Fast Flow Field Estimation for Various Applications with A Universally Applicable Machine Learning Concept

Michael Leer¹ · Andreas Kempf¹

Received: 6 May 2020 / Accepted: 23 November 2020 / Published online: 11 December 2020
© The Author(s) 2020

Abstract

This paper presents an approach for the prediction of incompressible laminar steady flow fields over various geometry types. In conventional approaches of computational fluid dynamics (CFD), flow fields are obtained by solving model equations on computational grids, which is in general computationally expensive. Based on the ability of neural networks to intuitively identify and approximate nonlinear physical relationships, the proposed method makes it possible to eliminate the explicit implementation of model equations such as the Navier–Stokes equations. Moreover, it operates without iteration or spatial discretization of the flow problem. The method is based on the combination of a minimalistic multilayer perceptron (MLP) architecture and a radial-logarithmic filter mask (RLF). The RLF acts as a preprocessing step and its purpose is the spatial encoding of the flow guiding geometry into a compressed form, that can be effectively interpreted by the MLP. The concept is applied on internal flows as well as on external flows (e.g. airfoils and car shapes). In the first step, datasets of flow fields are generated using a CFD-code. Subsequently the neural networks are trained on defined portions of these datasets. Finally, the trained neural networks are applied on the remaining unknown geometries and the prediction accuracy is evaluated. Dataset generation, neural network implementation and evaluation are carried out in MATLAB. To ensure reproducibility of the results presented here, the trained neural networks and sample applications are made available for free download and testing.

Keywords Machine learning · Neural networks · MLP · CFD · Flow estimation

1 Introduction

As computing capabilities increase, machine learning (ML) continues to open further areas and offers revolutionary opportunities. A widespread concept of ML are artificial neural networks, which offer suitable solutions for highly complex problems. These include physical phenomena with high-nonlinearities, where mathematical algorithms derived from the human understanding of scientific laws reach their limits. For this

✉ Andreas Kempf
andreas.kempf@uni-due.de

¹ Chair of Fluid Dynamics, IVG, Duisburg-Essen University, Carl Benz Straße 199, Duisburg, Germany

reason, neural networks and ML in general receive increasing attention in the field of fluid dynamics. Advances in turbulence modeling, the acceleration of classical CFD solution methods as well as the development of completely new solution methods based on the approximation capabilities of neural networks are conceivable. The combination of ML and fluid dynamics offers great potential for simulation and fundamental understanding of fluid flow. Yet in order to benefit from ML on a large scale, adapted concepts must first be developed that consider the unique physical conditions of fluid dynamics.

In recent years, many studies have been conducted in which machine learning has been applied to turbulence modeling, reflecting the increased attention given to ML in the fluid mechanics community. Ling et al. (2016) proposed to use deep neural networks for Reynolds averaged Navier–Stokes (RANS) models. Their turbulence modeling approach embedded Galilean invariance into the network architecture using a higher-order multiplicative layer. In other works (Tracey et al. 2013; Edeling et al. 2014; Ling and Templeton 2015) machine learning techniques were used to improve RANS models, by addressing turbulence uncertainties. Neural networks were applied in the area of large eddy simulations (LES) (Wang et al. 2018; Maulik et al. 2018). Fukami et al. (2019) utilized ML to generate synthetic turbulent inflow.

Besides turbulence modeling, recent studies tried to enhance the numerical simulation of flow by maintaining the CFD approach and applying machine learning on computationally intensive, delimited steps. Tompson et al. (2017) succeeded to leverage the approximation power of deep learning with the precision of CFD solvers. A deep neural network solves the Poisson equation in order to accelerate the euler equation velocity update in a classical CFD approach. ML-based pressure projection was also proposed by Yang et al. (2016). Wiewel et al. (2019) proposed a novel long-short-term memory network (LSTM) based approach to predict the changes of pressure fields in fluid flows over time.

In addition to the work mentioned so far, steps have also been taken towards completely new solution methods independent of the CFD approach, with the aim of predicting the flow field through neural networks. ML-based flow approximation models are increasingly establishing themselves in the field of computer graphics and animation, where fast and realistic flow field solutions are required. Recent contributions are the utilization of generative networks (Xie et al. 2018; Kim et al. 2019), mainly considering smoke reconstruction. More related to this work, some authors have tried to develop machine learning concepts, which we refer to as geometry-to-flow-mapping. Guo et al. (2016) proposed to use a neural network to learn a mapping from boundary conditions and geometry to steady-state flow. They were able to show that a signed distance-from-wall function can be better processed by the neural network than a binary wall function. A signed distance function was utilized to predict flow around airfoils (Bhatnagar et al. 2019). U-net architectures were applied on randomly generated 2D-shapes and airfoils, respectively (Chen et al. 2019; Thuerey et al. 2020). A common feature of the mentioned geometry-to-flow mapping approaches is that they learn on entire flow fields of defined lateral extent. This makes the input and output layer of the neural network rigid regarding the domain size. We refer to this prediction process as *field-to-field scheme*. Once trained, neural networks are limited to predicting flow fields of the same lateral extent, which largely remain within the same physical problem, e.g. flow around airfoils. In order to obtain more accurate prediction near the airfoil boundary, Sekar et al. (2019) trained a neural network to predict the flow field point by point. Their network was based on two steps: a CNN to extract geometrical parameters from airfoil shapes and a multilayer perceptron (MLP) for point-wise

prediction of the flow field from these parameters. With this setup a high prediction accuracy was reached. However, due to the field-wise input to the neural network, the method cannot be applied to physical problems other than airfoil flow without major adjustments.

The present paper introduces a new ML method for fast fluid flow estimation by geometry-to-flow-mapping using a *point-to-point scheme*. This method can be used for any type of geometry and offers high prediction accuracy and good generalization capability even with simple neural networks and small datasets. Also, it eliminates the explicit inclusion of model equations such as the Navier-Stokes equations, as well as time and spatial discretization. It consists of two steps: As the first step, we introduce the radial-logarithmic filter mask (RLF). Its purpose is to interpret the geometry by compressing the relevant information into a discrete number of features. In the second step, these features can be effectively processed by a subsequent minimalistic neural network as its input neurons. This network in turn provides the velocity components of the flow field as its output. The whole solution scheme is point-to-point related. This means that the RLF always refers to a single (focused) point in the geometry and the neural network provides the velocity components for this point in the flow field.

The paper is organized as follows Details on the network setup and the functionality of the RLF are given in Sect. 2. The method is tested on three different types of geometries: primitive car shapes, airfoils and internal flows through curved and branched channels. The data preparation required for each of them is explained in Sect. 3 and the prediction results for each geometry type are presented and discussed in Sect. 4. A conclusion in Sect. 5 will discuss the current state of the method based on the achieved prediction accuracy and the practicality. Also, suggestions for future work are given.

2 Network Setup

By training with large datasets, neural networks learn to predict the correct output as a function of the input. In the case of a flow field predictor, the desired output could be for instance a velocity or a pressure field. The definition of the required input, however, is not trivial. A flow field is a function of many physical parameters, of which the flow guiding geometry, due to its arbitrary complexity, is (probably the most) difficult to provide to a neural network as an input. A suitable way must be found to encode the geometry and make the information derived from it available as discrete input features. The number of input features on the other hand has a decisive influence on the size, complexity and manageability of a neural network. Although there exist techniques like convolutional neural networks (CNN) which make the network able to transform a large input field into relevant information through a series of folding and unfolding layers, this study follows the assumption that a neural network is particularly effective in this context when the input information is made available to it in the most compressed yet appropriate form. Each input feature should provide relevant information so that each input neuron is used as effectively as possible. As a suitable way to encode the geometry and make the information derived from it available within a discrete number of features, we introduce the radial-logarithmic filter mask (RLF). Its task is to translate the geometry information into a coding, that can be effectively interpreted by a subsequent neural network.

2.1 Radial-logarithmic Filter Mask

The RLF approach is derived from a set of initial assumptions according to the behavior of steady-state incompressible flow: (1) The flow at a certain point in the field is naturally predetermined by the physical boundary conditions and the surrounding geometry. (2) Any wall in contact with the flow will affect it. (3) Elements of the geometry that are near a considered point in the flow field tend to have a greater influence on the flow at that point than those that are farther away. These three assumptions are not universally valid but can provide a crucial clue to the mapping from geometry to flow. The specification for a geometry-compressing filter can be defined by combining these assumptions: considering a point for which the flow is to be determined, the surrounding geometry is to be compressed in a way, that wall-structures near the point are weighted higher than structures that are farther away, while the overall radius of the treated geometry should include a sufficiently large part of the geometry that stands in contact to the flow. These conditions can be fulfilled by a matrix which consists of cells divided in radial and circumferential direction. For each cell, the ratio of the area occupied by geometry over the total area is determined algorithmically, leading to a continuous cell value within $\{[0,1]\}$. This approach is illustrated by Fig. 1.

In addition, Table 1 shows the input features according to the example in Fig. 1 as they would be submitted to the neural network. “Appendix A” provides further illustration on the RLF.

It always refers to a specific point in the geometry and delivers information according to this point to the subsequent neural network.

The RLF is defined by the number of radial and circumferential layers R and Φ as well as the outer Radius r_R . The cells grow exponentially in radial direction depending on a growth factor g . The limit of each radial layer can be obtained by:

$$r_n = g^{(n-R)} r_R \quad (1)$$

By default a value of 2 is set for g , resulting in a doubling of each next layer in radius. The region is then divided into equal circumferential sections according to Φ . When applying the RLF, it is centered on the investigated point in the domain. For each cell, an entry $W_{r,\phi}$

Fig. 1 RLF operation illustrated on an arbitrary geometry. The binary geometry information is organized as a Cartesian map, where the value 1 refers to solid structures (highlighted in black) and 0 refers to points that are accessible to the flow (in white). The RLF (here: $R = 12$, $\Phi = 24$) is focused on a point in the Geometry. On the right side, the input-array values, $W_{r,\phi}$, as a result of the RLF are shown cell-wise as gray scale ($W_{r,\phi} = 0 \rightarrow \text{white}$; $W_{r,\phi} = 1 \rightarrow \text{darkgray}$), area outside the radius of treated geometry in black

Table 1 Input-array W according to the example in Fig. 1. Geometry representations of this kind can be effectively interpreted by neural networks, as will be shown later

φ	r											
	1	2	3	4	5	6	7	8	9	10	11	12
1	0	0	0	0	0.58	1	1	1	0.03	0	0.63	0.75
2	0	0	0	0	0.32	1	1	1	0.05	0	0.68	0.99
3	0	0	0	0	0	0.74	1	1	0.33	0	0.81	1
4	0	0	0	0	0	0.04	0.64	0.97	0.93	0.21	0.34	0.99
5	0	0	0	0	0	0	0	0.04	0.20	0.28	0.29	0.59
6	0	0	0	0	0	0	0.06	0.38	0.68	0.94	1	1
7	0	0	0	0	0.06	0.58	0.96	1	1	1	1	1
8	0	0	0	0.18	0.91	1	1	1	1	1	1	1
9	0	0	0.02	0.89	1	1	1	1	1	1	1	1
10	0	0	0.46	1	1	1	1	1	1	1	1	1
11	0	0	0.75	1	1	1	1	1	1	1	1	1
12	0	0	0.82	1	1	1	1	1	1	1	1	1
13	0	0	0.64	1	1	1	1	1	1	1	1	1
14	0	0	0.20	0.99	1	1	1	1	1	1	0.99	0.93
15	0	0	0	0.46	0.99	1	1	1	1	1	0.98	0.31
16	0	0	0	0	0.33	0.92	1	1	1	1	1	0.43
17	0	0	0	0	0	0.05	0.44	0.85	1	1	1	0.93
18	0	0	0	0	0	0	0	0	0.24	0.72	1	0.99
19	0	0	0	0	0	0	0	0	0	0	0.52	0.87
20	0	0	0	0	0	0	0	0.16	0.41	0.46	0.18	0.85
21	0	0	0	0	0	0.06	0.67	1	1	1	0.95	0.48
22	0	0	0	0	0.01	0.76	1	1	1	0.76	0.16	0.02
23	0	0	0	0	0.32	1	1	1	0.78	0.03	0	0.50
24	0	0	0	0	0.59	1	1	1	0.23	0	0.38	0.65

The input is by definition $\{[0,1]\}$ -normalized and the continuous transition from flow field areas to solid areas is beneficial

is obtained by dividing the wall-occupied area by the total cell area. All entries combined result in the input-array W , which is passed to the neural network after it has been reshaped into a 1D vector according to a certain pattern.

2.2 Multilayer Perceptron

The sensible compression of the geometry achieved by the RLF enables even simple neural networks to further process the information. An input formulated this way can be processed by a comparatively simple regression learner like the multilayer perceptron (MLP) we use (Rosenblatt 1957; Rumelhart et al. 1995). “Appendix B” provides a brief overview of MLP. For theoretical details on MLP and its training algorithms we refer to Kruse et al. (2016).

In this study, the architecture of the MLP is adjusted differently, depending on the geometric or fluid mechanical complexity of the respective dataset. However, the depth is never set to more than three hidden layers. It is implemented in MATLAB using the

feedforwardnet-functionality (The Mathworks 2019), largely using the default settings and the Levenberg–Marquardt algorithm for training (Levenberg 1944; Marquardt 1963). The initial goal is to train the neural network to predict the flow field at constant inlet and outlet conditions and changing geometry. This prediction refers to specific points in the flow field. This means that by giving the RLF-matrix over a specific point in the geometry, the neural network delivers the velocity components accordingly to that point. In contrast to conventional numerical approaches such as CFD, the result for this point can be determined instantaneously and independent of a total solution of the flow field. If a solution of the entire flow field is desired, this procedure can be repeated for each coordinate point separately. Fig. 2 illustrates the data flow through the neural network. A detailed data flow chart covering the entire methodology is provided in “Appendix C”.

Due to the point-to-point prediction scheme a once trained network is not limited to a certain side length of the geometry-fluid domains but can be used flexibly. This is an important distinguishing feature to field-to-field prediction schemes where a trained neural network is limited to predict flow fields which have the same domain size as the training flow fields (Chen et al. 2019; Guo et al. 2016; Thuerey et al. 2020; Bhatnagar et al. 2019). Furthermore, any type of geometry declaration can in principal be handled by the RLF, e.g. binary graphic, vector graphic or mesh data.

3 Data Generation

Thorough testing of the method requires a workflow for the generation of randomly parameterized geometries, CFD-simulation of associated flow fields and application of the data for the learning or testing stage. Such a workflow can be implemented in MATLAB.

For the training and testing of the neural network concept, three different two-dimensional datasets are generated in this study. The first one contains convex geometries inspired by cars, the second airfoil-like shapes and the third internal flow in curved and branched channels. In each case, the datasets are generated by a random-based geometric parameterization and completed by a subsequent CFD simulation with a MATLAB-based in-house code. To enable a large number of tested geometries, we had to stick to low Reynolds numbers and coarse grids.

3.1 Cars Dataset

In this dataset, geometries in the shape of cars are created and placed in a virtual 2D wind tunnel. This means that there are walls at the top and bottom, with the inlet on the left and

Fig. 2 Data flow chart of the learning task. On the left the RLF input-array (focused on Cartesian point P) of the size $R \cdot \Phi$, corresponding to the number of radial and circumferential layers. On the right the velocity-components

Fig. 3 Cars dataset: visualization of the CFD simulated flow (m/s) around a randomly modeled car

Fig. 4 Selection of randomly parameterized car shapes

the outlet on the right. The wind tunnels size corresponds to 141×75 pixels, with the two first and the two last rows in y-direction forming the tunnel wall. The car bodies themselves are formed by 12 randomly based parameters, resulting in a large variety of possible geometries. Figure 3 shows as an example, a generated geometry with the associated flow field.

In this way, a total of 138 geometries are created. The associated flow fields are then simulated under constant boundary conditions, i.e. constant inlet velocity and constant viscosity. This results in Reynolds numbers between 590 and 680, measured on the random car length. Figure 4 shows 6 exemplary geometries.

To generate the training set for the neural network, a defined number of data points (pairs of input and output) are then extracted from this dataset. In order to create a data point, the RLF is applied to a random pixel in the flow field of a random training geometry. The resulting array of input features is then combined with both 2D-velocity components obtained by CFD. This sequence is repeated until the desired number of data points for the training is collected. In order to have unlearned geometries for a subsequent validation, only a part of the geometries is used for training, separating the others for validation.

Table 2 Airfoil parameter ranges according to the NACA four-digit series definition

Property	Index	Range
Maximum camber $[0.01c]$	p	$\{0, 1, \dots, 6\}$
Axial position of maximum camber $[0.1c]$	m	$\{0, \{2, 3, \dots, 7\}\}$
Maximum thickness $[0.01c]$	xx	$\{06, 07, \dots, 21\}$
Angle of attack $[^\circ]$		$\{-1, 0, \dots, 7\}$

Geometries are defined by four digits, p.m.xx, all related to the chord length c . The chosen parameter ranges allow 784 possible combinations. With the additionally defined angle of attack range, a total of 7056 unique geometries could be created

Fig. 5 Second dataset: selection of generated airfoils. Top: NACA-2412 at 0° AoA, middle: NACA-3414 at 3° AoA, bottom: NACA-4721 at 7° AoA

3.2 Airfoil Dataset

The second dataset is generated similar to the previous one but with parametrized airfoils instead of cars. The walls above and below are replaced by periodic boundaries. The airfoils are modeled according to the 4-digit NACA series (Jacobs et al. 1933). In addition, the angle of attack (AoA) is varied. The exact value-ranges of the NACA parameters and the AoA as well as the resulting possible number of variants are summarized in Table 2.

The chord length is adjusted to a constant value of 50 pixels, so that at constant inflow conditions the Reynolds number (Re) stays the same. The Reynolds number is set to $Re_c = 125$. A total of 727 geometries are created by randomly combining the parameters and placed in a 203×53 pixels domain. Flow fields are then simulated with CFD. Figure 5 shows three exemplary geometries. (Note the low spatial resolution that was required for this study.)

In order to later test the suitability of a trained neural network especially regarding variation of the AoA and extrapolation capability beyond the training dataset bandwidth, a series of CFD simulations is carried out for the profile NACA-2412, in which AoA reaches from -4° to 11° in 0.5° steps.

3.3 Internal Flow Dataset

To proof the neural network concept on internal flows, a set of curved channels is generated using the GeoPhi (unpublished) geometry optimization code. The Internal Flow dataset consists of two fundamentally different geometry types, 28 curved flow channel geometries (Type I) and 45 branched flow channel geometries (Type II). For Type I, a single inlet and a single outlet of 21 pixels width each are placed at random positions in the first and last row of the initial 123×123 pixels geometry field, respectively. GeoPhi, whose optimization goal is to achieve a geometry with the lowest possible pressure loss, is then used to form the curved channel. In this way a set of 28 geometries is formed. Type II is generated analogously, just replacing the 21 pixels wide inlet and outlet with 3 randomly placed 11 pixels wide openings in the first and in the last geometry row each. Flow fields are then simulated analogous to the other two datasets by setting constant viscosity and inlet velocity. Measured by the wall distance at the inlet, the Reynolds-number is 190 for type 1 and is correspondingly smaller for type 2, or larger if several randomly generated branches merge.

Figure 6 shows 2 exemplary geometries of type I and 4 exemplary geometries of type II. These geometries are inspired by haemodynamics, but configurations of this kind can also be found in process engineering, internal flows in reactors, combustion chambers or similar applications.

Fig. 6 Third dataset: selection of geometries generated for the Internal flow dataset. With the Type I geometries (a) and (d) and the Type II geometries (b), (c), (e), (f)

4 Results and Discussion

The neural network implementation and evaluation are carried out in MATLAB using the *feedforwardnet*-functionality. First, as indicated in the previous section, a certain number of data points are extracted from the training datasets using the RLF. The number of extracted data points plays a decisive role for the prediction accuracy and generalization capability of the trained networks. Its analysis is therefore described separately in Sect. 4.1.1. Secondly, MLP's are trained for each geometry type (car shapes, airfoils, curved and branched channels) using the extracted data points. Finally, the trained neural networks are evaluated by comparing their prediction of the flow field for the validation geometries with the CFD reference.

For each MLP, the Levenberg Marquardt algorithm is set as the training function and mean squared error (MSE) as the loss function. The mean squared error can be defined as follows:

$$L = \frac{1}{2N} \sum_{m=1}^N (\hat{\mathbf{v}}_m - \mathbf{v}_m)^2 \quad (2)$$

where m denotes a data point, N is the total number of learned data points per batch, and $\hat{\mathbf{v}}_m$ and $\mathbf{v}_m$ are the CFD reference and the ML-approximated velocity vectors at point m , respectively.

To evaluate the accuracy of a trained neural network, the average relative error (ARE), is used (Guo et al. 2016; Ahmed and Qin 2009). First, the difference between the network-approximated and the CFD solved flow field is computed. Then for a pixel, the relative error is defined as the ratio of the magnitude of the difference to the CFD solved flow field. For 2D flows, the relative error at pixel (i, j) is formulated as:

$$err^n(i, j) = \frac{\sqrt{(\hat{v}_{x,ij}(g_n) - v_{x,ij}(g_n))^2 + (\hat{v}_{y,ij}(g_n) - v_{y,ij}(g_n))^2}}{\sqrt{v_{x,ij}(g_n)^2 + v_{y,ij}(g_n)^2}} \quad (3)$$

with the two velocity components v_x and v_y . The average relative error of the n -th data case is the mean value of the errors of all pixels outside the solid geometry, i.e. available for the flow:

$$ARE^n = \frac{\sum_i \sum_j err^n(i, j) \mathbb{1}_{ij}(g_n > 0)}{\sum_i \sum_j \mathbb{1}_{ij}(g_n > 0)} \quad (4)$$

where $\mathbb{1}_{ij}(g_n > 0)$ is 1 if and only if (i, j) is outside the solid geometry of g_n .

4.1 Cars Dataset

From the 138 geometries of the Cars dataset, 120 (87 %) are used for training and the remaining 18 (13 %) are reserved for subsequent validation. The neural network learns from an array of data points. Each data point consists of a defined number of input features and a defined number of desired output values. The input features of a data point are created by evaluating the RLF for a random pixel within the flow field of a random training

geometry. The RLF is set as follows: 12 layers in circumferential direction and 4 layers in radial direction with an outer radius of 80 pixels. The total number of input features is therefore $4 \times 12 = 48$. The corresponding output values are the two velocity components at this coordinate, which are obtained from the CFD-simulated velocity field. With a grid study, which is described in detail in the Sect. 4.1.1, 800 can be determined as the ideal number of created data points per geometry. With an average of 7360 pixels within a flow field in the Cars dataset, this corresponds to a ratio of 10.9 %. In other words, the neural network sees an average of 10.9 % of the pixels of a flow field in the training set during training and does not see the remaining 89.1 %. In summary, the training database consists of 800×120 data points with 48 inputs and 2 outputs each. The size of the input layer of an MLP is defined by the number of input features and the output layer is defined by the number of output values. In addition, the architecture of the neural network is supplemented by two hidden layers with 31 neurons each and one hidden layer with 7 neurons. Accordingly, the network architecture configuration is RLF(48)-31-31-7-2, with 3 hidden layers and in total 4 operating layers.

Using the given database, the neural network is trained until the loss function, Eq. (2) is converged, in this case after 27 epochs. The trained neural network can now be applied to new geometries to predict the flow field. For this purpose, an input array is formed by evaluating the RLF for each pixel outside the fixed geometry. The velocity components for each pixel are then predicted by the neural network.

Fig. 7 Visualization of the prediction (NET) compared to the reference solution (CFD) with the absolute error in the right column. On four exemplary validation geometries. Color-map axis-limits are consistent for CFD-result and NET-prediction of each geometry

Figure 7 visualizes the network prediction compared to the CFD-reference on 4 validation geometries (absolute velocity and absolute error).

Except for the separated flow behind the rear edge of the cars, the estimated flow fields show a good agreement with the CFD reference. In order to assess the prediction accuracy quantitatively, the ARE is calculated for each geometry according to Eqs. (3) and (4). It is observed that the neural network is able to predict the flow fields of the validation set with a mean ARE of 40.16 % and reproduce the flow fields of the training set with a mean ARE of 42.78 %.

The results show that errors increase in x-direction and most errors occur behind the vehicle. Comparing the error-plots in Fig. 7 shows, that the prediction accuracy is generally better for flow fields, where the stream leaving the car-body at the bottom stays attached to the ground than for flow fields, where the flow separates. The prediction results show, that the neural network can cope with former but becomes inaccurate for the latter in the prediction of the streams behind the cars. The fact that flow fields with a horizontal stream at the bottom appear in about 80 % of the cases, while this percentage is higher if only seen the 18 validation geometries, might explain, that the validation-set prediction is slightly better than the training-set prediction. Since the accuracy of the neural networks per definition strongly depends on solid geometry, it seems plausible, that the prediction is worse in areas far away from walls. This is especially true for the area behind the cars.

Interestingly, features are observed that may appear to look like turbulence, although the neural network has never been taught about it, and the Reynolds number is far too low.

4.1.1 Grid search on number of data points

It is well known that neural networks are especially effective if they can learn on the basis of large datasets. However, in the domain of fluid dynamics, the generation of large datasets is in most cases very costly. It is not possible to provide a limitless amount of data for learning tasks of any kind. Moreover, a computationally expensive training of neural networks with a large amount of CFD data, which in turn must be provided by computationally expensive simulations, would indirectly contradict the goal of adding neural networks to save time in flow prediction.

Instead, the goal is to achieve a maximum generalization capability of the neural network from a given limited set of geometries and associated flow fields. The number of extracted data points plays a decisive role in this process: The availability of many data points enhances the learning potential of the neural network. But then, a large number of extracted data points from a given number of geometries means that the neural network receives very detailed information about the flow field of each individual geometry which favors memorizing and thereby reduces the capability to generalize. For the point-to-point prediction scheme, the need for a trade off solution becomes foreseeable: On the one hand, the need for as many data points as possible to favor the neural networks learning ability. And on the other hand, avoiding memorizing by using only a small percentage of pixels per geometry.

The goal is therefore to give an approximate recommendation, how large the proportion of the sampled to the total number of pixels should be, using a defined number of training geometries. As indicated in Sect. 4.1, a logarithmic grid search is performed in order to narrow down the optimal number of extracted data points per geometry. One after another, neural networks are trained with an increasing number of data points and evaluated. Fig. 8 shows the result of the logarithmic grid-search.

Fig. 8 Logarithmic grid search on training-volume. On the left scale mean average relative error. On the right scale the coverage, i.e. average ratio of sampled to total number of data points per training geometry

The car geometries contain an average of 7360 pixels inside the flow field. The mean ARE of both, training and validation set show a regressive course until 1000 data points (corresponds to sampled/total pixel ratio of 13.6 %) are reached with an error of 42.6 % for the training and 42.3 % for the validation set at that point. Beginning with 2000 data points, the agreement between training and validation set prediction decreases significantly. While the training set error further decreases to 38 %, the validation set error increases immediately to 50 %. Further on, both curves seem to converge. A theoretical sampled to total pixel ratio of 100 % is reached at 7360, i.e. between 5000 and 10,000.

Based on this logarithmic grid study, a higher resolved grid search in the range of 200 to 1700 data points is performed to further narrow down the optimal number. In this resolved grid study, the optimum number of data points is reached at 800, which corresponds to a sampled to total pixel ratio of 10.9 %.

The evaluation confirms the assumption that an optimal number of extracted data points exists for which the average relative error of the validation geometry prediction and thus the overall network prediction-accuracy achieves the best value.

4.2 Airfoil Dataset

The division into training and validation dataset, the extraction of data points, as well as the training of the neural network are carried out analogous to the Car dataset, as described in Sect. 4.1. Of the total 727 generated NACA airfoil geometries, 650 (90 %) are used for training separating the other 77 geometries (10 %) for subsequent model validation. The RLF parameters are selected as follows: 12 layers in circumferential direction and 4 layers in radial direction with an outer radius of 194 pixels. The total number of input features

Fig. 9 NACA-2214 at 7° AoA flow field prediction. **a** u-CFD, **b** v-CFD, **c** u-prediction, **d** v-prediction, **e** u-error, and **f** v-error

Fig. 10 NACA-4308 at 6° AoA flow field prediction. **a** u-CFD, **b** v-CFD, **c** u-prediction, **d** v-prediction, **e** u-error, and **f** v-error

is therefore $4 \times 12 = 48$. The number of extracted data points per geometry is set to 500. In summary, the training database consists of 500×650 data points with 48 inputs and 2 outputs each. The neural network is equipped with two hidden layers with 47 neurons each and one hidden layer with 13 neurons. Accordingly, the network architecture configuration is RLF(48)-47-47-13-2, with 3 hidden layers and in total 4 operating layers.

The prediction results of the flow field (both velocity components and component error) for two validation geometries (NACA 2214, AoA = 7° and NACA 4308, AoA = 6°) are shown in Figs. 9 and 10, respectively.

The predicted flow fields show a good agreement with the CFD results. The mean ARE of the reproduced training geometry flow fields is 2.61 % and the mean ARE of the predicted validation geometry flow fields is 2.67 %. It can be seen that the deviations between the reference fields and the predicted fields are primarily found in the downstream region, i.e. in the airfoil wake. This indicates that the strong variation of this region across the

Fig. 11 Average relative error distribution of flow prediction. **a** Training set. **b** Validation set

dataset is difficult to capture by the neural network. The ARE distributions of the prediction of training and validation cases are shown in Fig. 11.

Their comparison indicates a good agreement between training and validation prediction and therefore a good generalization capability of the neural network.

In order to test the prediction accuracy regarding aerodynamic properties, the lift coefficient is determined for all predicted flow fields and compared with the lift coefficient determined from the CFD reference flow fields. Figure 12 shows the correlation between both. Lift coefficients are calculated from a surface integral of momentum around a control volume covering the domain.

As described in Sect. 3, an additional dataset was created, which contains the airfoil NACA 2412 in a range of different angles of attack. While the training dataset contains geometries with AoA's between -1° and $+7^\circ$, this additional dataset contains a range between -4° and $+11^\circ$. Figure 13 shows the predicted flow fields for NACA 2412 at five different AoA's while the first and the last geometry are outside the training range of -1° to $+7^\circ$. In addition, Fig. 14 shows the lift coefficient versus AoA for this dataset.

Fig. 12 Correlation of lift coefficients computed from predicted flow fields over lift coefficients computed from CFD reference flow fields shows a high coefficient of determination

Besides an accurate response to angle changes, it shows that the model is capable of slight extrapolation and can be numerically stable beyond the training data parameter range. However, this conclusion has no relation to a physically meaningful extrapolation covering the prediction of stall behavior, which is beyond the scope and ambitions of this paper.

4.3 Internal Flow Dataset

The procedure for the training and testing of the model on the Internal flow dataset is analogous to the previous two datasets. In this section only the key setting parameters and dataset partitioning are described. For a detailed explanation on the data point extraction see Sect. 4.1.

The Internal flow dataset consists of two fundamentally different geometry types Curved flow channels (Type I) and branched flow channels (Type II). A single neural network is trained using both types together. In the partitioning of this dataset, it therefore must be ensured, that the proportion of the two types is set equally between training and validation set. The training set is composed of 23 of the 28 generated Type I geometries and 37 of the 45 generated Type II geometries (both 82 % of the total number of geometries of each type). Thus, the training set contains a total of 60 geometries. The validation set is composed of 5 of the 28 generated Type I geometries and 8 of the 45 generated Type II geometries (both 18 % of the total number of geometries of each type). Hence, the validation set contains a total of 13 geometries and the ratio of the overall training set to the overall validation set is 60 to 13 (82 % to 18 %).

Fig. 13 Predicted flow field for NACA-2412 for different AoA's. -3.0° and $+9.0^\circ$ AoA are outside the training data bandwidth

Fig. 14 CL versus AoA. Within training bandwidth (*1) and outside (*2)

Fig. 15 Internal flow field prediction for a Type I geometry. **a** u-reference, **b** u-prediction, **c** u-error, **d** v-reference, **e** v-prediction, and **f** v-error

Fig. 16 Internal flow field prediction for a Type II geometry. **a** u-reference, **b** u-prediction, **c** u-error, **d** v-reference, **e** v-prediction, and **f** v-error

The RLF parameters are selected similarly to the previous datasets: 12 layers in circumferential direction and 4 layers in radial direction with an outer radius of 104 pixels. The total number of input features per data point is therefore $4 \times 12 = 48$. The number of extracted data points per geometry is set to 500. In summary, the training database consists of 500×60 data points with 48 inputs and 2 outputs (2 velocity components to be predicted) each. The neural network is equipped with two hidden layers with 47 neurons each and one hidden layer with 13 neurons. Accordingly, the overall

Table 3 Internal flow dataset prediction error

[%]	Type I	Type II	Type I + II
Training set	3.21	6.39	5.17
Validation set	3.80	20.07	13.81

configuration of the network architecture is RLF(48)-47-47-13-2, with 3 hidden layers and in total 4 operating layers.

Figures 15 and 16 show the prediction results (both velocity components compared to the CFD reference) for a Type I geometry and a Type II geometry, respectively. Both are part of the validation set and each has a prediction accuracy approximately equal to the mean prediction of the validation of its type.

The prediction of the Type I geometry shows a good agreement with the CFD reference over all areas of the flow field. In contrast, the prediction of the Type II geometry shows a significant deviation from the reference in some places of the flow domain. This difference is also confirmed by the quantitative evaluation. Table 3 shows the mean ARE for the training and validation set differentiated by the two geometry types.

Although Type II geometries are more present in the training database than Type I geometries, the neural network predicts the associated flow fields much worse. A close look at the Type II prediction in Fig. 16 shows that the neural network reproduces a velocity profile typical for a laminar flow in all channel branches but does not correctly estimate the flux within individual branches. In some areas of the fluid domain, the neural network expects a higher or lower flux than is actually present. Within the Type I prediction (Fig. 15) this error does not occur because the flux is by definition constant over all cross sections. Here the neural network not only correctly predicts the laminar flow profile, but also correctly estimates the flux.

The training and testing of the Internal flow dataset show, that the model is in principle capable of predicting internal flow fields. However, with the Type II geometries it is indicated, that the proposed machine learning model is not yet entirely capable of fulfilling the principle of mass conservation.

4.4 Further Testing

So far, for each data set, one neural network was trained specifically. In this part, a common neural network is trained with geometries of all datasets. The aim is to investigate whether a single neural network can be used for universal flow estimation. For this purpose, 82 % of the geometries of each category constitute the new combined training set, while 18 % of each category constitute the combined validation set. Thus the training set consists of 771 geometries and the validation set of 167 geometries. Note that viscosity and inlet velocity were the same across all datasets, while the Reynolds numbers varied due to different characteristic lengths (and definitions of the characteristic lengths).

The RLF parameters are selected as follows 12 layers in circumferential direction and 5 layers in radial direction with an outer radius of 192 pixels. The total number of input features is therefore $5 \times 12 = 60$. The number of extracted data points per geometry is set to 1000. In summary, the training database consists of 1000×771 data points with 60 inputs and 2 outputs each. The neural network is equipped with two hidden layers with 47 neurons each and one hidden layer with 13 neurons. Accordingly, the

Fig. 17 Visualization of the prediction (first and third row) compared to the reference solution (second and fourth row) on an exemplary geometry of each category. Color-map (not shown) axis-limits are consistent for prediction and reference of each geometry

network architecture configuration is RLF(60)-47-47-13-2, with 3 hidden layers and in total 4 operating layers. Figure 17 shows prediction results for one exemplary geometry of each category.

The mean error of the reproduced training geometry flow fields is 10.61 % and the mean error of the predicted validation geometries is 12.98 %. In comparison, the accordingly weighted mean error of all previously specifically trained neural networks is 8.60 % for their training sets and 10.35 % for their validation sets.

As could be expected, the prediction quality for the universally trained network is worse compared to the problem-specific trained networks. However, the decline remains within a few percentage points. It can be shown, that neural networks according to the presented method are able to be simultaneously trained on different physical problems and address these unrestricted to the shape and lateral extend of the quantity fields. To the best of our knowledge, these features are currently unique among other concepts.

5 Conclusions

In this study a new machine learning model for fast flow field estimation was presented and successfully tested on laminar flow fields in various types of geometries including internal and external flow at low Reynolds numbers.

In its main components, the model consists of a multilayer perceptron (MLP) and a preceding radial-logarithmic filter (RLF) for geometrical preprocessing. As an element for spatial encoding, the RLF translates the continuously defined flow guiding geometry into a fairly small number of discrete $\{[0,1]\}$ -normalized values by systematic filtering and compression. Transferred to the MLP as its input features, these values can be effectively interpreted by it. From these inputs, the MLP is trained to provide the desired flow field quantities, e.g. the velocity components. The prediction scheme is point-to-point related. This means, that the RLF refers to a considered (focused) point in the geometry and that the neural network predicts the flow quantities at that point. For stationary flows this aspect allows to immediately predict the state at a considered point or the entire flow field without an iteration in time.

The model was tested on three different datasets of geometries and associated reference flow fields simulated by CFD: a dataset of randomly modeled car shapes placed in a virtual wind tunnel, a dataset of systematically parametrized airfoil shapes and an internal flow dataset containing curved and branched flow channels. In all categories the model could estimate flow fields with acceptable accuracy. Relative error rates less than 43 % were reached for the Cars dataset, less than 2.7 % for the Airfoil dataset, less than 3.9 % for the curved flow channels and less than 21 % for the branched flow channels.

Overall, the model proved to be universally applicable for different categories of test cases. The utilization of the RLF allows a good flow field estimation and good generalization capability even with comparatively minimalistic neural networks and small datasets. In contrast to the field-to-field prediction scheme, the applied point-to-point prediction scheme eliminates the restriction to flow domains with a fixed lateral extent. In addition, the method eliminates the explicit inclusion of model equations such as Navier–Stokes, an iteration for steady-state flow fields, as well as a classical spatial discretization with numeric meshes and the associated difficulties of traditional CFD methods.

However, the flow conditions investigated so far are unequally simple in contrast to technically relevant conditions. They offer a good starting point, but leave some essential disadvantages of the concept unresolved. At the current state, RLF-extracted features are the only input parameters for the subsequent neural network. This leads to a decreasing prediction accuracy with increasing distance from solid geometry. It also results in an insufficient ability to meet mass conservation requirements or to predict transient flow behavior. Future work is therefore intended to address these problems in order to reach a level at which flow fields can be estimated, that are relevant for practical applications. This should involve the conduction of transient and high Reynolds number studies. For this purpose, the inclusion of further input parameters, e.g. material properties, dimensionless parameters or flow quantities of past time steps in the transient case, should be considered alongside the RLF-extracted features.

The concept has been proven on standard non-deep feed-forward networks and could benefit significantly from more sophisticated architectures such as CNN and LSTM as a replacement for the MLP to process RLF-information. Although inferior to CFD in almost every respect at the current state, we suggest that ML-based methods could open up an additional perspective on the behavior of fluid flow if they are further developed.

Funding Open Access funding enabled and organized by Projekt DEAL.. Not applicable.

Availability of data and material Not applicable.

Code availability Parts of the code and the trained neural networks are made available as supplementary material.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

Appendix A: Further Illustration on the RLF

Figure 18 shows the RLF in action at four different focused points, where the cell value is indicated as gray scale ($W_{r,\varphi} = 0 \rightarrow \text{white}$; $W_{r,\varphi} = 1 \rightarrow \text{black}$). Here, exactly the same settings are made for the matrix as for the training of the neural network used for the Internal Flow dataset: A total of 48 cells with 12 layers in circumferential direction and 4 layers in radial direction and an outer radius set to 104 pixels. Note that the cells become exponentially larger with each additional radial layer. This is intended to give higher weight to wall-structures in the vicinity of a point under consideration than to structures that are farther away. Table 4 shows the RLF-extracted features as they are submitted to the neural network at the same focused points.

Fig. 18 RLF operation illustrated on an internal flow geometry by evaluating four different focused points. Small incline of the channel in the inlet area (a). Greater incline and slight expansion of the channel at (b). Division of the channel at (c). Position close to the wall at (d). Note that the outer radius of the RLF has been set to 104 pixels and appears smaller for illustrative reasons

Table 4 Input-array W obtained from the RLF at the four focused points in Fig. 18. The inner radial layer is labeled $r = 1$. The first circumferential layer with $\varphi = 1$ starts at -45° in the lower right and all further circumferential layers follow in mathematically positive direction

φ	r			
	1	2	3	4
1	0.04	0.88	1	1
2	0	0.14	0.70	0.99
3	0	0.02	0.26	0.32
4	0.15	0.91	1	1
5	0.50	1	1	1
6	0.56	1	1	1
7	0.08	0.74	1	1
8	0	0	0.23	0.61
9	0	0.62	0.99	1
10	0.38	1	1	1
11	0.45	1	1	1
12	0.33	1	1	1
1	0.08	0.91	1	1
2	0	0.21	0.93	1
3	0	0	0.07	0.50
4	0	0.52	0.73	0.75
5	0.26	1	1	1
6	0.51	1	1	1
7	0.25	1	1	1
8	0	0.42	0.70	0.65
9	0	0	0.42	0.95
10	0	0.56	1	1
11	0.17	1	1	1
12	0.41	1	1	1
1	0	0.58	1	1
2	0	0.03	0.62	0.74
3	0.56	0.58	0.61	0.70
4	0.88	0.64	0.30	0.94
5	0.045	0.02	0.90	1
6	0	0.50	1	1
7	0	0.77	1	1
8	0	0.59	1	1
9	0	0.02	0.36	0.54
10	0	0.03	0.53	0.99
11	0	0.62	1	1
12	0	0.82	1	1
1	0.98	1	1	1
2	0.90	0.92	0.72	0.73
3	0.03	0.06	0.70	0.71
4	0	0.61	0.55	0.96
5	0	0.85	0.42	1
6	0	0.88	0.56	1
7	0	0.80	0.46	1
8	0	0.14	0.17	0.97

Table 4 (continued)

φ	r			
	1	2	3	4
9	0.08	0.30	0.43	0.51
10	0.90	1	1	1
11	0.98	1	1	1
12	1	1	1	1

Appendix B: Details on Multilayer Perceptron

This section provides a brief overview of MLP and points out its role for the proposed method of flow estimation. For theoretical details we refer to Kruse et al. (2016).

MLP is a machine learning technique inspired by a simplification of the learning principle in biological organisms. Analogous to neurons and synaptic connections in a natural neural network, computation units are connected with each other through weights. An MLP calculates a function of the input by propagating the values from the input layer through a certain number of hidden layers to the output layer, using weights as intermediate parameters. As long as a neuron does not belong to the output layer, it is an input for all connected neurons of the next layer multiplied by the weight of the respective connection (feed-forward operation). Neurons of the output layer provide the MLP result values. The initially random weights are adjusted gradually with a suitable optimization algorithm until the MLP calculates the desired function. This adaptation of the weights is referred as learning. The learning strategy used in this study is *supervised learning*. It is the task of learning a function that maps an input to an output based on a sampling-dataset, which contains a sufficient number of input-output pairs. Figure 19 shows an MLP as it is integrated into our concept.

Fig. 19 MLP architecture as it is used in the flow estimation network. At the left interface is the input layer, where the MLP receives the point-specific geometry features from the RLF. At the right is the output layer, which passes the predicted flow quantities at the focused point. In between are three hidden layers

Fig. 20 Data flow diagram during training stage (upper block) and testing stage (lower block)

Appendix C: Overview of the Methodology

The proposed ML-concept is composed of the RLF described in "Appendix A" and the MLP described in "Appendix B". Figure 20 illustrates the communication between these parts during training stage and testing stage, respectively. Point-related input features (provided by the RLF) and flow quantities at the same point (known from reference CFD simulations) form one data point. The MLP is trained using a dataset consisting of a large number of data points extracted from different geometries. Once the MLP is trained, it can be used in the testing stage to point-wise predict flow quantities within a new geometry from RLF input.

References

- Ahmed, M., Qin, N.: Surrogate-based aerodynamic design optimization: use of surrogates in aerodynamic design optimization. In: International Conference on Aerospace Sciences and Aviation Technology, vol. 13, pp. 1–26. The Military Technical College (2009)
- Bhatnagar, S., Afshar, Y., Pan, S., Duraisamy, K., Kaushik, S.: Prediction of aerodynamic flow fields using convolutional neural networks. *Comput. Mech.* **64**(2), 525–545 (2019)
- Chen, J., Viquerat, J., Hachem, E.: U-net architectures for fast prediction of incompressible laminar flows. *arXiv preprint arXiv:1910.13532* (2019)
- Edeling, W.N., Cinnella, P., Dwight, R.P., Bijl, H.: Bayesian estimates of parameter variability in the $k\text{-}\epsilon$ turbulence model. *J. Comput. Phys.* **258**, 73–94 (2014)
- Fukami, K., Nabae, Y., Kawai, K., Fukagata, K.: Synthetic turbulent inflow generator using machine learning. *Phys. Rev. Fluids* **4**(6), 064603 (2019)
- Guo, X., Li, W., Iorio, F.: 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 (2016)

- Jacobs, E.N., Ward, K.E., Pinkerton, R.M.: The Characteristics of 78 Related Airfoil Section from Tests in the Variable-Density Wind Tunnel. US Government Printing Office, Washington (1933)
- Kim, B., Azevedo, V.C., Thuerey, N., Kim, T., Gross, M., Solenthaler, B.: Deep fluids: a generative network for parameterized fluid simulations. *Comput. Graph. Forum* **38**, 59–70 (2019)
- Kruse, R., Borgelt, C., Braune, C., Mostaghim, S., Steinbrecher, M.: Computational Intelligence: A Methodological Introductio, chap. Multi-layer Perceptrons. Springer, Berlin (2016)
- Levenberg, K.: A method for the solution of certain non-linear problems in least squares. *Q. Appl. Math.* **2**(2), 164–168 (1944)
- Ling, J., Kurzawski, A., Templeton, J.: Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *J. Fluid Mech.* **807**, 155–166 (2016)
- Ling, J., Templeton, J.: Evaluation of machine learning algorithms for prediction of regions of high reynolds averaged navier stokes uncertainty. *Phys. Fluids* **27**(8), 085103 (2015)
- Marquardt, D.W.: An algorithm for least-squares estimation of nonlinear parameters. *J. Soc. Ind. Appl. Math.* **11**(2), 431–441 (1963)
- Maulik, R., San, O., Rasheed, A., Vedula, P.: Data-driven deconvolution for large eddy simulations of kraichnan turbulence. *Phys. Fluids* **30**(12), 125109 (2018)
- Rosenblatt, F.: The Perceptron, A Perceiving and Recognizing Automaton Project Para. Cornell Aeronautical Laboratory, New York (1957)
- Rumelhart, D.E., Durbin, R., Golden, R., Chauvin, Y.: Backpropagation: the basic theory. In: Chauvin, Y. and Rumelhart, D.E. (eds.) *Backpropagation: Theory, Architectures and Applications*, pp. 1–34. Lawrence Erlbaum Associates, Hillsdale (1995)
- Sekar, V., Jiang, Q., Shu, C., Khoo, B.C.: Fast flow field prediction over airfoils using deep learning approach. *Phys. Fluids* **31**(5), 057103 (2019)
- The Mathworks, Inc., Natick, Massachusetts: MATLAB version 9.7.0.1296695 (R2019b) (2019)
- Thuerey, N., Weissenow, K., Prantl, L., Hu, X.: Deep learning methods for reynolds-averaged Navier–Stokes simulations of airfoil flows. *AIAA J.* **58**(1), 25–36 (2020)
- Tompson, J., Schlachter, K., Sprechmann, P., Perlin, K.: Accelerating eulerian fluid simulation with convolutional networks. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3424–3433. JMLR. org (2017)
- Tracey, B., Duraisamy, K., Alonso, J.: Application of supervised learning to quantify uncertainties in turbulence and combustion modeling. In: *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, p. 259 (2013)
- Wang, Z., Luo, K., Li, D., Tan, J., Fan, J.: Investigations of data-driven closure for subgrid-scale stress in large-eddy simulation. *Phys. Fluids* **30**(12), 125101 (2018)
- Wiewel, S., Becher, M., Thuerey, N.: Latent space physics: towards learning the temporal evolution of fluid flow. *Comput. Graph. Forum* **38**, 71–82 (2019)
- Xie, Y., Franz, E., Chu, M., Thuerey, N.: Tempogan: a temporally coherent, volumetric gan for super-resolution fluid flow. *ACM Trans. Gr.* **37**(4), 1–15 (2018)
- Yang, C., Yang, X., Xiao, X.: Data-driven projection method in fluid simulation. *Comput. Animation and Virtual Worlds* **27**(3–4), 415–424 (2016)