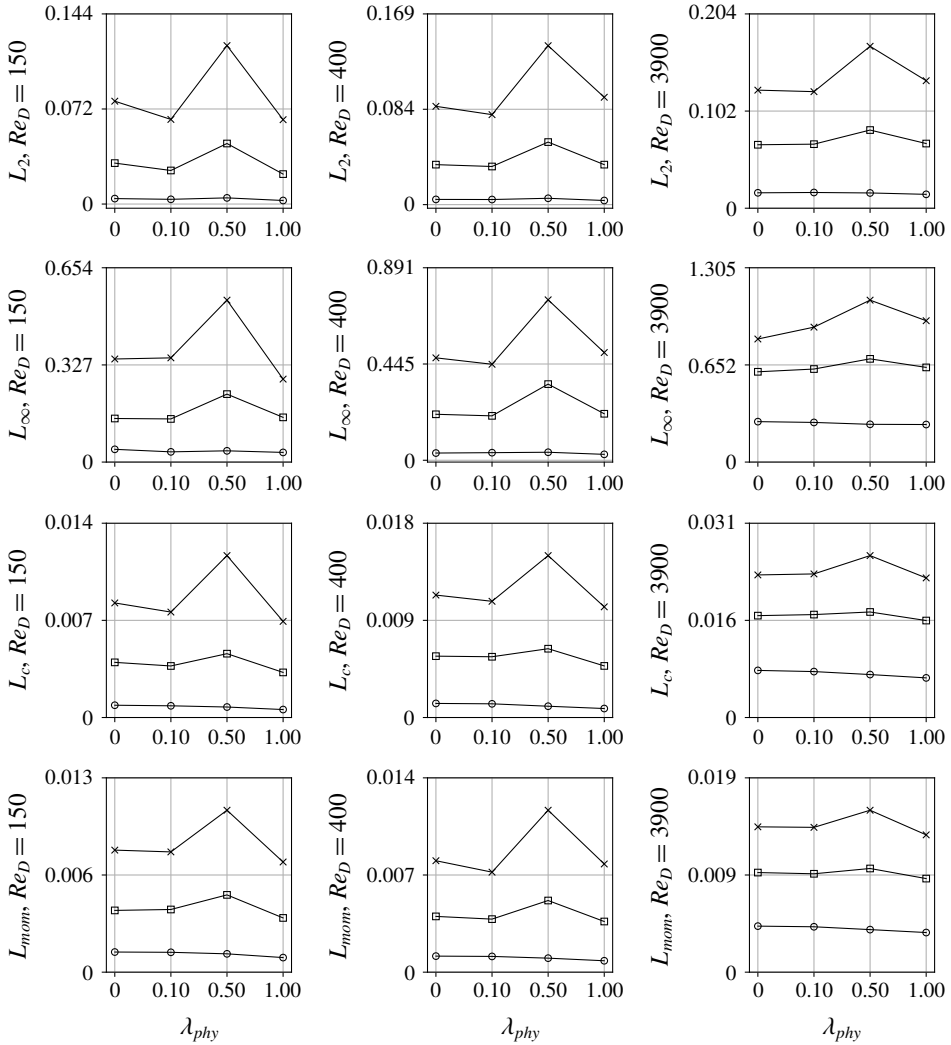
FIGURE 23. Errors as a function of λ_{phy}: $\bigcirc$, $\square$, and $\times$ indicate errors after $1\delta t$, $4\delta t$ and $10\delta t$, respectively.

Appendix D. Flow fields predicted by the GAN trained with a small time-step interval

Contour plots of the cross-stream velocity, the spanwise velocity and the pressure predicted by the GAN at $Re_D = 3900$ with prediction-step intervals of $1\delta t$ are shown in figures 24–27.
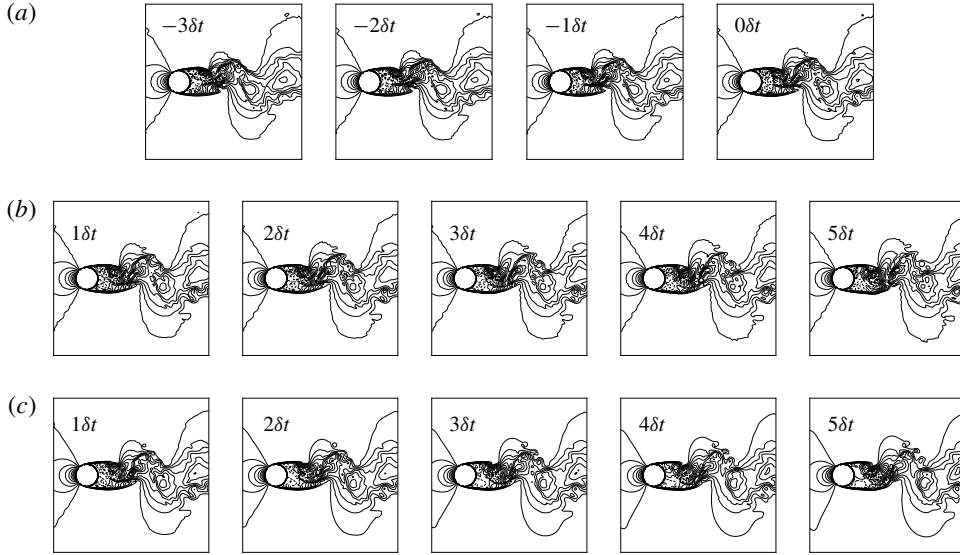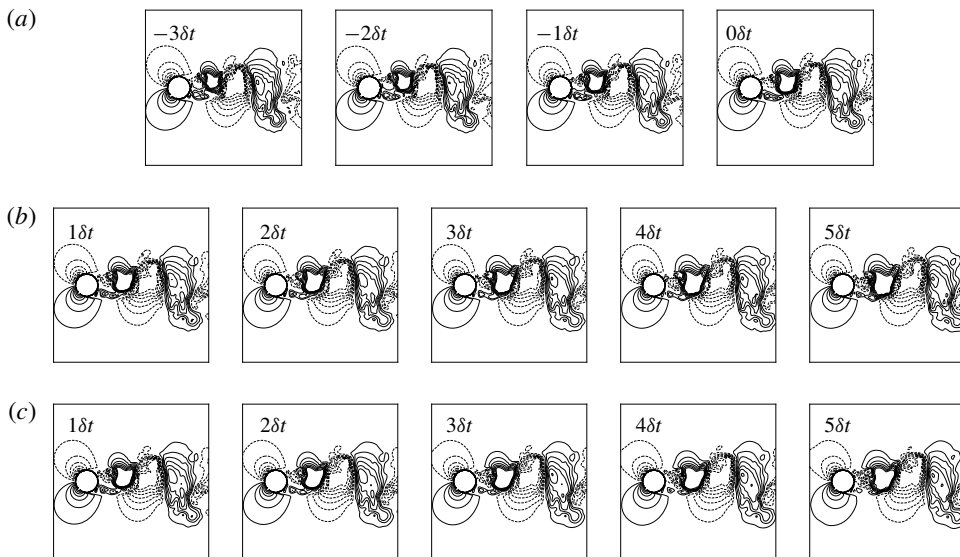
FIGURE 24. Contour plots of the streamwise velocity (u/U_∞) at $Re_D = 3900$ after $1\delta t$, $2\delta t$, $3\delta t$, $4\delta t$ and $5\delta t$, where $1\delta t = 20\Delta t U_\infty / D = 0.1$. Flow fields at $2\delta t$, $3\delta t$, $4\delta t$ and $5\delta t$ are recursively predicted (utilizing flow fields predicted from prior time steps as part of the input). (*a*) Input set, (*b*) ground truth flow fields and (*c*) flow fields predicted by the GAN. 20 contour levels from -0.5 to 1.0 are shown. Solid lines and dashed lines indicate positive and negative contour levels, respectively.
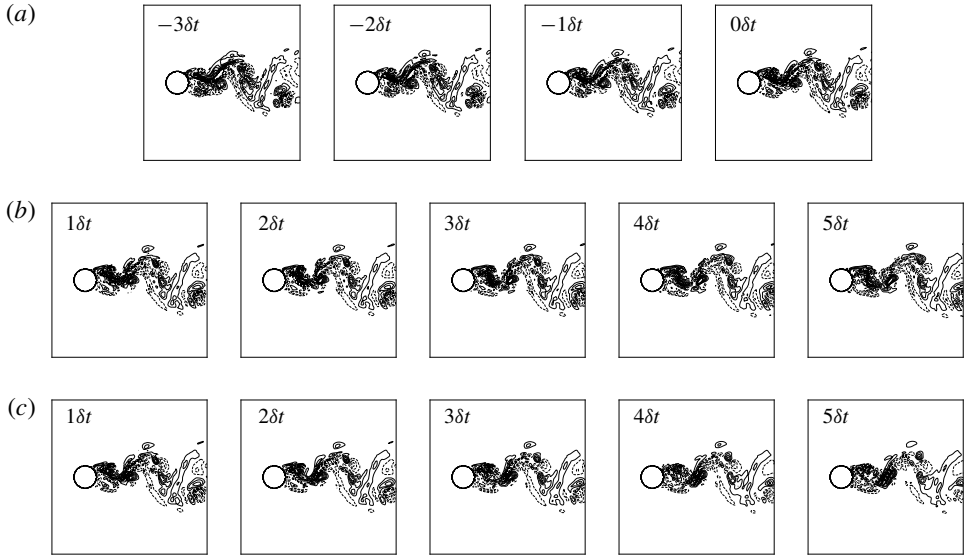

FIGURE 25. Contour plots of the cross-stream velocity (v/U_∞) at $Re_D = 3900$ after $1\delta t$, $2\delta t$, $3\delta t$, $4\delta t$ and $5\delta t$, where $1\delta t = 20\Delta t U_\infty / D = 0.1$. Flow fields at $2\delta t$, $3\delta t$, $4\delta t$ and $5\delta t$ are recursively predicted (utilizing flow fields predicted from prior time steps as parts of the input). (*a*) Input set, (*b*) ground truth flow fields and (*c*) flow fields predicted by the GAN. 20 contour levels from -0.7 to 0.7 are shown. Solid lines and dashed lines indicate positive and negative contour levels, respectively.

FIGURE 26. Contour plots of the spanwise velocity (w/U_∞) at $Re_D = 3900$ after $1\delta t$, $2\delta t$, $3\delta t$, $4\delta t$ and $5\delta t$, where $1\delta t = 20\Delta t U_\infty/D = 0.1$. Flow fields at $2\delta t$, $3\delta t$, $4\delta t$ and $5\delta t$ are recursively predicted (utilizing flow fields predicted from prior time steps as part of the input). (*a*) Input set, (*b*) ground truth flow fields and (*c*) flow fields predicted by the GAN. 20 contour levels from -0.5 to 0.5 are shown. Solid lines and dashed lines indicate positive and negative contour levels, respectively.
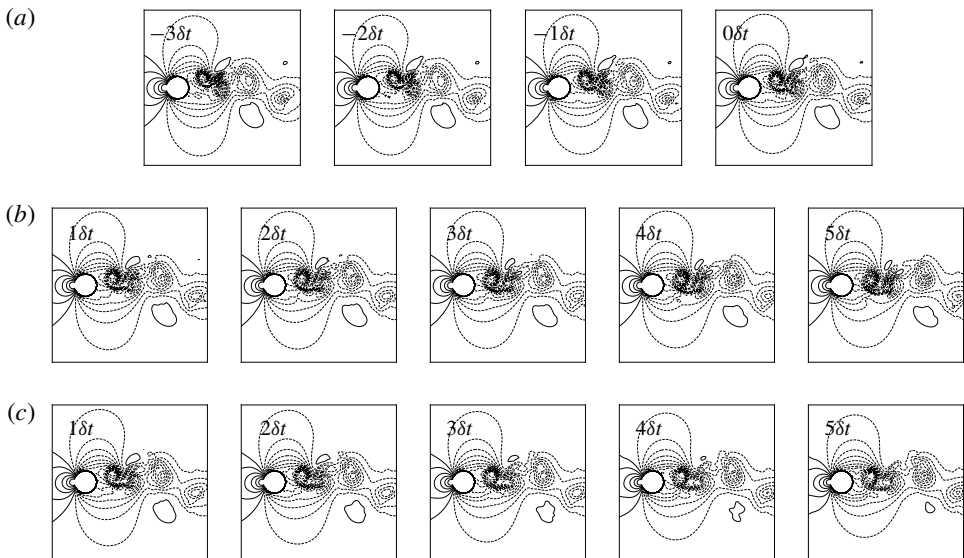

FIGURE 27. Contour plots of the pressure ($p/\rho U_\infty^2$) at $Re_D = 3900$ after $1\delta t$, $2\delta t$, $3\delta t$, $4\delta t$ and $5\delta t$, where $1\delta t = 20\Delta t U_\infty/D = 0.1$. Flow fields at $2\delta t$, $3\delta t$, $4\delta t$ and $5\delta t$ are recursively predicted (utilizing flow fields predicted from prior time steps as part of the input). (*a*) Input set, (*b*) ground truth flow fields and (*c*) flow fields predicted by the GAN. 20 contour levels from -1.0 to 0.4 are shown. Solid lines and dashed lines indicate positive and negative contour levels, respectively.

Appendix E. Flow fields predicted by the GAN trained with a large time-step interval

Contour plots of the cross-stream velocity, the spanwise velocity and the pressure predicted by the GAN at $Re_D = 3900$ with prediction-step intervals of $25\delta t$ are shown in figures 28–30 (see figure 18 for contour plots of the streamwise velocity).
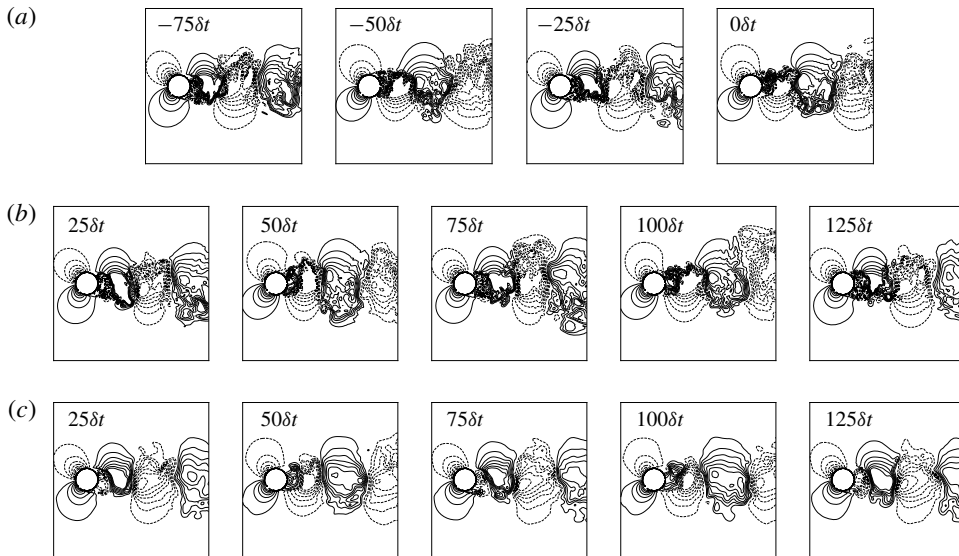

FIGURE 28. Contour plots of the cross-stream velocity (v/U_∞) at $Re_D = 3900$ after $25\delta t$, $50\delta t$, $75\delta t$, $100\delta t$ and $125\delta t$, where $1\delta t = 20\Delta t U_\infty/D = 0.1$. Flow fields at $50\delta t$, $75\delta t$, $100\delta t$ and $125\delta t$ are recursively predicted (utilizing flow fields predicted from prior time steps as part of the input). (*a*) Input set, (*b*) ground truth flow fields and (*c*) flow fields predicted by the GAN. 20 contour levels from -0.7 to 0.7 are shown. Solid lines and dashed lines indicate positive and negative contour levels, respectively.
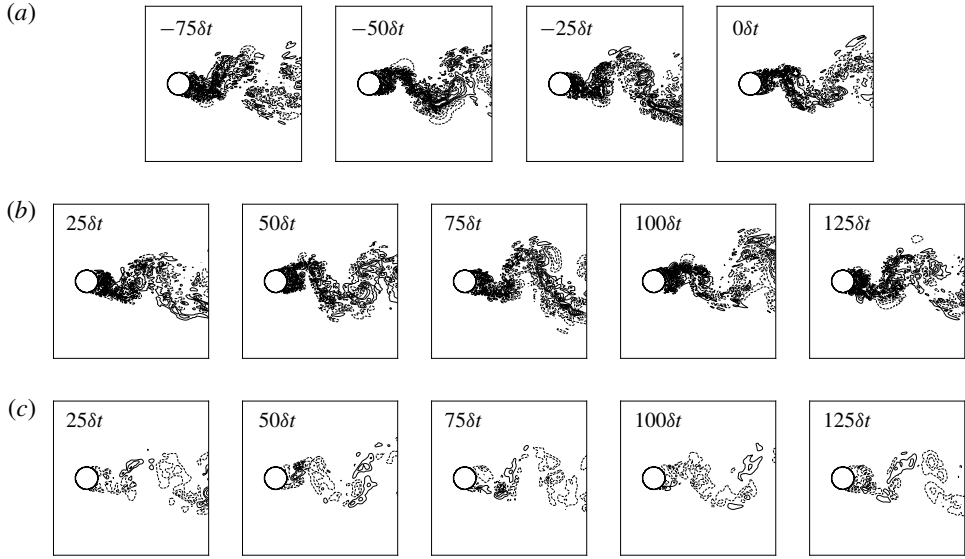
FIGURE 29. Contour plots of the spanwise velocity (w/U_∞) at $Re_D = 3900$ after $25\delta t$, $50\delta t$, $75\delta t$, $100\delta t$ and $125\delta t$, where $1\delta t = 20\Delta t U_\infty/D = 0.1$. Flow fields at $50\delta t$, $75\delta t$, $100\delta t$ and $125\delta t$ are recursively predicted (utilizing flow fields predicted from prior time steps as part of the input). (*a*) Input set, (*b*) ground truth flow fields and (*c*) flow fields predicted by the GAN. 20 contour levels from -0.5 to 0.5 are shown. Solid lines and dashed lines indicate positive and negative contour levels, respectively.
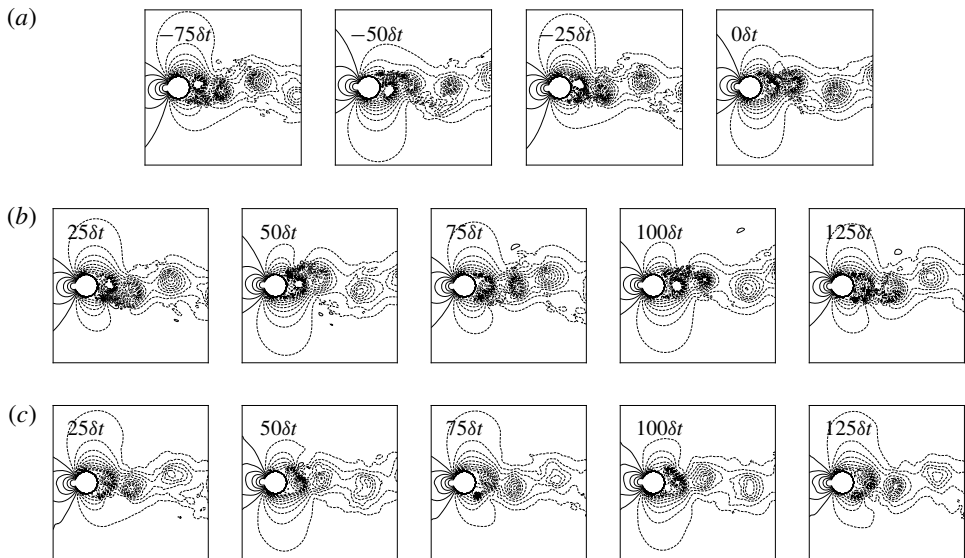

FIGURE 30. Contour plots of the pressure ($p/\rho U_\infty^2$) at $Re_D = 3900$ after $25\delta t$, $50\delta t$, $75\delta t$, $100\delta t$, and $125\delta t$, where $1\delta t = 20\Delta t U_\infty/D = 0.1$. Flow fields at $50\delta t$, $75\delta t$, $100\delta t$ and $125\delta t$ are recursively predicted (utilizing flow fields predicted from prior time steps as part of the input). (*a*) Input set, (*b*) ground truth flow fields and (*c*) flow fields predicted by the GAN. 20 contour levels from -1.0 to 0.4 are shown. Solid lines and dashed lines indicate positive and negative contour levels, respectively.

REFERENCES

BABUCKE, A., KLOKER, M. & RIST, U. 2008 DNS of a plane mixing layer for the investigation of sound generation mechanisms. *Comput. Fluids* **37** (4), 360–368.

BAGHERI, S. 2013 Koopman-mode decomposition of the cylinder wake. *J. Fluid Mech.* **726**, 596–623.

BERGER, E. & WILLE, R. 1972 Periodic flow phenomena. *Annu. Rev. Fluid Mech.* **4** (1), 313–340.

DALIN, P., PERTSEV, N., FRANDSEN, S., HANSEN, O.-F., ANDERSEN, H., DUBIETIS, A. & BALCIUNAS, R. 2010 A case study of the evolution of a Kelvin–Helmholtz wave and turbulence in noctilucent clouds. *J. Atmos. Sol.-Terr. Phys.* **72** (14), 1129–1138.

DENTON, E. L., CHINTALA, S. & FERGUS, R. 2015 Deep generative image models using a Laplacian pyramid of adversarial networks. In *Advances in Neural Information Processing Systems*, pp. 1486–1494. Curran Associates.

FREYMUTH, P. 1966 On transition in a separated laminar boundary layer. *J. Fluid Mech.* **25** (04), 683–704.

GOODFELLOW, I., POUGET-ABADIE, J., MIRZA, M., XU, B., WARDE-FARLEY, D., OZAIR, S., COURVILLE, A. & BENGIO, Y. 2014 Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2672–2680. Curran Associates.

GUO, X., LI, W. & IORIO, F. 2016 Convolutional neural networks for steady flow approximation. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 481–490. ACM.

KINGMA, D. P. & BA, J. 2014 Adam: a method for stochastic optimization. Preprint, arXiv:1412.6980.

KRIZHEVSKY, A., SUTSKEVER, I. & HINTON, G. E. 2012 Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pp. 1097–1105. Curran Associates.

LIBERGE, E. & HAMDOUNI, A. 2010 Reduced order modelling method via proper orthogonal decomposition (pod) for flow around an oscillating cylinder. *J. Fluids Struct.* **26** (2), 292–311.

LING, J., KURZAWSKI, A. & TEMPLETON, J. 2016 Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *J. Fluid Mech.* **807**, 155–166.

MAO, X., JOSHI, V., MIYANAWALA, T. P. & JAIMAN, R. K. 2018 Data-driven computing with convolutional neural networks for two-phase flows: Application to wave-structure interaction. In *ASME 2018 37th International Conference on Ocean, Offshore and Arctic Engineering*, p. V002T08A005. American Society of Mechanical Engineers.

MARCUS, P. S. 1988 Numerical simulation of Jupiter's great red spot. *Nature* **331** (6158), 693–696.

MATHIEU, M., COUPRIE, C. & LECUN, Y. 2015 Deep multi-scale video prediction beyond mean square error. Preprint, arXiv:1511.05440.

MEZIĆ, I. 2013 Analysis of fluid flows via spectral properties of the Koopman operator. *Annu. Rev. Fluid Mech.* **45**, 357–378.

MIYANAWALA, T. P. & JAIMAN, R. K. 2017 An efficient deep learning technique for the Navier-Stokes equations: Application to unsteady wake flow dynamics. Preprint, arXiv:1710.09099.

MIYANAWALA, T. P. & JAIMAN, R. K. 2018 A novel deep learning method for the predictions of current forces on bluff bodies. In *ASME 2018 37th International Conference on Ocean, Offshore and Arctic Engineering*, p. V002T08A003. American Society of Mechanical Engineers.

MORTON, J., JAMESON, A., KOCHENDERFER, M. J. & WITHERDEN, F. 2018 Deep dynamical modeling and control of unsteady fluid flows. In *Advances in Neural Information Processing Systems*, pp. 9258–9268. Curran Associates.

VAN DEN OORD, A., KALCHBRENNER, N., ESPEHOLT, L., VINYALS, O. & GRAVES, A. 2016 Conditional image generation with PixelCNN decoders. In *Advances in Neural Information Processing Systems*, pp. 4790–4798. Curran Associates.

VAN DEN OORD, A., KALCHBRENNER, N. & KAVUKCUOGLU, K. 2016*b* Pixel recurrent neural networks. In *Proceedings of the 33rd International Conference on Machine Learning, 20–22 June* (ed. M. Florina Balcan & K. Q. Weinberger), pp. 1747–1756. PMLR.

RADFORD, A., METZ, L. & CHINTALA, S. 2015 Unsupervised representation learning with deep convolutional generative adversarial networks. Preprint, arXiv:1511.06434.

RANZATO, M., SZLAM, A., BRUNA, J., MATHIEU, M., COLLOBERT, R. & CHOPRA, S. 2014 Video (language) modeling: a baseline for generative models of natural videos. Preprint, arXiv:1412.6604.

RUDERICH, R. & FERNHOLZ, H. H. 1986 An experimental investigation of a turbulent shear flow with separation, reverse flow, and reattachment. *J. Fluid Mech.* **163**, 283–322.

SCHMID, P. J. 2010 Dynamic mode decomposition of numerical and experimental data. *J. Fluid Mech.* **656**, 5–28.

SINGH, A. P., MEDIDA, S. & DURAISAMY, K. 2017 Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils. *AIAA J.* **55** (7), 1–13.

SIROVICH, L. 1987 Turbulence and the dynamics of coherent structures part I. Coherent structures. *Q. Appl. Maths* **45** (3), 561–571.

SOOMRO, K., ZAMIR, A. R. & SHAH, M. 2012 Ucf101: A dataset of 101 human actions classes from videos in the wild. Preprint, arXiv:1212.0402.

SRIVASTAVA, N., MANSIMOV, E. & SALAKHUDINOV, R. 2015 Unsupervised learning of video representations using LSTMs. In *International Conference on Machine Learning*, pp. 843–852. PMLR.

TENNEKES, H. & LUMLEY, J. L. 1972 *A First Course in Turbulence*. MIT Press.

TRACEY, B., DURAISAMY, K. & ALONSO, J. 2015 A machine learning strategy to assist turbulence model development. *AIAA Paper* 2015-1287.

WILSON, A. C., ROELOFS, R., STERN, M., SREBRO, N. & RECHT, B. 2017 The marginal value of adaptive gradient methods in machine learning. In *Advances in Neural Information Processing Systems*, pp. 4148–4158. Curran Associates.

WU, T. Y. 2011 Fish swimming and bird/insect flight. *Annu. Rev. Fluid Mech.* **43**, 25–58.

WU, X. & MOIN, P. 2009 Direct numerical simulation of turbulence in a nominally zero-pressure-gradient flat-plate boundary layer. *J. Fluid Mech.* **630**, 5–41.

XINGJIAN, S., CHEN, Z., WANG, H., YEUNG, D. Y., WONG, W. K. & WOO, W. C. 2015 Convolutional LSTM network: a machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems*, pp. 802–810. Curran Associates.

YONEHARA, Y., GOTO, Y., YODA, K., WATANUKI, Y., YOUNG, L. C., WEIMERSKIRCH, H., BOST, C.-A. & SATO, K. 2016 Flight paths of seabirds soaring over the ocean surface enable measurement of fine-scale wind speed and direction. *Proc. Natl Acad. Sci. USA* **113** (32), 9039–9044.

YOU, D., HAM, F. & MOIN, P. 2008 Discrete conservation principles in large-eddy simulation with application to separation control over an airfoil. *Phys. Fluids* **20** (10), 101515.

ZHANG, Z. J. & DURAISAMY, K. 2015 Machine learning methods for data-driven turbulence modeling. *AIAA Paper* 2015-2460.