Machine Learning to Predict Aerodynamic Stall

Ettore Saetta a, Renato Tognaccini a and Gianluca Iaccarino b

aDepartment of Industrial Engineering, University of Naples Federico II, Naples, Italy; bDepartment of Mechanical Engineering, Stanford University, Stanford, CA, USA

ABSTRACT
A convolutional autoencoder is trained using a database of airfoil aerodynamic simulations and assessed in terms of overall accuracy and interpretability. The goal is to predict the stall and to investigate the ability of the autoencoder to distinguish between the linear and non-linear response of the airfoil pressure distribution to changes in the angle of attack. After a sensitivity analysis of the learning infrastructure, we investigate the latent space identified by the autoencoder targeting extreme compression rates, i.e. very low-dimensional reconstructions. We also propose a strategy to use the decoder to generate new synthetic airfoil geometries and aerodynamic solutions by interpolation and extrapolation in the latent representation learned by the autoencoder.

ARTICLE HISTORY
Received 6 July 2022
Accepted 15 January 2023

KEYWORDS
Machine Learning; Autoencoders; Aerodynamic stall; RANS; Latent Space

1. Introduction
Experimental and computational techniques to investigate aerodynamic performance have matured in the last decades providing increasingly large amounts of quantitative information. These extensive datasets and the recent success of machine learning (ML) techniques in a variety of applications have spurred interest in developing data-driven techniques for fluid mechanics. Artificial neural networks and other supervised ML techniques algorithms have demonstrated promising potential to improve turbulence models (Duraisamy, Iaccarino, and Xiao 2019; Duraisamy, Zhang, and Singh 2015; Tracey, Duraisamy, and Alonso 2015; Milano and Koumoutsakos 2002), to accelerate shape optimisation (Yan et al. 2019; Li et al. 2020) and to investigate flow control strategies (Noack 2019; Verma, Novati, and Koumoutsakos 2018).

In this work, we investigate the ability of an unsupervised ML strategy, convolutional autoencoders, to predict aerodynamic characteristics and specifically the stall of classic wing cross-sections.

Airfoil stall is a strongly non-linear phenomenon corresponding to the loss of lift force and a primary design consideration for airplanes and rotor-crafts. Figure 1 illustrates the lift curves (variation of lift coefficient $C_l$ as function of the angle of attack $\alpha$) for the Boeing VR12 airfoil in steady Figure 1(a) and unsteady Figure 1(b) regimes, experiment by Matalanis et al. (2016). The figure also shows the capabilities of CFD simulations as obtained by present authors (SU2 RANS and URANS solver (Economon et al. 2016)) and by Matalanis et al. (CFL3D) in predicting these phenomena.

The stall condition corresponds to the maximum lift. For fixed-wing aircraft at a low angle of attack, classical aerodynamic theories predict a lift coefficient ($C_l$) that varies linearly with $\alpha$; on the other hand for $\alpha > 10^\circ$ a strong non-linear behaviour is observed with the lift decreasing after the stall angle due mainly to the presence of turbulent separated flow on the upper surface of the airfoil. The aerodynamics of rotating wing aircraft is even more complex because of the periodic change of the wing position with respect to the incoming flow. In this case, a sinusoidal variation of the angle of attack in time (pitching airfoil) corresponds to an unsteady flow and the maximum lift condition is referred to as a dynamic stall. In Figure 1(b) a pitching airfoil with a reduced frequency of $k = 0.05$ and $\alpha \in [0^\circ, 20^\circ]$ is reported for the same VR12 airfoil. The dynamic stall occurs at higher angles of attack compared to the steady counterpart, with flow separation persisting when the angle of attack reduces (hysteresis effect).
The comparison of the numerical simulations is quite satisfactory in the proposed picture. The small shift of numerical and experimental results is very likely due to a not perfect angle of attack correction of wind tunnel data. Present limits of RANS and URANS methods however appear in the post-stall and deep-stall conditions dominated by a very massive separated flow.

The static and dynamic stall characteristics have a critical role in defining the performance of aircrafts, helicopters and wind turbines. Experimental studies are typically limited by the ability to achieve flight conditions in laboratory settings; numerical simulations, on the other hand, require the representation of the thin turbulent boundary layer developing on the airfoil surfaces, and several complex physical processes such as unsteadiness, laminar-turbulent transition and flow separation. The corresponding predictions are computationally expensive and extensive research continues to be devoted to the improvement and assessment of the corresponding simulations. A summary of the state of the art for a static stall is the AIAA high-lift prediction workshop (Rumsey, Slotnick, and Sclafani 2019).

As mentioned earlier, in this work we concentrate on static stall predictions and approach the problem using a machine-learning technique.

Autoencoders (AE) are unsupervised deep-learning algorithms, which fundamentally target data compression, i.e. reducing the dimensionality of the input data (Lee and Carlberg 2020). The architecture of an autoencoder consists of two main elements, the encoder and the decoder. Giving an input dataset, AEs construct (learn) an equivalent low-dimensional representation, termed the latent space using the encoder portion of the network. On the other hand, The decoder rebuilds the input from the latent space. A simplified scheme of an autoencoder is reported in Figure 2 and summarised mathematically as:

\[ V = e(x); \quad \hat{x} = d(V); \quad \hat{x} = f(x) = d(e(x)), \]  

(1)

where \( e() \) is the encoder function, \( d() \) is the decoder function, \( f() \) is the entire transformation encoder+decoder, \( V \) is the vector of latent variables (also referred to as latent space) and \( x \) and \( \hat{x} \) are respectively the input and its reconstruction.

\[ x, \hat{x} \in \mathbb{R}^{n \times m \times c}; \quad V \in \mathbb{R}^{p}, \]

where \( n \times m \times c \) is the dimension of the input dataset and \( p \) is the dimension of the latent space.

In a Convolutional Autoencoder, at least one of the layers of the network performs a convolution operation (denoted with *), which, in discrete form, is given by:

\[ s(t) = (x \ast w)(t) = \sum_{a=-\infty}^{\infty} x(a) w(t-a) \]  

(2)

where \( x \) is the input, \( w \) is the kernel and \( s \) is the feature map.
Convolutional autoencoders have been already used for aerodynamic predictions, for instance, Bhatnagar et al. (2019) showed their capability to reproduce the flow around airfoils with different shapes, angles of attack and Reynolds numbers. In their work, they used a signed distance function (SDF) as input to the encoder and provided the velocity and pressure fields as output of the decoder. In this way, it is possible to obtain the flow field around a new geometry in real time by using only geometrical information (the SDF). The same methodology was adopted by Tangsali, Krishnamurthy, and Hasnain (2020), but using a very rich training database (11,000+ CFD simulations), in order to demonstrate the generalizability of this approach for a wide range of airfoil geometries, Reynolds numbers and Mach numbers.

Rather than focusing on overall accuracy, Agostini (2020) focused on the interpretation of the latent space in the case of a flow around a cylinder. In his work, Agostini, showed the possibility to derive a low-dimensional dynamical model (in the latent space) by applying a clustering algorithm.

Other studies on convolutional neural networks (CNNs) and autoencoders have been carried out for the aerodynamic design of airfoils. In this context, Tangsali, Krishnamurthy, and Hasnain (2020) carried out a comprehensive analysis using an extensive airfoil database (more than 10,000 geometries) and demonstrated the remarkable performance of the autoencoder in predicting the fluid flow and generalizability in different flow conditions. Sekar et al. (2019) trained a CNN to predict the airfoil geometry giving in input to the pressure coefficient distribution on the airfoil surface. Li, Zhang, and Chen (2022) proposed a variational autoencoder for the inverse design of supercritical airfoils, injecting physical information (free-stream Mach number, lift coefficient, shock wave position and Mach number upstream of the shock wave) in the latent space of the autoencoder. This approach assumes the physical parameters necessary to describe the phenomenon are known as a-priori and provided to the autoencoder during the training phase.

The present investigation is informed by these previous studies but its aim is quite different: instead of constructing the autoencoder to generate the flow given a geometrical representation of the airfoil, we trained the autoencoder to reproduce the flow field itself (as sketched in Figure 2, input and output are the same) and then focused on the latent space representation to generate new airfoils and flow fields by using only the decoder. One critical goal of this work is to investigate the ability of autoencoders to extract low-dimensional latent spaces characterised by physical imprinting: the latent variables closely correlate to aerodynamic characteristics of the airfoil. This correspondence is extracted automatically from the training process and not imposed as a constrain on the process. The imprinting is demonstrated by constructing low-dimensional databases that only encompass simple geometrical changes and, by limiting the study to very few latent variables.

The article is organised as follows: first, we provide a description of the training dataset construction; then we analyse the latent variables of the autoencoder trained in the linear regime (airfoil operating in the linear part of the lift curve) for both images and raw data in input. We performed a sensitivity analysis on the autoencoder architecture, hyperparameters and regularisation method. We present an investigation of the latent space for different values of its dimension to assess interpretability. Finally, we investigate how to use the latent space with only the decoder in order to generate new synthetic airfoils and flow fields by interpolation and extrapolation beyond the trained latent space.

2. Database definition

A critical element of data-driven models is the dataset used for training. In the present work, we consider numerical solutions of the Reynolds-Averaged Navier–Stokes (RANS) equations we obtained using the open-source software SU2. We constructed the database using RANS solutions for simplicity and efficiency, however, it is possible to perform feed the network with results obtained with higher fidelity solutions such as Large Eddy Simulations (LES), Direct Numerical Simulations (DNS) or even experiments. It is also possible to combine solutions obtained using either RANS or LES, one possible choice is the bi-fidelity approximation based on the Interpolative Decomposition developed for uncertainty quantification in Fairbank et al. (2020).

Computations of the steady-state aerodynamic characteristics of NACA 4-digit airfoils are carried out. The free-stream conditions of Mach number and Reynolds number are fixed for all the simulations:
\[ M_\infty = 0.15, \quad Re_\infty = 5 \times 10^6. \] The Spalart-Allmaras turbulence model is adopted and the flow is assumed fully turbulent.

In particular, we constructed 2 databases:

- **Linear case** (33 RANS solutions), changing 2 parameters:
  - airfoil curvature (NACA 2412, 4412, 6412),
  - angle of attack \( \alpha \in [-5^\circ, 5^\circ] \).

- **Non-linear case** (124 RANS solutions), changing 2 parameters:
  - airfoil maximum thickness \( t \) (NACA 0012, 0014, 0016 and 0018),
  - angle of attack \( \alpha \in [0^\circ, 30^\circ] \).

The first dataset is used to verify the autoencoder characteristics as it learns linear aerodynamics (low angle of attack) and it is described in Section 3. The second dataset is used for the investigation of the airfoil static stall discussed in Section 4.

The datasets include a subset of the field variables computed as part of the RANS solutions; specifically, we included the pressure coefficient \( Cp \), the vorticity \( \omega \) and the Mach number \( M \) as they are directly relevant to the stall detection. \( Cp \) is the critical contribution to the aerodynamic forces and therefore critical to establish the flow regime and the loss of lift; \( \omega \) provides information about the boundary layers and the presence and position of eventual separation points on the surface of the airfoil; \( M \) is correlated to the velocity field and yields an indication of the flow separation and the wake structure. In addition to field quantities, we also include geometrical information in the dataset to distinguish different airfoils; specifically we included the \( x \) and \( y \) coordinates of the mesh grid points.

The SU2 computations were carried out using a structured O-mesh (as reported in Figure 3); the field variables are organised in matrices using the \( i \) and \( j \) indexes of the mesh. The \( i \) index is a curvilinear coordinate on the airfoil surface and it corresponds to the horizontal axis of the encoded mapping, while the \( j \) index tracks the radial direction away from the airfoil and corresponds to the vertical axis encoded mapping. This input representation has also the advantage of amplifying the boundary layer as \( j \) does not follow the mesh clustering at the airfoil surface (see Figure 3); this in turn enables us to highlight the boundary layers in the training step. In summary, the input to the convolutional autoencoder consists of 2D matrices (\( ij \)-mappings) each with five channels (\( cp, \omega, M, x \) and \( y \)), and the variables are scaled between 0 and 1. The datasets are composed of either images (RGB channels) or raw solutions (matrix of floating points). Therefore, the number of input channels is 15 for the images (3 RGB \( \times \) 5 variables) and 5 for the raw data (1 channel \( \times \) 5 variables).

### 3. Autoencoder sensitivity analysis

#### 3.1. Architecture

As a preliminary step of the present investigation, we carried out an extensive analysis to assess the effect of the network architecture (number of layers, activation functions, etc.) and the training strategy and related hyperparameters. As a trade-off between ease of training and accuracy, we focused on the following structure:

- **Encoder:**
  - 4 convolutional layers.
  - 2 linear layers.
  - ReLU activation functions.
- **Latent space dimension**: \( p = 1, 2, 3 \).
- **Decoder:**
  - 2 linear layers.
  - 4 transposed convolutional layers.

![Figure 3. ij mapping.](image-url)
ReLU activation functions; Sigmoid for the last layer.

We trained the autoencoder using a completely randomised dataset with a batch size equal to 2, the Adam optimiser and the Mean Squared Error (MSE) as a loss function. The algorithm is implemented using the Pytorch library (Paszke et al. 2019) and trained on a single GeForce GTX Titan X GPU.

The trainable parameters of the autoencoder are $\sim 43$ millions, while the database dimension is $\sim 53$ millions: $124 \times 5 \times 170 \times 512$ (124 solutions, 5 physical variables and $170 \times 512$ grid points for each solution). The loss minimisation process used for training is effectively a regression.

The latent space dimension $p$ is the most important hyperparameter of an autoencoder. By increasing $p$ it is possible to reach a certain level of tolerance in the training process with less epochs. However, a high dimensional latent space can lead to overfitting and ultimately, poor performance on new unseen cases. For our purposes, another critical design goal is interpretability: a very high dimensional latent space limits our ability to extract physical understanding from the latent variables. In the results sections, we focus on extremely low values of ($p \leq 3$) and in many cases, we have achieved error tolerances that are comparable to richer latent representations.

As an example, in Figure 4 we report the convergence of the training procedure for the same autoencoder and dataset (the non-linear regime) for $p = 1, 2$ and 3. The results illustrate that $p = 2$ and $p = 3$ reach a reasonably close level of accuracy in the training, while $p = 1$ is clearly insufficient in this case.

In Figure 5 we compare the convergence of the training procedure for the two datasets (linear and non-linear) using the same autoencoder with $p = 3$. The two cases show the same rate of decay of the loss function. The comparison is done using the same training parameters (e.g. learning rate) for both linear and non-linear cases without any tuning to speed up the training process.

In our specific case, two parameters are varying in the database: the airfoil thickness and the angle of attack. For this reason, at least two latent variables are necessary and sufficient, to describe our dataset. For our analysis, we decided to use three latent variables that is a good compromise between latent space interpretability and training speed performance.

3.2. Input type: images or raw data?

As a first step, we investigate the effect of the type of input for the convolutional autoencoder. Images are widely used in this context and therefore, we start by investigating if images provide a sufficient quantitative representation of the physical solutions. We performed a sensitivity analysis by training the autoencoder with the linear dataset and investigated the behaviour of the latent variables (with $p = 3$).

Figure 6(a–c) shows normalised latent variables learned by the autoencoder trained with images constructed using different contour levels: 50, 500 and 5000 respectively. These are also compared to using the raw data directly (Figure 6(d)). It is clear that the latent variables reported as a function of the angle of attack are quite sensitive to the input format. It is expected that in these conditions the latent space represents closely the variability expressed by the angle of attack. The results confirm that increasing the number of contour levels in the image (e.g. increasing the precision in representing the data) leads to a better correlation between the latent variables and the expected...
variability. Using the raw data as input leads to a collapse of the latent space and effectively a purely linear response to the changes in the angle of attack.

Figure 7 shows the same sensitivity analysis carried out in the non-linear regime. As before, we notice a sensitivity of the latent space to the number of contour levels. Moreover, the raw data in input produced latent variables which are linear for low angle of attack consistently with what was shown previously. All the results reported in what follows are obtained by using raw data in input.

3.3. Regularisation

As described by Newson et al. (2020), different regularisation methods can lead to different behaviour of the latent variables and different reconstruction performances by the autoencoder. We study the sensitivity of the present algorithm to using $L_2$ norm regularisation on the encoder and/or decoder weights and on the latent variables. Eventually, we found that the $L_2$ norm regularisation applied only to the encoder weights leads to a tidy and compact latent space which is amenable to interpretation. As a comparison of the various types of regularisation, we report the latent variables obtained using the non-linear regime dataset and $p = 3$ (Figure 8): the results are not vastly different, although it is clear that regularising the encoder weights leads to a compact latent space. The latent space regularisation in Figure 8(b) is obtained by applying a penalisation on the $L_2$ norm of the latent variables, while the encoder weights regularisation in Figure 8(c) is an $L_2$ norm of the weights of the Encoder.

4. Latent space investigation

Based on the sensitivity analysis illustrated earlier, we trained the autoencoder applying the $L_2$ norm regularisation on the encoder weights. The non-linear regime dataset is considered for the investigation of the latent space, considering raw data and variables normalised between 0 and 1. The input channels are $C_p$, $M$, $\omega$, $x$ and $y$.

Figure 9 shows the latent variables for $p = 1$, $p = 2$ and $p = 3$ learned by the autoencoder. The dataset consists of RANS computations of the flow around symmetric airfoil of different thickness at various angles of attack; effectively two free input parameters (thickness $t$ and $\alpha$) span the entire database.

By mapping the latent space to the free input parameters is clear that the autoencoder is correctly ordering
the solutions although \( t \) and \( \alpha \) are not explicitly given as inputs and the training process is randomised.

Figure 9(a) shows that \( p = 1 \) is not sufficient to describe the entire dataset; for \( \alpha < 10^\circ \), one value of the latent variable corresponds to multiple airfoils. However, this case illustrates an important finding that connects the latent variables to the physics of the phenomenon. The autoencoder infers that symmetrical airfoils have the same behaviour at low angle of attacks, while in the non-linear regime, the thickness plays an important, differentiating role. This is confirmed by the analysis of the lift curves extracted from the RANS solutions (Figure 10(a)), which completely overlap for \( \alpha < 10^\circ \).

The existence of a strong correlation between the lift coefficient \( C_l \) and the latent variable \( V_1 \) for \( p = 1 \) is clearly visible also in Figure 11, where the \( C_l \) is reported as a function of \( V_1 \). The different curves for the 4 airfoils collapse almost perfectly. In particular, the correlation between \( C_l \) and \( V_1 \) is greater than 90\% in the linear regime.

5. Generating synthetic airfoils

The decoder element of the present algorithm can be used to quickly generate new synthetic solutions by modifying the values of the latent variables. As a final test, we investigated the ability of the autoencoder to generate new airfoils and their aerodynamic characteristics.

5.1. Latent variables translation

A simple modification to the latent variables is first attempted. We considered the latent variables of the NACA 0012 as baseline. Figure 12 shows the variables as function of the angle of attack for the NACA 0012. We apply a translation of one of the component \( V_3 \) by 10\%.

In Figure 13, the output of the decoder with the new latent variables is compared with the NACA 0012 baseline in terms of pressure distributions and airfoil geometry for 4 selected angles of attack (\( \alpha = \)
0°, 10°, 20° and 30°). As a first observation, it is clear that the synthetic geometry changes with the angle of attack, which can be interpreted as an indication that changes in the latent space variables cannot be arbitrary.

In the $C_p$ plots (Figure 13) the continuous lines refer to the lower surface of the airfoil, while the dashed lines to the upper surface. The decoder extracts an airfoil and a $C_p$ distribution for $\alpha = 0^\circ$ that corresponds to a negative angle of attack for a symmetric airfoil.

The $C_p$ curves and the airfoil geometries reported in Figure 13 are obtained by extracting the first row of the reconstructed channels ($C_p$, $x$ and $y$). The AE is trained giving in input $170 \times 512 \times 5$ tensors, therefore the airfoil geometry is the reconstruction of 2 vectors ($x$ and $y$ coordinates) of 512 elements each. This means that the airfoil geometry is only 0.23% of the information that the AE is reproducing. In addition, the poor quality of the leading and trailing edge reconstructions are due to the presence of a thickening of the mesh grid in these regions, so the values of the coordinates are very close to each other and small error influences the reconstruction quality.

The translation of $V_3$ seems to correspond to a translation of $\alpha$. In particular, considering the slope of $V_3$ in its linear part ($\Delta V_3/\Delta \alpha \approx 0.037/\circ$), the translation we applied ($\Delta V_3 = -0.15$) corresponds to a new zero-lift angle of attack $\alpha_{zl} = 4^\circ$.

Figure 14 shows that this is consistent with the results extracted by the autoencoder for $\alpha = 4^\circ$: the pressure coefficient on the upper and lower surfaces of the synthetic airfoil overlap, returning $C_l = 0$.

The behaviour of $V_3$ resembles the lift curve, and the results corroborate this observation.

### 5.2. Interpolation in the latent space

It is not possible to manipulate the latent space arbitrarily, and it is necessary to preserve the correlation between the latent variables. We developed a controlled
strategy to extract new synthetic airfoils and flow fields from the autoencoder.

In the first step, we created a mapping of the latent space on the database free parameters (airfoil thickness and angle of attack). The contour mapping is reported in Figure 15 for each latent variable of the latent space with \( p = 3 \). This mapping allows us to easily interpolate, and we constructed the latent variables corresponding to 13% airfoil thickness (represented by the black line in Figure 15).

The resulting latent variables as function of the angle of attack \( \alpha \) are reported in Figure 16 and compared with the latent variables of the NACA 0012 airfoil.
Starting from these interpolated latent variables the decoder generates solutions that can also be compared to the RANS solutions generated for the same expected profile (NACA 0013). Figure 17 shows the Mach number contours for 4 angles of attack of the NACA 0012, 0014 and for the synthetic NACA airfoil generated by the decoder. In order to have a more quantitative analysis of the result, it is interesting to look at the pressure coefficient $C_p$ on the airfoil surface as reported in Figure 18. The yellow curves refer to the NACA 0013 RANS solution, while the red curves refer to the decoder output (the synthetic NACA 0013), and they perfectly match both in the linear regime (with the attached flow in the first two plots of Figure 18) and the non-linear regime and stall conditions (the last two plots of Figure 18). Also, the geometry computed by the decoder is closely matching the expected
NACA 0013 airfoil, and it is not changed with the angle of attack, showing that a coherent latent space manipulation leads to a consistent result.

5.3. **Extrapolation beyond the latent space**

This result presented in the previous section clearly shows that it is possible to interpolate in the latent space learned by an autoencoder in order to obtain accurate aerodynamic predictions for a strongly nonlinear phenomenon such as the airfoil stall. But is it possible to extrapolate beyond the training dataset? We performed a cubic extrapolation of the latent variables in the $\alpha$-$t$ plane, obtaining the new contour maps in Figure 19, where the black dashed rectangle represents the original boundaries of the input free parameters.

We extrapolated the values of the three latent variables at constant thickness $t = 20\%$ and $t = 10\%$, respectively represented by the continuous red and black lines in Figure 19.

Considering the $t = 20\%$ extrapolation, the latent variables are plotted as a function of the angle of attack $\alpha$ in Figure 20. Figure 21 shows the Mach number contours (for $\alpha = 0^\circ$, $10^\circ$, $20^\circ$ and $30^\circ$) extracted from the RANS solutions of the NACA 0018 and 0020 and the ones generated by the decoder with the extrapolated latent variables in input.

The contour of the Mach number of the synthetic flow field appear to be in reasonable agreement with the RANS solution, however, in order to obtain a better evaluation of the results, we reported the $C_p$ on the airfoil surface in Figure 25. Once again, the output of
Figure 22. Comparison of the Mach number contour of the NACA 0012, and NACA synthetic for $\alpha = 0^\circ, 10^\circ, 20^\circ$ and $30^\circ$.

Figure 23. Comparison of the $C_p$ on the airfoil surfaces for $\alpha = 0^\circ, 10^\circ, 20^\circ$ and $30^\circ$.

the decoder is in agreement with the RANS solution for the NACA 0020 for both the linear and non-linear regimes.

Figures 22 and 23 show the result of the extrapolation for $t = 10\%$ compared with the CFD results. Also in this case the main physical phenomenon is captured by the autoencoder.

Finally, we performed an extrapolation in $\alpha$ at constant thickness $t = 12\%$. Figure 24 shows the extrapolation result for $\alpha > 30^\circ$. The result highlights that the farther we move away from the training dataset, the

Figure 24. Comparison of the $C_p$ on the NACA 0012 airfoil for $\alpha = 30^\circ - 33^\circ$. 
more the accuracy degrades (consider for instance the airfoil leading edge of the geometry reconstruction).

In these extrapolatory cases, the accuracy is somewhat degraded with the respect to the interpolation case; specifically, the maximum expansion region (the negative $c_p$ peak) is not exactly captured by the decoder in the strongly non-linear part of the phenomenon.

**Conclusions**

The results we presented in this work show that convolutional autoencoders are a powerful tool for the prediction of the aerodynamic performance of airfoils in both linear and non-linear aerodynamic regimes. We studied the sensitivity of the latent space of the trained autoencoder, showing that to pursue a physical interpretation it is necessary to use low-dimensional latent variables and it is preferable to use a database composed of raw data instead of images.

We showed that by using a randomised training dataset in which airfoil thickness and angle of attack vary, the autoencoder is able to automatically learn these parameters by organising the latent space.

The autoencoder is also able to extract other physical information about the stall phenomenon: the latent variables are linear in the linear part of the phenomenon, and non-linear when the stall occurs. Moreover, by using a latent space dimension $p = 1$, the autoencoder learns that symmetric airfoils have the same behaviour for low angles of attack, and respond in a different way depending on their thickness towards the stall.

It is possible to interpolate and extrapolate in the latent space learned by the autoencoder in order to generate accurate aerodynamic predictions and flow fields for synthetic airfoils not seen in the training process.

**Disclosure statement**

No potential conflict of interest was reported by the author(s).

**ORCID**

Ettore Saetta https://orcid.org/0000-0002-3792-6152
Renato Tognaccini https://orcid.org/0000-0003-4528-5540
Gianluca Iaccarino https://orcid.org/0000-0003-3659-7745

**References**

Agostini, L. 2020. “Exploration and Prediction of Fluid Dynamical Systems Using Auto-encoder Technology.” *Physics of Fluids* 32 (6): 067103. doi:10.1063/5.0012906.

Bhatnagar, S., Y. Afshar, S. Pan, K. Duraisamy, and S. Kaushik. 2019. “Prediction of Aerodynamic Flow Fields Using Convolutional Neural Networks.” *Computational Mechanics* 64 (2): 525–545. doi:10.1007/s00466-019-01740-0.

Duraisamy, K., G. Iaccarino, and H. Xiao. 2019. “Turbulence Modeling in the Age of Data.” *Annual Review of Fluid Mechanics* 51: 357–377. doi:10.1146/annurev-fluid-010518-040547.

Duraisamy, K., Z. J. Zhang, and A. P. Singh. 2015. “New Approaches in Turbulence and Transition Modeling Using Data-driven Techniques.” AIAA SciTech, 53rd AIAA Aerospace Sciences Meeting, Kissimmee, FL, USA, January 5–9.

Economon, T. D., F. Palacios, S. R. Copeland, T. W. Lukaczyk, and J. J. Alonso. 2016. “SU2: An Open-Source Suite for Multiphysics Simulation and Design.” *AIAA Journal* 54 (3): 828–846. doi:10.2514/1.J053813.
Fairbanks, H. R., L. Jofre, G. Geraci, G. Iaccarino, and A. Doostan. 2020. “Bi-fidelity Approximation for Uncertainty Quantification and Sensitivity Analysis of Irradiated Particle-laden Turbulence.” Journal of Computational Physics 402: 108996. doi:10.1016/j.jcp.2019.108996. https://www.sciencedirect.com/science/article/pii/S002199-9119307016.

Lee, K., and K. T. Carlberg. 2020. “Model Reduction of Dynamical Systems on Nonlinear Manifolds Using Deep Convolutional Autoencoders.” Journal of Computational Physics 404: 108973. doi:10.1016/j.jcp.2019.108973. https://www.sciencedirect.com/science/article/pii/S0021999119306783.

Li, R., Y. Zhang, and H. Chen. 2022. “Physically Interpretable Feature Learning of Supercritical Airfoils Based on Variational Autoencoders.” AIAA Journal 60 (11): 6168–6182. doi:10.2514/1.J061673.

Li, J., M. Zhang, J. R. R. A. Martins, and C. Shu. 2020. “Efficient Aerodynamic Shape Optimization with Deep-learning-based Geometric Filtering.” AIAA Journal 58 (10): 102415. doi:10.2514/1.J059254.

Matalanis, C. G., P. O. Bowles, S. Jee, B. -Y. Min, A. E. Kuczek, P. F. Croteau, B. E. Wake, et al. 2016. “Dynamic Stall Suppression Using Combustion-Powered Actuation (COMPACT).” NASA Tech. Rep. NF1676L-24901, NASA/CR-2016-219336.

Milano, M., and P. Koumoutsakos. 2002. “Neural Network Modeling for Near Wall Turbulent Flow.” Journal of Computational Physics 182: 1–26. doi:10.1006/jcph.2002.7146.

Newson, A., A. Almansa, Y. Gousseau, and S. Ladjal. 2020. “Processing Simple Geometric Attributes with Autoencoders.” Journal of Mathematical Imaging and Vision 62 (3): 293–312. doi:10.1007/s10851-019-00924-w.

Noack, B. R. 2019. “Closed-Loop Turbulence Control—From Human to Machine Learning (and Retour).” In Proceedings of the 4th Symposium on Fluid Structure-Sound Interactions and Control (FSSIC), edited by Y. Zhou, M. Kimura, G. Peng, A.D. Lucey, and L. Huang, 23–32. Singapore: Springer. doi:10.1007/978-981-10-7542-1_3.

Paszke, A., S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, et al. 2019. “PyTorch An Imperative Style, High-Performance Deep Learning Library.” In Advances in Neural Information Processing Systems 32, 8024–8035. Curran Associates Inc., Red Hook, NY, USA.

Rumsey, C. L., J. P. Slotnick, and A. J. Sclafani. 2019. “Overview and Summary of the Third AIAA High Lift Prediction Workshop.” Journal of Aircraft 56 (2): 621–644. doi:10.2514/1.C034940.

Sekar, V., M. Zhang, C. Shu, and B. C. Khoo. 2019. “Inverse Design of Airfoil Using a Deep Convolutional Neural Network.” AIAA Journal 57 (3): 993–1003. doi:10.2514/1.J057894.

Tangsali, K., V. R. Krishnamurthy, and Z. Hasnain. 2020. “Generalizability of Convolutional Encoder–Decoder Networks for Aerodynamic Flow-Field Prediction Across Geometric and Physical-Fluidic Variations.” Journal of Mechanical Design 143 (5): 051704. doi:10.1115/1.4048221.

Tracey, B., K. Duraisamy, and J. J. Alonso 2015. “A Machine Learning Strategy to Assist Turbulence Model Development.” AIAA SciTech, 53rd AIAA Aerospace Sciences Meeting, Kissimmee, FL, January 5–9. American Institute of Aeronautics and Astronautics.

Verma, S., G. Novati, and P. Koumoutsakos. 2018. “Efficient Collective Swimming by Harnessing Vortices Through Deep Reinforcement Learning.” Proceedings of the National Academy of Sciences 115 (23): 5849–5854. doi:10.1073/pnas.1800923115.

Yan, X., J. Zhu, M. Kuang, and X. Wang. 2019. “Aerodynamic Shape Optimization Using a Novel Optimizer Based on Machine Learning Techniques.” Aerospace Science and Technology 86: 826–835. doi:10.1016/j.ast.2019.02.003.