機械学習を用いた円柱周り流れのレイノルズ数依存性の予測

Prediction of the Reynolds number dependency of flow around a circular cylinder using machine learning

○ 長谷川 一登，慶大, 横浜市港北区日吉 3-14-1
深見 開，慶大院, 横浜市港北区日吉 3-14-1
村田 高彬，慶大院, 横浜市港北区日吉 3-14-1
深潟 康二，慶大, 横浜市港北区日吉 3-14-1, E-mail：fukagata@mech.keio.ac.jp
Kazuto HASEGAWA, Keio University, Hiyoshi 3-14-1, Kohoku-ku, Yokohama 223-8522
Kai FUKAMI, Keio University, Hiyoshi 3-14-1, Kohoku-ku, Yokohama 223-8522
Takaaki MURATA, Keio University, Hiyoshi 3-14-1, Kohoku-ku, Yokohama 223-8522
Koji FUKAGATA, Keio University, Hiyoshi 3-14-1, Kohoku-ku, Yokohama 223-8522

We present the machine-learned reduced order model (ML-ROM) for predicting the Reynolds number dependency of two-dimensional flow around a circular cylinder. Convolutional Neural Network (CNN) autoencoder is used to compress the flow field, and Long Short Term Memory (LSTM) is employed to predict the temporal evolution of compressed flow field. Training/validation and test datasets are made by numerical simulation at $\mathrm{Re}_D = 20-160$. Predicted flow field shows good agreement with numerical simulation. Furthermore, it can be seen that ML-ROM can compress and predict the flow field successfully against unknown Re_D datasets. The current study suggests the possibility of ML-ROM for various situations in fluid dynamics.

1. 背景及び目的

高い自由度と強い非線形性を持つ非定常の流れ場に対して，制御や現象理解を目的とした Reduced Order Model (ROM) の需要があり，その方法として，Proper Orthogonal Decomposition[1] や Dynamic Mode Decomposition[2] などの手法が知られている．しかし，既存の手法は線形理論にとどまっており，今後非線形現象のモデル化は必須の課題となるであろう．

近年，機械学習は流体力学における強力なツールとして知られ始めてきた [3]．Ling et al.[4] は Neural Network (NN) を用いて Reynolds-averaged Navier-Stokes (RANS) シミュレーションにおけるレイノルズ応力の予測を行い，乱流モデルへの機械学習の適用可能性を示した．また，Gamahara & Hattori[5] は Large-eddy simulation (LES) における SGS モデルを NN を用いて提案した．さらに，Yilmaz & German[6]，Zhang et al.[7] は Convolutional Neural Network (CNN)[8] を用いて翼周りの流れ場より，翼性能の評価を行った．このように非線形性を加味できる機械学習と流体力学の親和性に注目が集まり始めている．

以上の背景から，本研究では，機械学習を用いて ROM を作成し，レイノルズ数依存性の予測を行う．まず，数値シミュレーションを用いて，対象となる流れ場である 2 次元円柱周り流れのデータを作成した．次に，オートエンコーダ型 [9] の CNN (CNN autoencoder) を用いて流れ場の次元圧縮を行い，さらに，Long Short Term Memory (LSTM)[10] を用い，圧縮された流れ場の時間発展を予測した．モデルに数種類のレイノルズ数の流れ場を学習させることで，学習していない未知のレイノルズ数の流れ場の時間発展の予測を行った．

2. 理論

2.1 訓練データ

本研究では，2 次元円柱周り流れを予測の対象とした．訓練データは，2 次元直接数値シミュレーション (DNS) を用いて作成した．支配方程式は連続の式及びナビエ・ストークス方程式

$$\nabla \cdot \boldsymbol{u} = 0 \tag{1}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} = -\nabla \cdot (\boldsymbol{u}\boldsymbol{u}) - \nabla p + \frac{1}{\mathrm{Re}_D}\nabla^2 \boldsymbol{u} \tag{2}$$

である．支配方程式中の全ての物理量は動粘性係数 ν^*，主流速度 U^*_∞，円柱直径 D^* により無次元化されている．なお，$(\cdot)^*$ は有次元数を表す．本研究では，レイノルズ数 Re_D を

$$\mathrm{Re}_D = \frac{U^*_\infty D^*}{\nu^*} = 20, 40, 60, 80, 100, 120, 140, 160 \tag{3}$$

とし，シミュレーションを行った．Figure 1 に計算領域を示す．計算領域は無次元化された代表長さ D を用いて $(L_x \times L_y) = (25.6D \times 20.0D)$，格子数は $(N_x \times N_y) = (1024 \times 800)$，時間刻み幅は $\Delta t = 2.5 \times 10^{-3}$ である．円柱表面での境界条件を適応する方法として，埋め込み境界法 [11] を用いた．

本研究では，DNS によって計算された流れ場より空間方向・時間方向に一部を抽出することで訓練データとした．空間方向には Fig. 1 に示された赤線で囲まれた領域を，時間方向には訓練データの時間刻み幅が $\Delta t' = 2.5 \times 10^{-1}$ となるよう抽出した．なお，抽出された流れ場の格子数は (384×192) である．抽出した流れ場の速度・圧力 (u, v, p) を本研究での訓練データ $(384 \times 192 \times 3)$ とした．

2.2 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN)[7] は，画像認識の分野で幅広く用いられている手法であり，データ形状を加味できるというメリットから近年流体力学分野でも用いられ始めている [6],[12]．

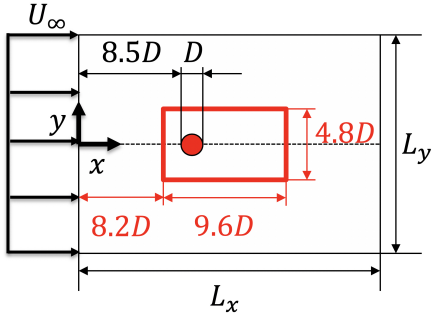

Fig. 1: Computational domain.

CNN は畳み込み層とプーリング層の 2 種類の層を組み合わせることによって形成される．畳み込み層とは，畳み込み演算

$$u_{ijm} = \sum_{k=0}^{K-1} \sum_{p=0}^{H-1} \sum_{q=0}^{H-1} x_{i+p,j+q,k} h_{pqkm} + b_m \qquad (4)$$

を行う層である．畳み込み層に入力される $(W \times W \times K)$ のデータに対してデータ点をインデックス (i, j, m) で表し，点 (i, j, m) における値を x_{ijm} と表す．h_{pqkm} は，畳み込み層における m 個の $(H \times H \times K)$ のフィルタの値を入力データと同様にして表し，b_m はフィルタごとに存在するバイアスを表す．u_{ijm} は畳み込み演算により出力される 2 次元データの (i, j, m) 成分である．Fig. 2 に畳み込み演算のバイアスを省略した概略図を示す．畳み込み層の出力 z_{ijm} は，

$$z_{ijm} = f(u_{ijm}) \qquad (5)$$

と表され，畳み込み演算によって得た u_{ijm} に対して活性化関数 $f(\cdot)$ を施したデータである．Figure 3 に畳み込み層での演算の概略図を示す．プーリング層とは，入力されたデータに対して拡大・収縮を行う層である．プーリングには，最大プーリングや平均プーリングなどいくつかの方法が存在する．

2.3 Long Short Term Memory (LSTM)

Long Short Term Memory (LSTM)[9] は時系列情報を保持することのできるネットワークである．その特性から自然言語処理の分野で幅広く活用されている．本研究では流れ場の時間発展の予測を行うため，時系列問題を扱える LSTM は最適なネットワークであると言える．

LSTM は時系列情報を保つためにメモリセルを持つ．また，時系列情報の更新のために入力ゲート，出力ゲート及び忘却ゲートの 3 つのゲートを持つ．添字 t を時間の指標として，LSTM に外部から入力 x_t，前の状態のセルから出力 h_{t-1}，メモリセル情報 C_{t-1} が入力された際，それぞれのゲートに対して，

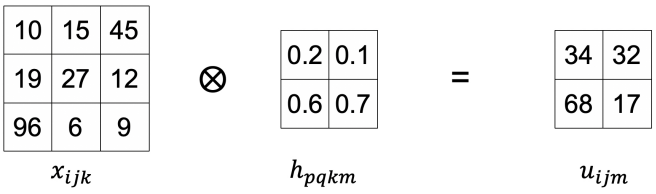

Fig. 2: Convolutional operation.

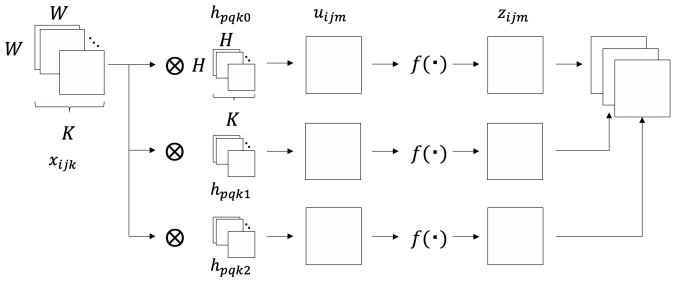

Fig. 3: Convolutional layer.

$$\begin{aligned} f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) && (6)\\ i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) && (7)\\ \widetilde{C}_t &= \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) && (8)\\ C_t &= f_t \times C_{t-1} + i_t \times \widetilde{C}_t && (9)\\ o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) && (10)\\ h_t &= o_t \times \tanh(C_t) && (11) \end{aligned}$$

と計算される．$f, i, o, C, \widetilde{C}, h$ はそれぞれ，忘却ゲートの出力，入力ゲートの出力，出力ゲートの出力，メモリセル情報，メモリセルに追加する情報及び，LSTM の出力である．W, b はそれぞれのゲートでの重みとバイアスであり，添字は存在するゲートもしくはセルを示す．また，式中の σ はシグモイド関数を示す．

3. ネットワークの構成及び訓練手法

本研究では，まず，CNN autoencoder を用いて流れ場を圧縮し，LSTM を用いて CNN autoencoder によって圧縮された流れ場の時間発展の予測を行う．

3.1 CNN autoencoder

CNN autoencoder は，学習の際，入力と出力に同じデータを与える．また，入力と出力の間でデータの次元が小さくなる構造を持つため，与えられたデータを圧縮することが可能である．Figure 4 に CNN autoencoder の概略図を示す．本研究では，3 つのスケールのフィルタを持つ Multi-scale CNN[13] を用いて，CNN autoencoder を構築した．1 つのスケールの CNN の圧縮部の構造を Tab. 1 に示す．フィルタサイズは各々 $3 \times 3, 5 \times 5$ 及び 9×9 とした.CNN autoencoder は入出力に $(384 \times 192 \times 3)$ の形状を持ち，内部で $(6 \times 3 \times 4)$ まで圧縮する構造を持つ．訓練データとして，$\mathrm{Re}_D = 40, 80, 120, 160$ の瞬時場をそれぞれ 3000 枚ずつ，計 12000 枚を学習に用いた．これらのデータのうち, 70 % を Training data，30 % を Validation data として学習に用いた．

また誤差関数には, Mean Squared Error (MSE) を用いた．MSE (E_{CNN}) は，

$$E_{\mathrm{CNN}} = \frac{1}{N_x}\frac{1}{N_y}\frac{1}{N_\phi} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \sum_{k=1}^{N_\phi} (u_{\mathrm{true},ijk} - u_{\mathrm{deco},ijk})^2 \qquad (12)$$

として計算される．ここに，$\boldsymbol{u}_{\mathrm{true}}, \boldsymbol{u}_{\mathrm{deco}}$ はそれぞれ DNS によって計算された流れ場，CNN autoencoder によって出力された流れ場である．これらは，いずれも x 方向成分に N_x, y 方向成分に N_y ，物理量の種類を表す指標として N_ϕ の次元を持つ．本研究においては $(N_x, N_y, N_\phi) = (384, 192, 3)$ である．

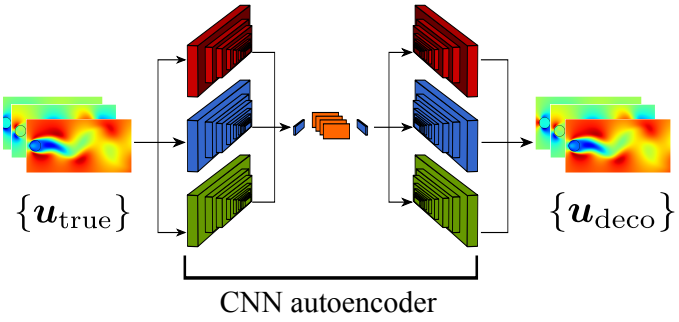

Fig. 4: Schematic of CNN autoencoder : Filter size of CNN are $(3 \times 3), (5 \times 5)$ and (9×9).

Tab. 1: Structure of CNN autoencoder.

Layer	Output shape	Activation function
Input Layer	(384, 192, 3)	-
1st Conv.	(384, 192, 16)	ReLU
1st Max Pooling	(192, 96, 16)	-
2nd Conv.	(192, 96, 8)	ReLU
2nd Max Pooling	(96, 48, 8)	-
3rd Conv.	(96, 48, 8)	ReLU
3rd Max Pooling	(48, 24, 8)	-
4th Conv.	(48, 24, 8)	ReLU
4th Max Pooling	(24, 12, 8)	-
5th Conv.	(24, 12, 8)	ReLU
5th Max Pooling	(24, 12, 8)	-
6th Conv.	(12, 6, 4)	ReLU
6th Max Pooling	(6, 3, 4)	-
Add Layer	(6, 3, 4)	-
7th Conv.	(6, 3, 4)	ReLU
Encoded Layer	(6, 3, 4)	-

3.2 LSTM

LSTM は，CNN autoencoder によって圧縮された圧縮場の時間発展を予測する役割を持つ．学習は，時刻 $t = (n-4)\Delta t', (n-3)\Delta t', (n-2)\Delta t', (n-1)\Delta t', n\Delta t'$ における圧縮場を入力とし，時刻 $t = (n+1)\Delta t'$ における圧縮場を出力として行った．本研究における LSTM の概略図を Fig. 5 に示す．訓練データとして，$\mathrm{Re}_D = 40, 60, 80, 120, 160$ の圧縮場，計 5000 枚を学習に用いた．これらのうち，70 % が Training data， 30 % が Validation data である．構造として，入出力に圧縮場の値 (72 個) と，その瞬時場のレイノルズ数の情報 (1 個) の計 73 個の形状を持つ．また，隠れ層において過適合を抑制するため，出力部で Batch Normalization (BN), Dropout (DO)[14] を採用した．構造の詳細を Tab. 2 に示す．また，誤差関数として CNN autoencoder と同様の MSE を用いた．LSTM における MSE (E_{LSTM}) は，

$$E_{\mathrm{LSTM}} = \frac{1}{N_x}\frac{1}{N_y}\frac{1}{N_z}\sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\sum_{k=1}^{N_z}(\tilde{u}_{\mathrm{true},ijk} - \tilde{u}_{\mathrm{pred},ijk})^2 \tag{13}$$

として表せる．ここに，$\tilde{u}_{\mathrm{true}}$ は時刻 $t = (n+1)\Delta t'$ における正しい圧縮場を表し，$\tilde{u}_{\mathrm{pred}}$ は LSTM によって予測された圧縮場を示す．添字の i, j, k はそれぞれ圧縮場の x, y, z 方向成分を表し，N_x, N_y, N_z の次元を持つ．本研究では，$(N_x, N_y, N_z) = (6, 3, 4)$ である．

3.3 Machine-learned reduced order model

本研究では，学習済みの CNN autoencoder と LSTM を組み合わせることによって Machine-learned reduced order model (ML-ROM) を構築する．ML-ROM の概略図を Fig. 6 に示す．まず，CNN autoencoder を流れ場を圧縮する部分 (CNN encoder) と拡大する部分 (CNN decoder) に分割する．DNS によって計算された初期場を CNN encoder に入力し，初期場を圧縮する．得られた圧縮場を LSTM 入力することによって，次のタイムステップでの圧縮場を得る．予測された圧縮場を再度 LSTM に入力することによって繰り返し予測を行い，圧縮場の時間発展を得る．得られた圧縮場を CNN decoder に入力することで，流れ場の時間発展を得る．

4. 結果

4.1 CNN autoencoder

二次元円柱周りの $\mathrm{Re}_D = 40, 80, 120, 160$ における流れ場を学習した CNN autoencoder に対し $\mathrm{Re}_D = 20, 40, 60, 80, 100, 120, 140, 160$ の流れ場を入力することで評価を行った．Figure 7 に CNN autoencoder の結果の一例として入力した $\mathrm{Re}_D = 100$ の速度・圧力場と，出力された速度・圧力場を示す．Figure 7 より，CNN autoencoder を用いることで流れ場が適切に圧縮できていることが確認された．学習済みの CNN autoencoder を用いて，各レイノルズ数について 2000 枚の瞬時場を入力した際の MSE を Fig. 8 に示す．それぞれの値は，各瞬時場について得られた MSE について平均を取ったものであり，エラーバーは標準偏差の 2 倍を表す．Figure 8 より，全てのレイノルズ数において学習が正しく行われていることが確認できた．また，学習に用いたレイノルズ数の流れ場に対し，用いていない流れ場の MSE が大きい傾向が確認できる．特に，$\mathrm{Re}_D = 20, 60$ は他と比べて MSE が大きい．$\mathrm{Re}_D = 20$ で MSE が予測がうまく行われなかった原因として，$\mathrm{Re}_D = 20, 40$ の流れ場には渦放出がないということが挙げられる．CNN autoencoder は渦放出のない流れ場を $\mathrm{Re}_D = 40$ のみしか学習していないため，$\mathrm{Re}_D = 20$ が入力された際，学習した $\mathrm{Re}_D = 40$ の流れ場に影響されてしまうと考えられる．つまり，CNN autoencoder は渦放出のない流れ場についての予測性能を持っていないと考えられる．これを回避するためには，渦放出のない流れ場をさらに学習する必要があると考えられる．また，$\mathrm{Re}_D = 60$ については 40 と 80 の間で渦放出がない流れ場の構造から，渦放出のある構造へ遷移するため，他の流れ場と比べて学習するのが難しいと考えられる．

Tab. 2: Structure of LSTM.

Layer	Output shape	Activation function
Input Layer	(5, 73)	-
1st LSTM	(5, 128)	tanh
1st BN	(5, 128)	-
1st DO	(5, 128)	-
2nd LSTM	(5, 128)	tanh
2nd BN	(5, 128)	-
2nd DO	(5, 128)	-
3rd LSTM	(128)	tanh
Output Layer	(73)	-

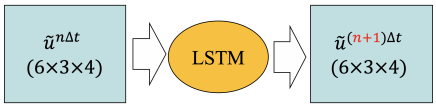

Fig. 5: Schematic of LSTM.

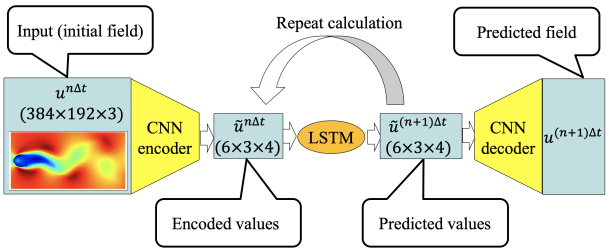

Fig. 6: Schematic of ML-ROM.

4.2 LSTM

学習済みの CNN autoencoder を用いて圧縮した $\mathrm{Re}_D = 40, 60, 80, 120, 160$ における圧縮場の時間発展を学

習した LSTM を $\mathrm{Re}_D = 20, 40, 60, 80, 100, 120, 140, 160$ の圧縮場を用いて評価を行った．LSTM が予測した圧縮場と正しい次のタイムステップの圧縮場の MSE を Fig. 9 に示す．それぞれの値は各レイノルズ数について 2000 枚の圧縮場を入力した際の MSE の平均であり，エラーバーは標準偏差の 2 倍を表す．Figure 9 より，全てのレイノルズ数において学習が正しく行われていることが確認できた．CNN autoencoder と同様に，学習に用いたレイノルズ数の流れ場に対して，用いていない流れ場の MSE がわずかに大きい傾向が確認できる．$\mathrm{Re}_D = 20$ に関して，CNN autoencoder と同様，渦放出がない，かつ LSTM は渦放出のない流れ場の圧縮場を $\mathrm{Re}_D = 40$ の圧縮場のみしか学習していないため，他のレイノルズ数の MSE より大きいと考えられる．これを回避するためには，渦放出のない流れ場の圧縮場をさらに学習する必要があると考えられる．

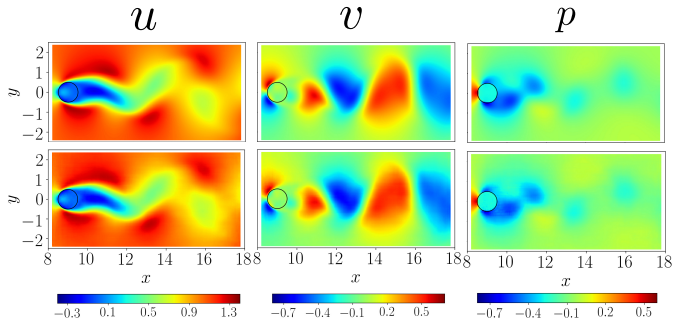

Fig. 7: Results of CNN autoencoder ($\mathrm{Re}_D = 100$) : The true DNS flow field are shown in upper line. The decoded flow field are shown in lower line. The left, middle and right are u, v and p respectively.

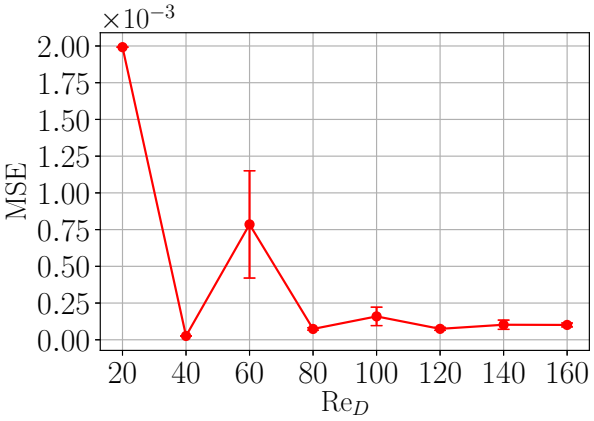

Fig. 8: MSE in each Re_D of CNN autoencoder.

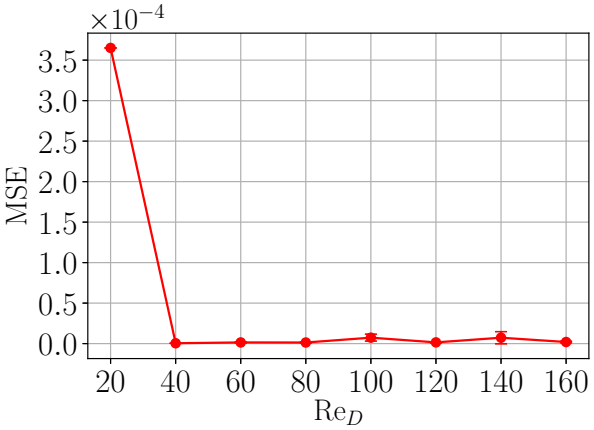

Fig. 9: MSE in each Re_D of LSTM.

4.3 ML-ROM

$\mathrm{Re}_D = 20, 40, 60, 80, 100, 120, 140, 160$ の流れ場において ML-ROM を用いて時間発展の予測を行った．まず，それぞれのレイノルズ数において DNS によって計算された初期場を CNN autoencoder に入力し初期場の圧縮場を得た．さらに，その圧縮場を LSTM に入力し，得た出力を繰り返し LSTM に入力することで予測を行った．CNN autoencoder，LSTM どちらの学習にも用いていない $\mathrm{Re}_D = 100, 140$ の流れ場の時間発展を予測した結果から計算した円柱後方での平均主流方向速度分布を Fig. 10 に示す．Figure 10 から，ML-ROM によって予測した流れ場の主流平均速度分布が DNS と一致していることが確認できる．このことから，ML-ROM は学習に用いていないレイノルズ数の流れ場の時間発展も予測できていることが確認できる．また，それぞれのレイノルズ数において，ML-ROM によって予測された流れ場からストローハル数 (St)，抗力係数 (C_D) を計算した結果を Fig. 11，Fig. 12 にそれぞれ示す．Figure 11，12 においてエラーバーは，5 個の LSTM のモデルを作成し，それぞれのモデルを用いて数値を計算した際の標準偏差の 2 倍である．ストローハル数について，DNS とほぼ一致していることが確認できた．また，抗力係数については，$\mathrm{Re}_D = 20, 60$ について良い結果が得られておらず，これは，DNS と差が大きい CNN autoencoder の予測精度に起因していると考えられる．その他のレイノルズ数については，DNS に近い値を示しており，ML-ROM が流れ場を予測できていることが確認できた．

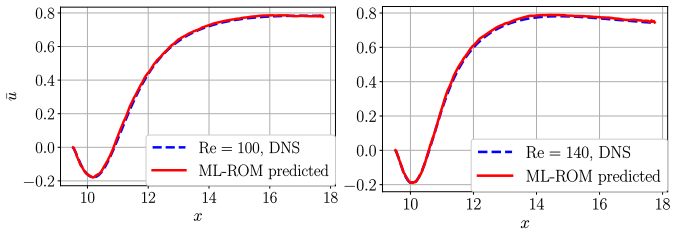

Fig. 10: Mean streamwise velocity on the centerline : Left and right are $\mathrm{Re}_D = 100$ and $\mathrm{Re}_D = 140$ respectively.

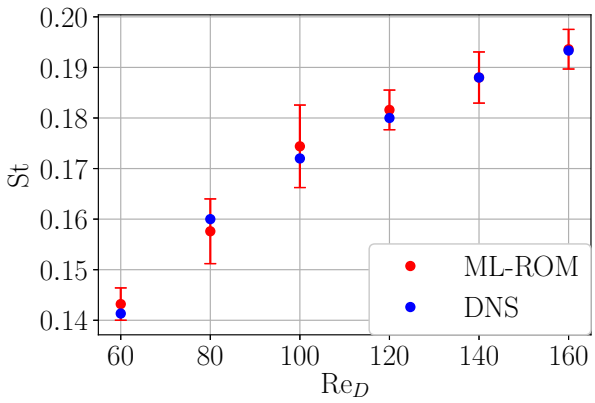

Fig. 11: Strouhal number.

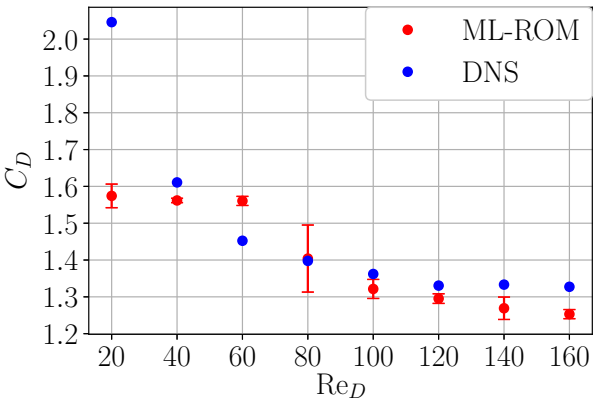

Fig. 12: Drag coefficient.

5. 結論

本研究では，機械学習を用いた縮約モデル (ML-ROM) を提案した．具体的には，まず Convolutional Neural Network autoencoder を用いて流れ場の次元削減を行い，Long Short Term Memory を用いることによって CNN autoencoder によって圧縮された流れ場の時間発展を予測した．提案手法を二次元円柱周り流れに適応し，流れ場の時間発展を予測した．さらに，ML-ROM を学習に用いていないレイノルズ数の流れ場にも適応することで，流れ場のレイノルズ数依存性の予測が可能であることを示した．

参考文献

(1) Lumley, J. L., “The structure of inhomogeneous turbulent flows, In Atmospheric turbulence and wave propagation,” eds. Yaglom, A. M. and Tatarski, V. I., Moscow, Nauka (1967), pp. 166-178.

(2) Schmid，P., “Dynamic mode decomposition of numerical and experimental data,” *J. Fluid Mech.*, Vol. 656 (2010), pp. 5-28.

(3) Kutz, N., “Deep learning in fluid dynamics, ” *J. Fluid Mech.*, Vol. 814 (2017), pp. 1-4

(4) Ling, J., Kurzawski, A. and Templeton, J., “Reynolds averaged turbulence modelling using deep neural networks with embedded invariance,” *J. Fluid Mech.*, Vol. 807 (2016), pp. 155-166.

(5) Gamahara, M. and Hattori, Y., “Searching for turbulence models by artificial neural network,” *Phys. Rev. Fluids*, Vol. 2 (2017), 054604.

(6) Yilmaz, E. and German, B., “A convolutional neural network approach to training predictors for airfoil performance,” 18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference (2017).

(7) Zhang, Y., Sung and W., Mavris, D., “Application of convolutional neural network to predict airfoil lift coefficient,” AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference (2018).

(8) LeCun, Y. A., Bottou, L. , Bengio, Y. and Haffner, P., “Gradient-based learning applied to document recognition,” *Proc. IEEE*, Vol. 86, No. 11 (1998), pp. 2278-2324.

(9) Hinton, G. E., and Salakhutdinov., “Reducing the dimensionality of data with neural networks,” *Science*, Vol. 313 (2006), pp. 504-507

(10) Hochreiter, S. and Schmidhuber, J., “Long short-term memory,” *Neural Comput.*, Vol. 9 (1997), pp. 1735-1780.

(11) Kor, H., Ghomizad, M. and Fukagata, K, “A unified interpolation stencil for ghost-cell immersed boundary method for flow around complex geometries,” *J. Fluid Sci. Technol.*, Vol. 12 (2017), JFST011.

(12) Fukami, K., Kawai, K. and Fukagata, K., “A synthetic turbulent inflow generator using machine learning,” arXiv preprint (2018), arXiv:1806.08903 [physics.flu-dyn]

(13) Du, X., Qu, X., He, Y. and Guo, D., “Single image super-resolution based on multi-scale competitive convolutional neural network,” *Sensors*, Vol. 18, No. 789, pp. 1-17.

(14) Srivastava, N., Hinton, G., Krizhevsky, A. Sutskever, I. and Salakhutdinov, R., “Dropout: a simple way to prevent neural networks from overfitting,” *J. Mach. Learn. Res.*, Vol. 15 (2014), pp. 1929-1958