Computers and Fluids 302 (2025) 106837

Contents lists available at ScienceDirect

Computers and Fluids

journal homepage: www.elsevier.com/locate/compfluid

Disentangled latent spaces for reduced order models using deterministic autoencoders ☆

Henning Schwarz [a,*], Pyei Phyo Lin [a], Jens-Peter M. Zemke [b], Thomas Rung [a]

[a] Institute for Fluid Dynamics and Ship Theory, Hamburg University of Technology, Am Schwarzenberg-Campus 4, 21073 Hamburg, Germany
[b] Institute of Mathematics, Hamburg University of Technology, Am Schwarzenberg-Campus 3, 21073 Hamburg, Germany

ARTICLE INFO

Keywords:
Autoencoder
Reduced order model
Disentangled latent space
Latent variable correlation
Computational fluid dynamics
Aircraft ditching

ABSTRACT

Data-driven reduced-order models based on autoencoders generally lack interpretability compared to classical methods such as the proper orthogonal decomposition. More interpretability can be gained by disentangling the latent variables and analyzing the resulting modes. For this purpose, probabilistic β-variational autoencoders (β-VAEs) are frequently used in computational fluid dynamics and other simulation sciences. Using a benchmark periodic flow dataset, we show that competitive results can be achieved using non-probabilistic autoencoder approaches that either promote orthogonality or penalize correlation between latent variables. Compared to probabilistic autoencoders, these approaches offer more robustness with respect to the choice of hyperparameters entering the loss function. We further demonstrate the ability of a non-probabilistic approach to identify a reduced number of active latent variables by introducing a correlation penalty, a function also known from the use of β-VAE. The investigated probabilistic and non-probabilistic autoencoder models are finally used for the dimensionality reduction of aircraft ditching loads, which serves as an industrial application in this work.

1. Introduction

Machine learning (ML) approaches based on either classical dimension reduction methods like the proper orthogonal decomposition (POD) [1] or autoencoders are frequently used to reduce the dimension of fluid flows in data-driven surrogate models [2–7]. An autoencoder passes the input through a dimension-reducing encoder and subsequently a dimension-increasing decoder. It aims to learn a lower-dimensional representation of the input by forcing the output of the network to be close to the input. When considering two- or three-dimensional data, which is often the case in physics, autoencoders are usually used with convolutional layers to capture local dependencies in the data. To predict the temporal dynamics of unsteady physical phenomena, autoencoders can be combined with networks suitable for time series prediction like a long short-term memory (LSTM) network [8], a transformer [9] and other ML frameworks such as the sparse identification of nonlinear dynamics (SINDy) [10] or approximations to the Koopman operator [11]. The dynamics are then usually learned in the obtained lower-dimensional latent space of the autoencoder [2,4,7,12–17].

By using nonlinear activation functions, an autoencoder performs nonlinear dimension reduction in contrast to the linear POD framework. Due to this, autoencoders usually achieve better reconstruction accuracy and a more compact latent space compared to POD. This comes, however, at the cost of losing interpretability, as the latent variables of the autoencoder are generally not disentangled or not orthogonal. The difficulty is that the influence of individual latent variables is hard to assess if the change in one latent variable inevitably forces a change in another latent variable due to entanglement. Therefore, several studies have recently focused on obtaining disentangled (orthogonal) latent variables in autoencoders to gain interpretability [16,18,19]. There are several strategies to achieve this goal. The β-variational autoencoder (β-VAE) [20], which is based on the variational autoencoder (VAE) [21], is probably the most used option. It is a probabilistic framework that aims at normally distributed latent space variables. A hyperparameter $\beta > 0$ in the loss function controls the importance of reconstructive capability, typically in terms of the mean squared error, relative to a Kullback–Leibler divergence term that assesses the closeness of the normally distributed latent space variables to the standard normal distribution. In contrast, the orthogonal autoencoder (OAE) [22], originally introduced for clustering tasks, enforces orthogonality of the latent variables in the loss function in a deterministic manner. Again, there is a hyperparameter ($\lambda > 0$) that controls the importance of orthogonality in comparison to the

☆ This article is part of a Special issue entitled: 'fusing data and physics' published in Computers and Fluids.

* Corresponding author.
E-mail address: henning.schwarz@tuhh.de (H. Schwarz).

https://doi.org/10.1016/j.compfluid.2025.106837
Received 21 February 2025; Received in revised form 1 August 2025; Accepted 14 September 2025
Available online 19 September 2025

reconstruction quality in the training process. In [23], the OAE was used to provide uncorrelated latent variables for a statistical process control framework.

Note that latent variables do of course not necessarily need to be disentangled. Rather, it is an additional, welcome feature to improve the interpretability of the latent representation, which is often linked to the explainability of an observation and thus supports, for example, targeted design optimization. Disentanglement characterizes a latent space in which changes in individual latent variables are rigidly linked to changes, preferably, in a single relevant aspect, frequently labeled generative parameters or ground-truth generative parameters. Adequate quantification of disentanglement levels has motivated research into meaningful measures for the degree of disentanglement. In particular, in the field of image recognition, some measures have been developed based on tests with simple geometric bodies, e.g., regarding their color, shading, or offsets, to assess the mutual information between the data and the learned representation with respect to these generative factors [24,25]. However, for many practical physics problems, the knowledge of the underlying generative parameters is lacking, which makes it difficult to assess the quality of disentangling along the route outlined in [24–26]. In these cases, correlations between the latent variables are often used for analysis [16,18,19,27].

Work on disentangling the latent variables in the context of low-dimensional representations of fluid dynamic fields has been focusing on β-VAEs. Eivazi et al. [18] proposed using a β-VAE for the extraction of near-orthogonal modes of turbulent flows. More recently, Solera-Rico et al. [16] combined this approach with a transformer to model the dynamics of fluid flows in the obtained disentangled latent space. Kang et al. [19] employ a β-VAE in combination with a Gaussian process regression to construct a reduced-order model (ROM) for transonic flows. After dimension reduction, the input parameters of the full-order simulation are mapped to the disentangled latent space using regression techniques similar to other works [3,5,28–30] to obtain a more interpretable surrogate model. They demonstrate the ability of the β-VAE to use only a few physics-aware latent variables, when trained with a larger than needed latent space dimension. Notice that the phenomenon of VAEs having inactive latent variables was reported before, e.g., [31–33].

In this work, we compare the established β-VAE with two non-probabilistic frameworks, the OAE and an approach we call uncorrelated autoencoder (UAE), which directly uses the correlation matrix of the latent variables to enforce the desired property of a disentangled latent space. Slightly similar proposals to complement the loss function have recently been used for an image style transfer application, where correlations between different channels of encoded feature maps are reduced [34], as well as for the uncorrelated sparse autoencoder, pursuing other encoding goals [35].

Two data sets are used in this work. The first example is a benchmark example for periodic flows that has already been used in publications of similar strategies for disentangled latent space ROM. The second case contains aircraft ditching load data and is used as an industrial application in this work. Ditching is the emergency landing on water, and its analysis is critical for the certification of a commercial aircraft. To reduce the computational effort, simulation-based ditching analysis is usually performed using a one-way coupling between the fluid and the structure, i.e., in a first step the hydrodynamic loads are calculated and subsequently the structural analysis is performed. In the long-term, the current research aims to include approximate deformations to the calculation of the hydrodynamic loads with ML. First steps towards the spatio–temporal prediction of ditching loads using combinations of convolutional autoencoder (CAE) with LSTM networks and Koopman operator approaches that can include deformation caused changes of the loads are documented in [17]. Interpretabilty of the data-driven surrogate model is obviously critical in an industrial process. The goal within this work is to advance the employed ML methods in this regard. To the best of our knowledge, this is the first work for applications related to fluid dynamics that applies a loss function focusing on the correlation matrix of the latent variables and compares the OAE to the β-VAE and the UAE.

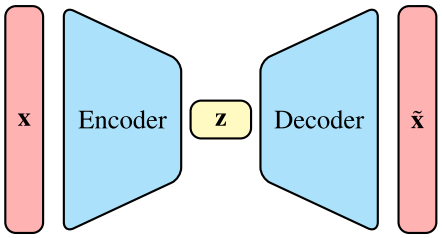

Fig. 1. Basic autoencoder scheme.

The remainder of the paper is structured as follows: In Section 2, the used autoencoder frameworks are briefly presented. Results on both datasets are shown in Section 3 and conclusions are drawn in Section 4.

2. Methods

An autoencoder comprises an encoder and a decoder, as illustrated in Fig. 1. The encoder maps the input $\mathbf{x} \in \mathbb{R}^n$ to a usually lower dimensional latent space representation $\mathbf{z} \in \mathbb{R}^m$ and the decoder subsequently maps $\mathbf{z}$ to the full dimension to obtain a reconstruction of the input, $\tilde{\mathbf{x}} \in \mathbb{R}^n$. Encoder and decoder usually have symmetric structure. All models in this work are based on a CAE, as the considered physical data is two-dimensional in space and convolutional layers are usually better than fully connected layers at capturing spatial dependencies.

A standard autoencoder reconstruction loss comparing input $\mathbf{x}$ and the reconstruction $\tilde{\mathbf{x}}$, i.e., mean squared error (MSE), is considered for all models so that encoder and decoder can be used to decrease or increase the spatial dimension of the data at hand. In this work, the goal of the reduced order models is to obtain disentangled latent variables $\mathbf{z}$ while keeping the reconstruction error as low as possible. To this end, the three models considered, i.e., β-VAE, OAE and UAE, each use a different second term in the loss function that acts in the latent space to effect the disentanglement.

2.1. Orthogonal autoencoder (OAE)

The orthogonal autoencoder (OAE) was introduced by Wang et al. [22] for clustering tasks. In addition to the reconstruction loss, the OAE enforces the orthogonality of the latent variables within the loss function for a mini batch of size b

$$\mathrm{Loss}_{\mathrm{OAE}}(\mathbf{X}, \tilde{\mathbf{X}}) = \frac{1}{b \cdot n} \|\mathbf{X} - \tilde{\mathbf{X}}\|_{\mathrm{F}}^2 + \frac{\lambda}{m^2} \|\mathbf{Z}^T \mathbf{Z} - \mathbf{I}\|_{\mathrm{F}}^2, \quad \lambda > 0, \tag{1}$$

where $\mathbf{X} \in \mathbb{R}^{b \times n}$ contains the inputs for one mini batch, $\mathbf{Z} \in \mathbb{R}^{b \times m}$ is a matrix containing the latent vectors of a mini batch as rows and $\mathbf{I} \in \mathbb{R}^{m \times m}$.

2.2. Uncorrelated autoencoder (UAE)

To measure the independence of the latent variables, the Pearson correlation matrix $\mathbf{R} \in \mathbb{R}^{m \times m}$ of the latent vectors, and its determinant $\det(\mathbf{R})$ are often considered [16,18,19,23,27]. The matrix is computed as follows

$$\mathbf{R}_{ij} = \frac{\sum_{k=1}^{K} (z_i^k - \bar{z}_i)(z_j^k - \bar{z}_j)}{\sqrt{\sum_{k=1}^{K} (z_i^k - \bar{z}_i)^2 \sum_{k=1}^{K} (z_j^k - \bar{z}_j)^2}}, \quad 1 \le i, j \le m,$$

where K is the number of considered training or validation examples used for the calculation, z_i^k denotes the value of the ith latent variable on the kth data example, and $\bar{z}_i$ is the mean value of the ith latent variable over all k examples. The matrix $\mathbf{R}$ is symmetric and has unit

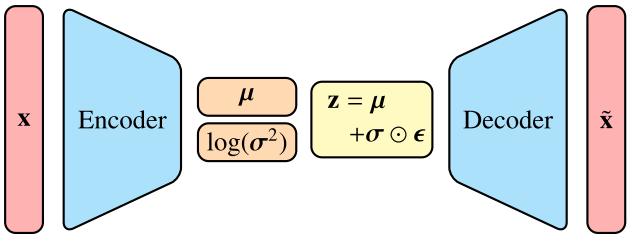

Fig. 2. Scheme of a VAE.

values on the diagonal. A correlation matrix close to the identity matrix and thus a determinant close to one corresponds to a good level of disentanglement.

The second non-probabilistic approach considered here, which we term uncorrelated autoencoder (UAE), uses this criterion directly in the loss function. The employed loss function is similar to the correlation losses in [34,35], where the absolute values of the non-diagonal entries of the correlation matrix are forced to be small. The present study follows the OAE method and uses the squared entries

$$\text{Loss}_{\text{UAE}}(\mathbf{X},\tilde{\mathbf{X}}) = \frac{1}{b\cdot n}\|\mathbf{X}-\tilde{\mathbf{X}}\|_{\text{F}}^2 + \frac{\nu}{m^2}\|\mathbf{R}-\mathbf{I}\|_{\text{F}}^2, \quad \nu > 0, \tag{2}$$

where $\mathbf{R}\in\mathbb{R}^{m\times m}$ is the Pearson correlation matrix of the latent vectors of a mini batch of size b and $\mathbf{I}\in\mathbb{R}^{m\times m}$. The UAE therefore focuses directly on the two properties of interest, reconstruction quality and disentanglement, without forcing the latent space to follow a normal distribution as the (β-)VAE presented next.

2.3. Variational autoencoder (VAE)

The latent space representations in a VAE are forced to follow a probability distribution, which is usually assumed to be normally distributed: $\mathbf{z}\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})$, with mean $\boldsymbol{\mu}\in\mathbb{R}^m$ and assumed diagonal covariance matrix $\boldsymbol{\Sigma}\in\mathbb{R}^{m\times m}$ where the diagonal entries are $(\sigma_i)^2, i\in\{1,\dots,m\}$. The encoder provides the two outputs $\boldsymbol{\mu}$ and $\log(\boldsymbol{\sigma}^2)$, as illustrated in Fig. 2, where for stability reasons the log-variance $\log(\boldsymbol{\sigma}^2)$ is returned. Using $\boldsymbol{\sigma}=\exp(0.5\cdot\log(\boldsymbol{\sigma}^2))$ and generating $\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})$, the latent variables $\mathbf{z}=\boldsymbol{\mu}+\boldsymbol{\sigma}\odot\boldsymbol{\epsilon}$ are computed, which are the input for the decoder. This is the *reparameterization trick* [21] to enable classical backpropagation, as $\mathbf{z}$ can be considered a deterministic variable during training. The gradient with respect to the encoder can otherwise cause problems [21]. The loss function of a VAE is composed of a reconstruction loss, as well as a term that forces the posterior distribution in the latent space to be close to the chosen prior distribution. In this work, these are the MSE and the Kullback–Leibler (KL) divergence, which forces the posterior $\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})$ to be close to the prior $\mathcal{N}(\mathbf{0},\mathbf{I})$. The formula for the loss is

$$\text{Loss}_{\text{VAE}}(\mathbf{X},\tilde{\mathbf{X}}) = \frac{1}{b\cdot n}\|\mathbf{X}-\tilde{\mathbf{X}}\|_{\text{F}}^2 + \frac{\beta}{b\cdot 2}\sum_{i=1}^{b}\sum_{j=1}^{m}(\sigma_{ij}^2+\mu_{ij}^2-1-\log(\sigma_{ij}^2)), \quad \beta>0, \tag{3}$$

where b is the mini batch size, $\mathbf{X}\in\mathbb{R}^{b\times n}$ contains the inputs for one mini batch, and μ_{ij} and σ_{ij}^2 denote the jth component of mean and variance for the ith example in the mini batch. For $\beta=1$, this is referred to as the VAE [21] and $\beta\neq 1$ denotes the β-VAE [20].

The latent space obtained from VAEs or β-VAEs is usually continuous. That is a useful feature for the generation of new data, for which samples of the probability distribution in the latent space are decoded. Because $\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})$ is generated every time the VAE is used to encode data, the trained model is not deterministic. Mind that in the work of Solera-Rico et al. [16], the trained β-VAE is applied in a deterministic manner, as only the mean values $\boldsymbol{\mu}$ are used by the models that learn the dynamics of fluid flows, which seems to be a usual technique in downstream tasks. Using a deterministic model from the beginning could, however, be the simpler approach as the training of VAEs can be challenging.

A common challenge is the so-called *posterior collapse*, in which the posterior matches the prior distribution and the decoder ignores the latent vector $\mathbf{z}$ [31,33,36]. In that case, the model does not learn meaningful representations of the inputs $\mathbf{x}$. Although techniques exist to mitigate this problem, such as including a warm-up phase for the reconstruction loss in the beginning of the training in which β is initially set to 0 and then gradually increased [31,33], or using a different prior distribution, e.g., [37,38], this requires additional consideration for the training of VAEs.

On the contrary, the collapse or inactiveness of several *single* latent variables z_i, which do not help the reconstruction, is a beneficial feature for a disentangled latent space [39] and for surrogate modeling, as this might help to identify the physics-aware latent variables [19].

2.4. Model comparison

The following section attempts to explain the similarities and differences between the three loss function of the three autoencoder models. Let $\mathbf{Z}\in\mathbb{R}^{b\times m}$ contain the b latent space vectors of a mini batch, and $\mathbf{z}_j$, $j=1,\dots,m$ denote the columns of $\mathbf{Z}$. By initialization of the networks, for which typically a uniform distribution centered at zero is used, and the loss functions used, we can assume that all m latent space features have approximately zero mean and a nonzero variance. If the features have zero mean, $\|\mathbf{z}_j\|^2=\langle\mathbf{z}_j,\mathbf{z}_j\rangle = b\text{Var}(z_j^k)$ for all $j=1,\dots,m$. The OAE enforces nearly orthonormal columns of $\mathbf{Z}$, which also affects the variance, as the columns are scaled to have approximately unit length. The KL divergence in the (β-)VAE shifts the latent variables towards zero mean values with unit variances. A combination of both, i.e., latent variables with a zero mean and (scaled) orthonormal columns, results in the minimization of the latent space loss function employed by the UAE: with $i\neq j$, $\mathbf{z}_i^T\mathbf{z}_j=0$ and $\bar{z}_i=\bar{z}_j=0$. Then $\sum_{k=1}^b(z_i^k-\bar{z}_i)(z_j^k-\bar{z}_j)=\sum_{k=1}^b z_i^k z_j^k=0\Rightarrow\mathbf{R}_{i,j}=0$. Vice versa, an identity correlation matrix $\mathbf{R}=\mathbf{I}$ is obtained exactly, if the columns of $\mathbf{Z}$ shifted by their mean are nonzero orthogonal. By assumption the mean will be approximately zero, thus it makes sense to directly minimize the UAE loss.

2.5. Analysis of latent variables

Having disentangled latent variables, the impact of the different variables may be analyzed. This can either be done by decoding latent vectors and analyzing the reconstruction, or solely in the latent space by encoding data from the full dimensional space.

Eivazi et al. [18] use the former approach in conjunction with a β-VAE. They map full order data to the latent space and set all components except the component z_i to zero. Subsequently, the resulting vectors are decoded to obtain the ith modes, and the whole procedure is repeated for all latent variables. Finally, the obtained energy from those reconstructions is calculated and the mode providing the largest energy is identified. If the largest energy is obtained using the jth latent variable, this jth mode is ranked as the first mode. This procedure is then repeated to determine the second mode in the ranking, but without setting z_j to zero, and so on. Kang et al. [19] perform their analysis of the β-VAE latent variables completely in the latent space, which avoids the application of the decoder. They calculate the KL divergence of each latent variable to observe which variables contain useful information. Additionally, they consider the standard deviation of each latent variable to assess, which variables are active [inactive] and thus change [do not change] for varying inputs.

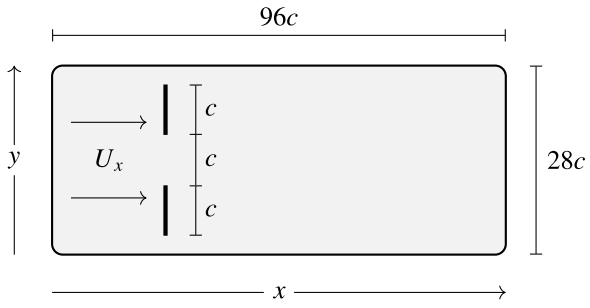

Fig. 3. Illustration of the 2D periodic flow configuration employed for the verification analysis [16,40,42].

2.5.1. Mode generation

Single latent variables can be associated with modes. To this end, the latent variables are ranked as described above to identify the most active latent variables. Inspired by Kang et al. [19], this ranking is performed inside the latent space using the standard deviation and KL divergence of the latent variables. The spatial modes are generated by decoding the ith unit vector of the latent space dimension for some of the highest ranked i [4,16]. However, there is a non-linear behavior in the output when changing the value of z_i, cf. supplementary information of [16]. For this reason, when analyzing the impact of a latent variable, we do not look at a single value for that latent variable, but rather at a range of values while keeping the other latent variables fixed.

3. Results

We first verify the performance of the three autoencoder models in enforcing a disentangled latent space for model reduction of the data extracted from a periodic flow benchmark case suggested by Solera-Rico et al. [16,40]. Subsequently, we apply all models to reduce the load data extracted from aircraft ditching simulations. This more complex industrial test case uses the same data recently used in [17]. All models are implemented in PyTorch [41] and trained on a workstation equipped with an NVIDIA GeForce RTX 3090 GPU.

3.1. 2D periodic benchmark flow

A 2D benchmark case suggested in [16] and previously described in [42] is used for the verification of the models. The configuration is illustrated in Fig. 3. The simulation data includes 1000 equidistant time steps of a periodic flow over two flat plates of length c whose centers are offset by $2c$ in the vertical (y-) direction and approached at an angle of attack of 90°. The first 900 time steps are used for training and the subsequent 100 time steps are used for validation. The homogeneous, unidirectional horizontal inflow velocity U_x results in a Reynolds number of $Re_{c,U_x} = 40$. The computational domain is of size $96c \times 28c$, and the spatial resolution of the data refers to 300×88 homogeneously spaced, equidistant grid points in the horizontal (x-) and vertical (y-) direction, respectively. The temporal resolution of the data refers to $\Delta t = c/U_x$.

Attention will be restricted to the reduction of the horizontal and vertical velocities by the autoencoders. The data of this case is particularly suitable for comparison between machine learning and classical model reduction strategies, since it can be accurately represented with a latent space dimension of $m = 2$. Therefore, the flow field was previously used to compare the results obtained from a β-VAE with a POD approach [16], where the generated modes of the β-VAE were found to be in good agreement with the POD modes. The ability of the models to reduce to two modes and the behavior of the two modes is the focus of this subsection. To safe space, we limit ourselves to the reconstruction of the 11th snapshot of the validation cycle as an exemplary reference snapshot.

3.1.1. Autoencoder configuration

The employed β-VAE agrees with that of Solera-Rico et al. [16], and the decoder structure of the β-VAE was also used for the decoder of the OAE and the UAE, see Table 1. The encoder structure of the OAE and UAE is generally consistent with that of the β-VAE, with one exception being the last layer, which has been modified to have m neurons instead of $2 \cdot m$, since unlike the β-VAE, the OAE and UAE do not require two outputs for the mean and log-variance. Similar to Solera-Rico et al. [16], ELU [43] is the activation function used for the hidden layers in the encoder and the decoder. The output layers of the encoder and the decoder are linear.

The models are trained using Adam [44] with a mini batch size of 256. The training employs 1000 epochs using a 1-cycle learning rate [45] that starts at $1 \cdot 10^{-4}$, increases to $2 \cdot 10^{-4}$ until the 200th epoch and subsequently decreases to $5 \cdot 10^{-6}$ until the 1000th epoch as in [16]. The presented results refer to studies without warm-up. Therefore, in all cases, no pretraining with only the reconstruction loss is performed for the autoencoder, but both the reconstruction loss and the latent space loss are used in every epoch with the corresponding value for ν, λ and β, respectively.

3.1.2. Influence of loss function weights

To compare how the models perform when changing the weight for the disentanglement contribution to the loss function, we train each model with 6 different weighting values 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1} and 10^{0} for β (VAE), λ (OAE) and ν (UAE), respectively. Note that for the β-VAE, the value $\beta = 10^{-3}$ used herein does not correspond to the value of $\beta = 10^{-3}$ also used in [16], since the KL divergence was divided by the latent space dimension of $m = 2$. The latter is concluded from the accompanying code of [16]. Our value of $\beta = 10^{-3}$ is thus two times larger.

We compare the results in terms of (a) the achieved reconstruction losses and (b) the achieved correlation coefficients between the two latent variables. For this comparison, each model is trained five times for each of the six weighting values to include the influence of the randomness in the training process. Fig. 4 shows the reconstruction errors (left) that are obtained on the validation set as well as the absolute values of the correlation coefficients between the two latent variables (right).

In general, the variance of the reconstruction error between the five training runs is small for all autoencoders considered. As the green symbols in the left graph show, the reconstruction error of the β-VAE was not minimized at all during training for the three larger β-values, i.e., $\beta = 10^{0}, 10^{-1}, 10^{-2}$. This indicates too high β-values that lead to a dominating behavior of the KL divergence. Reducing the weight to $\beta = 10^{-3}$, this is not the case anymore and the average reconstruction error decreases by about two orders of magnitude. For $\beta = 10^{-4}$, the error decreases even further and is similarly low as for $\beta = 10^{-5}$. To ensure that the results do not change with or without warm-up, we also performed studies with a KL warm-up phase for 50 epochs, in which β is initially set to 0 and then gradually increased to the specific value. Calculations with and without warm-up strategy indeed yielded very similar results, although the reconstruction loss for $\beta = 10^{-2}$ was minimized to values around 10^{-2} in a few warm-up cases. In combination with $\beta = 10^{-1}$ and $\beta = 10^{0}$ the reconstruction errors obtained with a warm-up strategy at the end of the training are at the same level as depicted in Fig. 4. The reconstruction errors of the other two models show a more robust behavior for the choice of weighting values, with the UAE (blue) delivering lower average reconstruction errors than the OAE (orange) in all cases presented. In none of the cases investigated did the reconstruction loss of the UAE and the OAE models remain at the initial level.

With regard to the correlation coefficients, we can observe a larger variance between the different training runs, especially for the OAE results displayed in orange. The UAE results displayed in blue yield the lowest average correlation coefficient for the four largest values of

Table 1
Structure of encoder and decoder. The kernel size is 3×3, and the stride is 2 for all Conv2d and ConvTranspose2d layers. For the β-VAE, one Linear(4) layer representing $\boldsymbol{\mu}$ and $\log(\boldsymbol{\sigma}^2)$ and the calculation of $\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$ replace the single Linear(2) layer in the encoder.

Encoder	Output shape	Decoder	Output shape
Input	(300, 88, 2)	Linear(256)	(256)
Conv2d(out_channels = 8)	(150, 44, 8)	Linear(2560)	(2560)
Conv2d(out_channels = 16)	(76, 22, 16)	Unflatten(5, 2, 256)	(5, 2, 256)
Conv2d(out_channels = 32)	(38, 12, 32)	ConvTranspose2d(out_channels = 128)	(10, 4, 128)
Conv2d(out_channels = 64)	(20, 6, 64)	ConvTranspose2d(out_channels = 64)	(20, 6, 64)
Conv2d(out_channels = 128)	(10, 4, 128)	ConvTranspose2d(out_channels = 32)	(38, 12, 32)
Conv2d(out_channels = 256)	(5, 2, 256)	ConvTranspose2d(out_channels = 16)	(76, 22, 16)
Flatten()	(2560)	ConvTranspose2d(out_channels = 8)	(150, 44, 8)
Linear(256)	(256)	ConvTranspose2d(out_channels = 2)	(300, 88, 2)
Linear(2)	(2)		

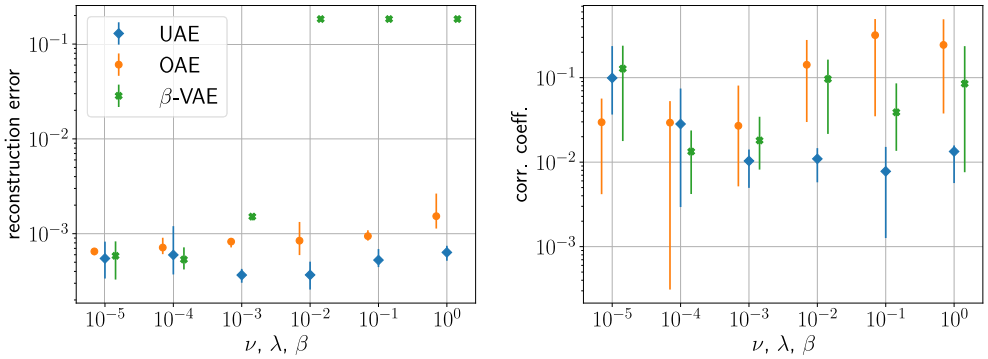

Fig. 4. Reconstruction errors obtained for the validation set and resulting absolute values of correlation coefficients between the two latent variables returned by the three different autoencoder models. The errorbars correspond to average, lowest and highest values obtained from five trainings.

Table 2
Training time and time taken to reconstruct high-dimensional data for the 52800 discrete spatial points from the latent space for the three autoencoder models on the verification case using a single NVIDIA GeForce RTX 3090 GPU.

Model	Training time [s]	Decoder inference time [s]
UAE	206	0.0036
OAE	188	0.0037
β-VAE	176	0.0036

the weight, i.e., $\nu = 10^0, 10^{-1}, 10^{-2}, 10^{-3}$. Although in these cases the second lowest value is returned by the β-VAE, this must be considered under the limitation that the reconstruction error was not minimized for the three largest values of β. For the two smallest weighting values, i.e., 10^{-4} and 10^{-5}, the β-VAE and the OAE, respectively, yield the lowest average correlation coefficient, with the UAE providing the second lowest value for both weights.

The average training and reconstruction times of the three autoencoder models shown in Fig. 4 are given in Table 2. These refer to the use of a single NVIDIA GeForce RTX 3090 GPU. For comparison, training a POD on an Intel Xeon Silver 4314 CPU takes around 117 s. The shortest training time is returned by the β-VAE, taking around 6% less time than the OAE. Training the UAE takes around 10% more time than training the OAE due to the required computations of $\mathbf{R}$. The time taken to apply the decoder is very similar for all models, which could be expected as all models use the same decoder architecture.

3.1.3. Mode analysis for small latent space dimension

To analyze the impact of the latent variables, we generate an exemplary snapshot, verify its reconstruction quality and subsequently change one latent variable, while keeping the other latent variable fixed. This procedure is done for both latent variables (z_1, z_2) and all three models, i.e., UAE, OAE and β-VAE. Fixing the second latent variable can be done in two ways. On the one hand, one can use the value corresponding to the exemplary snapshot, as in [19,20,39]. On the other hand, the other latent variable can simply be set to zero. However, because the investigated periodic benchmark flow is quite simple and does not contain many significantly different states, we choose to keep the exemplary value of one latent variable and observe how the other latent variable changes the state.

Figs. 5–9 display the baseline results obtained in combination with $\nu = 10^{-2}$ for the UAE, and $\lambda = \beta = 10^{-4}$ for the OAE and β-VAE for the reference snapshot. This choice of weights should ensure a satisfactory reconstruction error and a significant level of disentanglement. Fig. 5 compares the true values with the autoencoder-based reconstructions for the horizontal u-velocity (left) and the vertical v-velocity (right) in an upstream portion of the domain. The color coding used is indicated in the figure and is consistent for all u- and v-velocity images, respectively. The figure reveals that only small reconstruction errors occur in conjunction with any of the three models for the chosen exemplary snapshot.

In Figs. 6–9, one of the two latent variables, outlined by the dark background color, is changed, while the other latent variable, indicated by the light background color, remains unchanged. The imposed changes of the variable values refer to the upper, lower and mean value of the entire value range. Displayed results are confined to a spatial window ranging from $-1 \leq x \leq 15$ and $-5 \leq y \leq 6$ to focus on the region of the most significant influences, and the employed color codes agree. Looking at the u-velocity depicted in Figs. 6–7, all models show a similar behavior. One latent variable, i.e., the first for the UAE and the OAE and the second for the β-VAE seems to induce a shift of the u-velocity pattern in horizontal (x-) direction, while the other seems to induce a shift in vertical (y-) direction including a corresponding wave shaped transition that is observed during the initial phase, cf. center graphs in Fig. 7. With attention given to the v-velocity depicted in Figs. 8–9, no vertical displacement is observed in response to any changes of the latent variables. Instead, only horizontal shifts can be observed. The changes induced by z_1 (UAE, OAE) and z_2 (β-VAE) are much lower than in the other case.

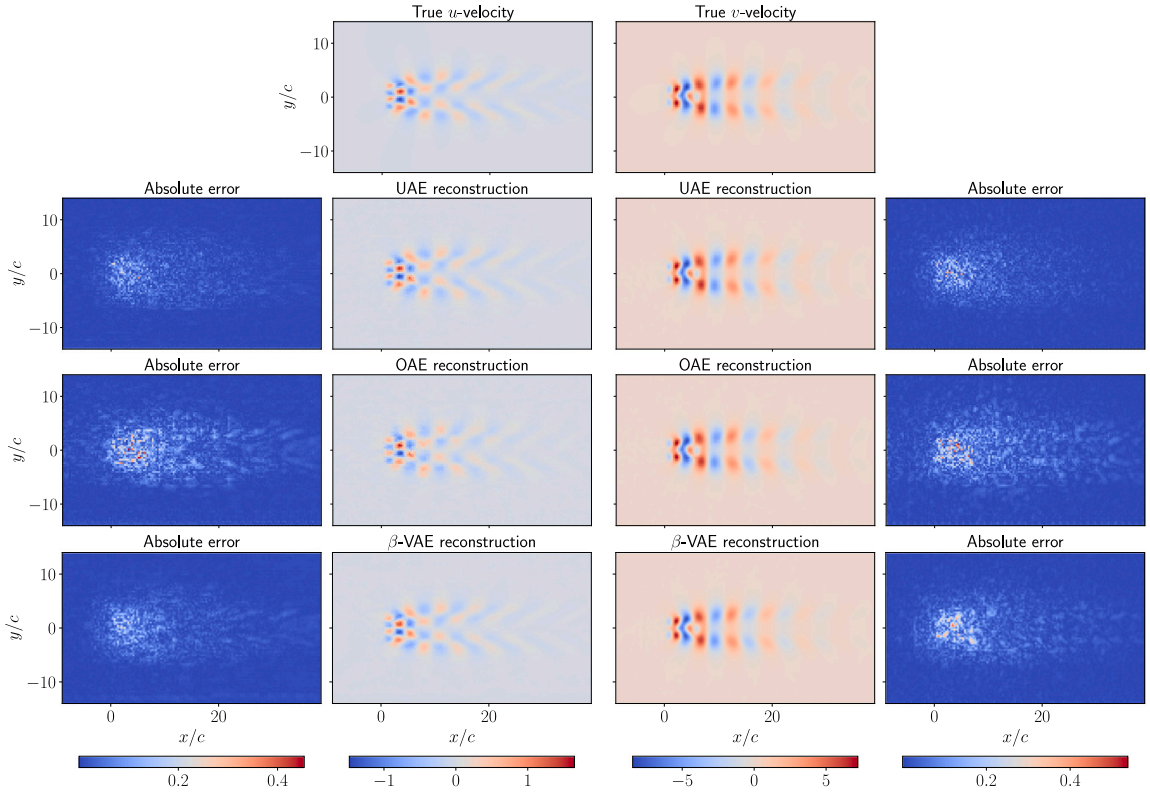

Fig. 5. True u- and v-velocity fields, model reconstructions and related errors for the exemplary reference snapshot of the verification case.

Fig. 10 displays the latent trajectories of the three models. For the β-VAE, the trajectories for both $\mathbf{z}$ and $\boldsymbol{\mu}$ are shown, which largely agree. Overall, the three models return similar results and the two latent variables show a nearly periodic behavior centered around zero, which could be expected as the flow is also periodic and symmetric. The main difference lies in the amplitude, which was already evident from Figs. 6–9. As mentioned in Section 2.4, the OAE enforces orthonormal columns of $\mathbf{Z}$, being the matrix containing the latent vectors of a mini batch as rows. This leads to the OAE latent variables having the smallest amplitudes. The amplitude of the UAE trajectories is in this case approximately twice as high as that of the β-VAE.

3.1.4. Physics awareness for higher latent space dimension

For the periodic flow at hand, two modes or latent variables are enough to capture (most of) the underlying physics [16]. Solera-Rico et al. [16] emphasize that a β-VAE with a larger latent space dimension $m > 2$ still only finds two meaningful modes. This is in line with the findings by Eivazi et al. [18] and Kang et al. [19], where the β-VAE is shown to find physics-aware latent variables. To assess the physics-awareness properties of the deterministic autoencoder models, we train all three models with a latent space dimension of $m = 10$ and analyze whether they can learn using only two disentangled latent variables. Inspired by Kang et al. [19], we employ the standard deviation over the validation dataset (for all models) and the KL divergence (only for the β-VAE) of the latent variables over the validation dataset to judge which latent variable is active, i.e., physically significant. In the context of the deterministic UAE and OAE models, the only possible way to calculate the standard deviation for the individual latent variables is to apply the encoder to the dataset and compute the standard deviation over all examples for each latent component z_i. In contrast, within the framework of the β-VAE, various strategies for calculating the standard deviation and the KL divergence are conceivable. One option is to perform the calculation analogously to the procedure for UAE and OAE. However, computing the standard deviation from $\mathbf{z}$ contains the noise from the computation of $\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$, which can be detrimental for the subsequent analysis. Alternatively, the standard deviation of the mean $\boldsymbol{\mu}$ can be computed to assess how the mean changes. In the present study, the KL divergence is computed per example using the obtained mean $\boldsymbol{\mu}$ and log-variance $\log(\boldsymbol{\sigma}^2)$ from the encoder and for all latent components the mean over all examples is computed afterwards. Subsequently, we generate and compare the modes obtained from the latent variables featuring the four highest standard deviations for an appropriate weighting value along the route outlined in Section 3.1.3.

Fig. 11 depicts the standard deviations of the considered ten latent variables in combination with three different weights of the disentanglement contribution to the loss function. The investigated weighting values were taken from the successful studies for $m = 2$ latent variables, cf. Fig. 4. They refer to $\nu = 10^0$, $\nu = 10^{-1}$ and $\nu = 10^{-2}$ for the UAE, $\lambda = 10^{-3}$, $\lambda = 10^{-4}$ and $\lambda = 10^{-5}$ for the OAE, as well as $\beta = 10^{-3}$, $\beta = 10^{-4}$ and $\beta = 10^{-5}$ for the β-VAE.

Considering the UAE results on the left of Fig. 11, we observe that for $\nu = 10^0 = 1$ and $\nu = 10^{-1}$ two latent variables – i.e., z_1 and z_5, as well as z_1 and z_7, respectively – have noticeably higher standard deviations than the others. In conclusion, these two latent variables change their values more for different inputs to the encoder and are therefore more active than the other latent variables. This in turn suggests that the model has successfully extracted the two essential variables required to describe this flow, although this remains to be verified by analyzing their modes. For $\nu = 10^{-2}$ the correlation penalty is less strongly involved during training, and four latent variables are more active than the rest. This behavior is probably explained by the balance between the numerator and the denominator inside the loss function (2) forcing a latent variable to be close to its mean over all inputs. A smaller ν/m^2 allows more active latent variables as they are less strongly forced to be close to their mean over all inputs. For ($m = 10$, $\nu = 1$) and ($m = 10$, $\nu = 10^{-1}$), the MSE and the correlation penalty are balanced in a way that the UAE uses a sufficient amount of two latent variables for the reconstruction. For the OAE depicted in the center graphs of

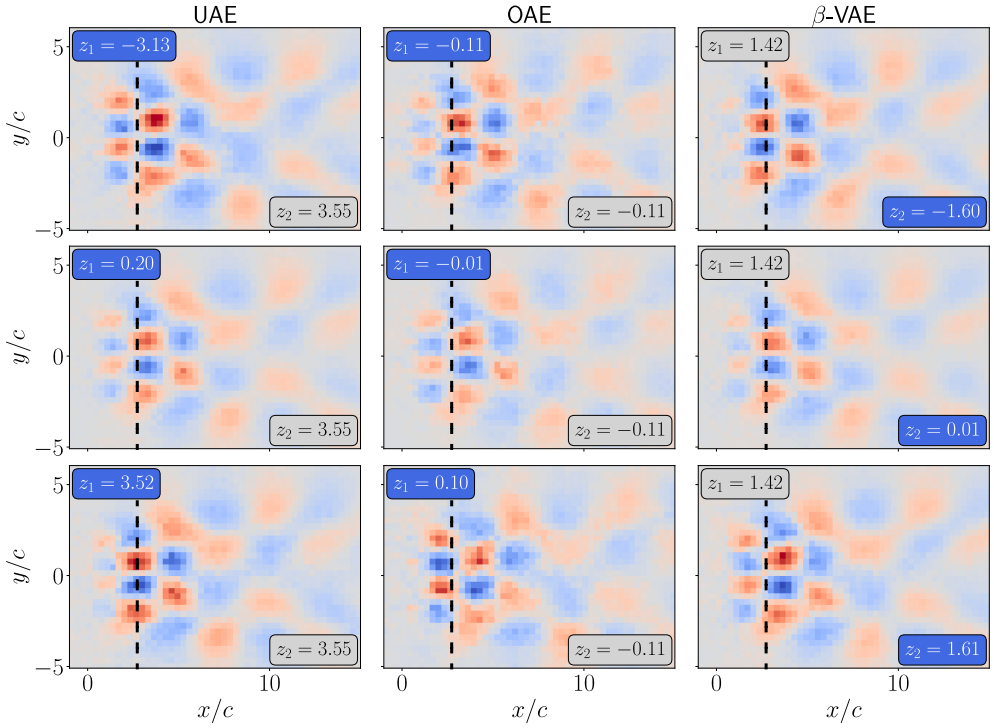

Fig. 6. Impact of changing z_1 for the UAE/OAE and z_2 for the β-VAE on the u-velocity in the reference snapshot of the verification case using $m = 2$ latent variables. The changed latent variable takes equidistant values ranging from the minimum to the maximum obtained for that variable on the validation set. The other latent variable is fixed to the corresponding latent space representation of the reference snapshot. The color code is the same for all examples, ranging from −1.87 (dark blue) to 1.90 (dark red).

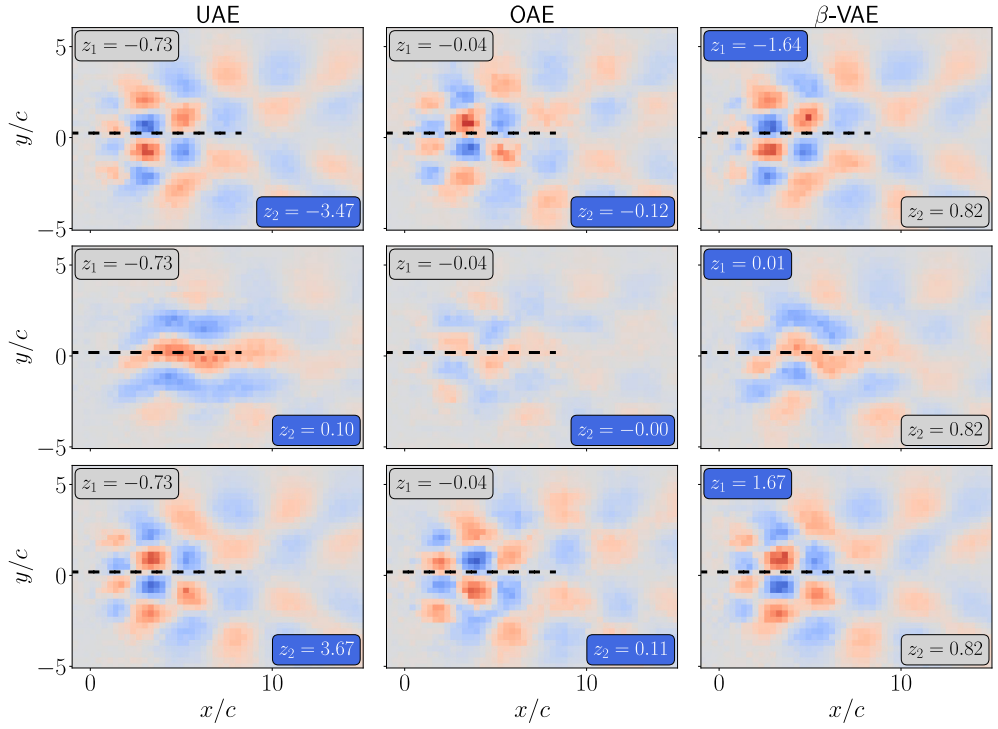

Fig. 7. Impact of changing z_2 for the UAE/OAE and z_1 for the β-VAE on the u-velocity in the reference snapshot of the verification case using $m = 2$ latent variables. The changed latent variable takes equidistant values ranging from the minimum to the maximum obtained for that variable on the validation set. The other latent variable is fixed to the corresponding latent space representation of the reference snapshot. The color code is the same for all examples, ranging from −1.87 (dark blue) to 1.90 (dark red).

Fig. 11, the standard deviations of the different latent variables are generally closer to each other. For the three displayed values of λ, two latent variables with a clearly higher standard deviation than the others cannot be identified, indicating that the OAE has difficulties to

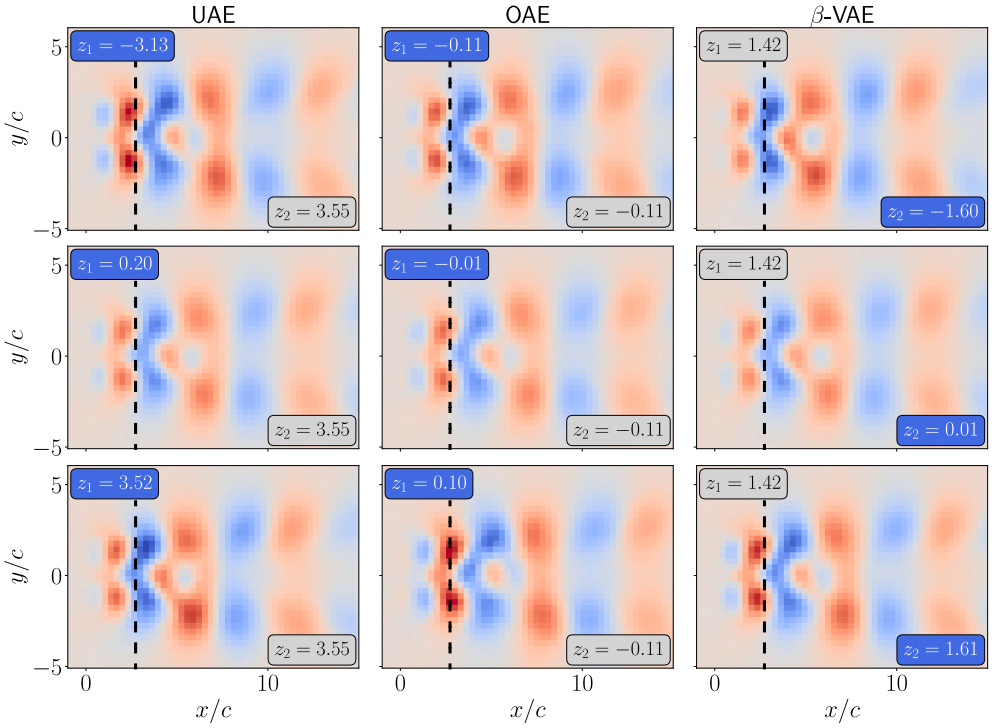

Fig. 8. Impact of changing z_1 for the UAE/OAE and z_2 for the β-VAE on the v-velocity in the reference snapshot of the verification case using $m = 2$ latent variables. The changed latent variable takes equidistant values ranging from the minimum to the maximum obtained for that variable on the validation set. The other latent variable is fixed to the corresponding latent space representation of the reference snapshot. The color code is the same for all examples, ranging from −11.58 (dark blue) to 10.41 (dark red).

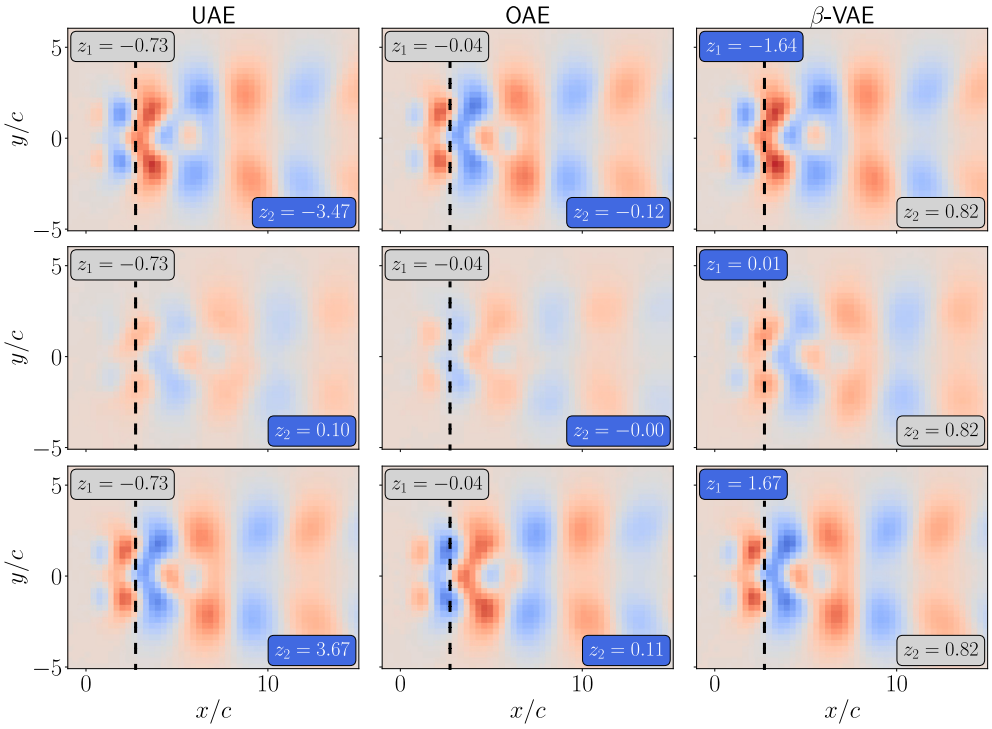

Fig. 9. Impact of changing z_2 for the UAE/OAE and z_1 for the β-VAE on the v-velocity in the reference snapshot of the verification case using $m = 2$ latent variables. The changed latent variable takes equidistant values ranging from the minimum to the maximum obtained for that variable on the validation set. The other latent variable is fixed to the corresponding latent space representation of the reference snapshot. The color code is the same for all examples, ranging from −11.58 (dark blue) to 10.41 (dark red).

extract the two most crucial latent variables when trained with a latent space dimension of $m = 10$ in combination with these λ values. In combination with $\lambda = 10^{-4}$ the largest standard deviation refers to z_1, z_3 and z_5. Results for the β-VAE are shown in the right graphs of Fig. 11.

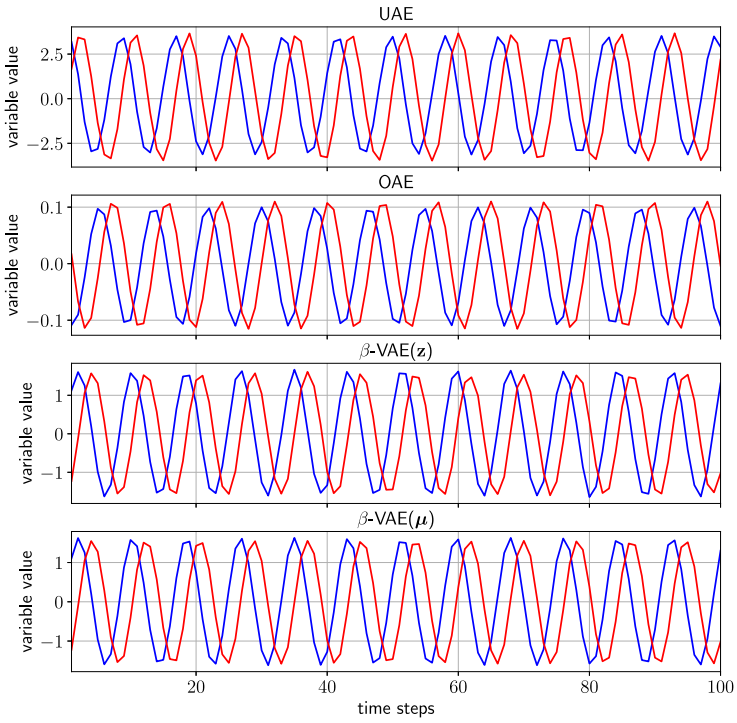

Fig. 10. Latent trajectories of the three autoencoder models over all time steps of the validation set for the verification case using a latent space dimension of $m = 2$. Colors refer to blue [red] for the first [second] component.

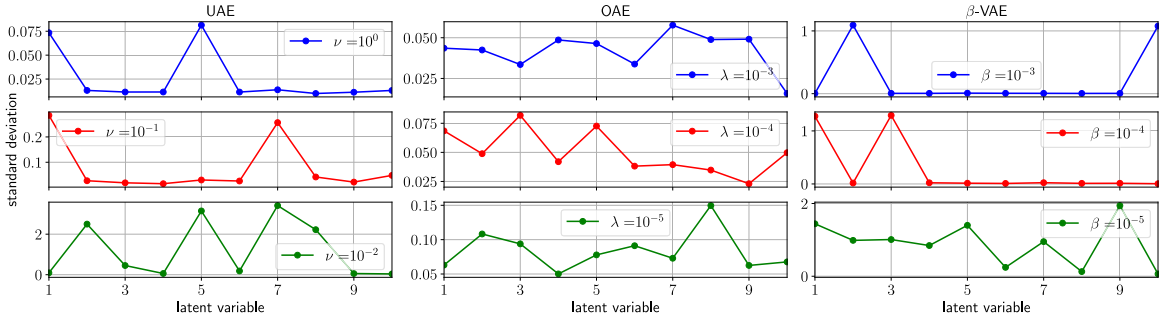

Fig. 11. Standard deviations of latent variables over the validation set for all three models obtained in the verification case for a latent space dimension of $m = 10$. The standard deviations for the β-VAE are computed over the means $\boldsymbol{\mu}$ obtained from the encoder. Each model is trained with three different weights of the disentanglement contribution to the loss function.

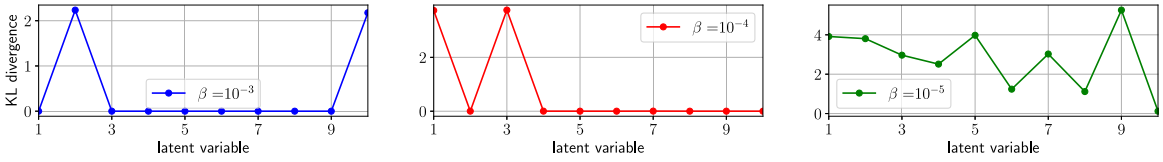

Fig. 12. KL divergence of the latent variables over the validation set for the β-VAE obtained in the verification case for a latent space dimension of $m = 10$. The model is trained with three different values of β in the loss function.

For the large value $\beta = 10^{-3}$, the standard deviation of the mean $\boldsymbol{\mu}$ is close to zero for all but two latent variables, suggesting that two active variables have been identified and the other eight variables do only contribute redundant information or noise. This is confirmed by the KL divergence shown in Fig. 12, which is close to zero for these eight variables, corresponding to $\boldsymbol{\mu} \approx 0$ and $\boldsymbol{\sigma} \approx 1$. For $\beta = 10^{-4}$, μ_1 and μ_3 have a higher standard deviation than the other latent variables, which have standard deviations are close to 0. Accordingly, the KL divergence for the two variables z_1 and z_3 increases and is approximately zero for the rest. For $\beta = 10^{-5}$, the standard deviation of the mean $\boldsymbol{\mu}$ is greater than zero for most latent variables. Moreover, the KL divergence is visibly greater than 0 for all but one variable, suggesting that β is too small to separate the two crucial modes.

We note that the UAE and the β-VAE showed robustness in the ability to extract two active latent variables over the course of multiple training runs for $\nu = 1$ and $\nu = 10^{-1}$ as well as $\beta = 10^{-3}$ and $\beta = 10^{-4}$, respectively. The UAE was also able to extract two active latent variables with a larger value of $\nu = 10$ in our experiments, whereas the reconstruction loss of the β-VAE was not minimized for a larger value of $\beta = 10^{-2}$.

To verify how the modes for the active and inactive latent variables differ, we generate modes for the four most active latent variables of

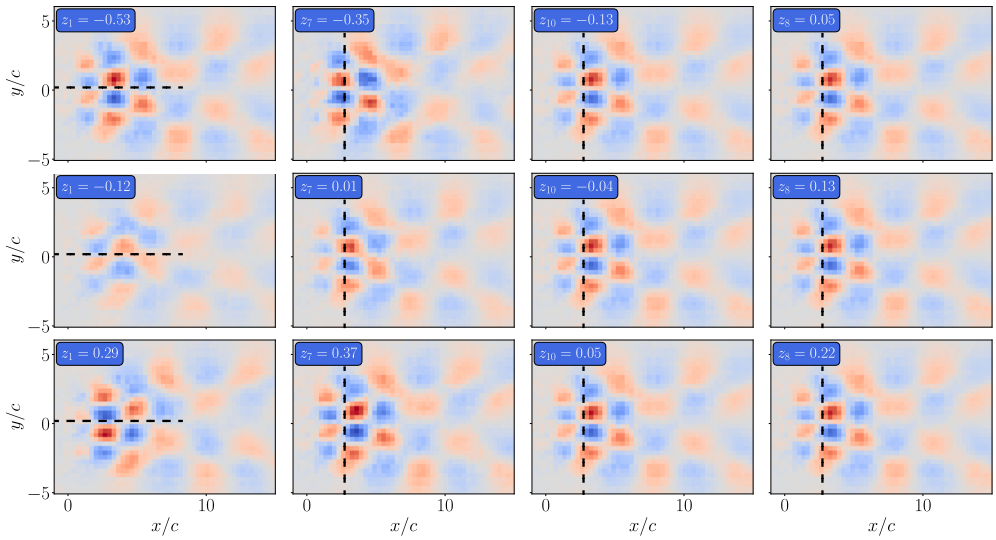

Fig. 13. Impact of single latent variables on the u-velocity in the reference snapshot of the verification case for the UAE using a latent space dimension of $m = 10$ and $\nu = 10^{-1}$. In each column, nine latent variables are fixed to the corresponding latent space representation of the snapshot and one variable takes equidistant values ranging from the minimum to the maximum value obtained for the validation set. Color code is the same for all examples in this figure, ranging from −1.65 (dark blue) to 1.66 (dark red).

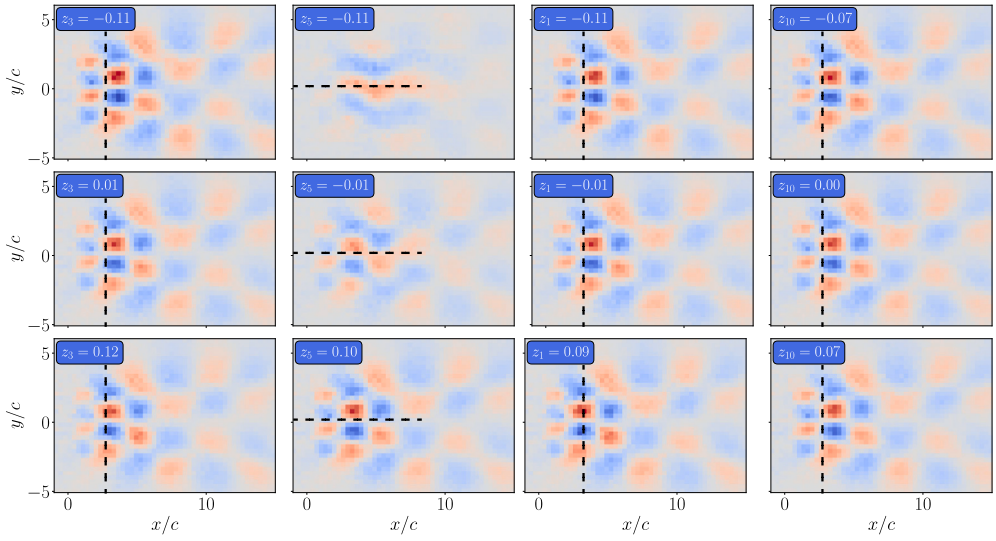

Fig. 14. Impact of single latent variables on the u-velocity in the reference snapshot of the verification case for the OAE using a latent space dimension of $m = 10$ and $\lambda = 10^{-4}$. In each column, nine latent variables are fixed to the corresponding latent space representation of the snapshot and one variable takes equidistant values ranging from the minimum to the maximum value obtained for the validation set. Color code is the same for all examples in this figure, ranging from −1.65 (dark blue) to 1.74 (dark red).

the latent space dimension $m = 10$ in combination with the intermediate weight values, i.e., the red curves in Fig. 11 for the UAE and the OAE, and the small value (blue curve) for the β-VAE. The standard deviation is used for identification in combination with all models. To safe space, we restrict ourselves to the influence of the modes on the horizontal u-velocity fields in a confined spatial domain as outlined in Figs. 13–15. Consideration of the vertical velocity leads to the same conclusions.

Fig. 13 shows the UAE modes obtained for $\nu = 10^{-1}$, where default values for $m = 10$ read $z_1 = -0.47$, $z_7 = 0.19$, $z_{10} = -0.10$ and $z_8 = 0.20$. The modes for the most active latent variable z_1 displayed in the first column clearly induce a shift in vertical direction. Similarly, the modes of the second most active variable z_7 displayed in the second column introduce a horizontal shift. The behavior is similar to that observed in the first columns of Figs. 6 and 7. For the less active latent variables z_{10} and z_8, the outputs are nearly identical for every latent value tested. This supports the assumption that the UAE model successfully identified the two crucial latent variables while being trained with a latent dimension of $m = 10$. The inactiveness of latent variables in the UAE is in line with the findings by Kim et al. [34], in which the correlation penalty leads to inactive channels of encoded feature maps.

Fig. 15 displays the modes for the four most active latent variables of the β-VAE in combination with $\beta = 10^{-3}$, where default values for $m = 10$ are $z_2 = -1.18$, $z_{10} = 0.90$, $z_7 = 2.18$ and $z_5 = -1.10$. Again, the modes for the most active latent variables z_2 and z_{10} display strong similarities to the modes investigated for $m = 2$ in Figs. 6 and 7. Varying the value of the third most active latent variables z_7 and z_5 does not significantly change the output of the decoder.

For the OAE we obtain slightly different results, as Fig. 14 shows for the mode study based on $\lambda = 10^{-4}$. Here default values for $m = 10$ refer to $z_3 = -0.01$, $z_5 = 0.10$, $z_1 = -0.03$ and $z_{10} = -0.03$. Horizontal shifts induced by the highest ranked latent variable z_3 and the third highest ranked latent variable z_1 are in close agreement. They are

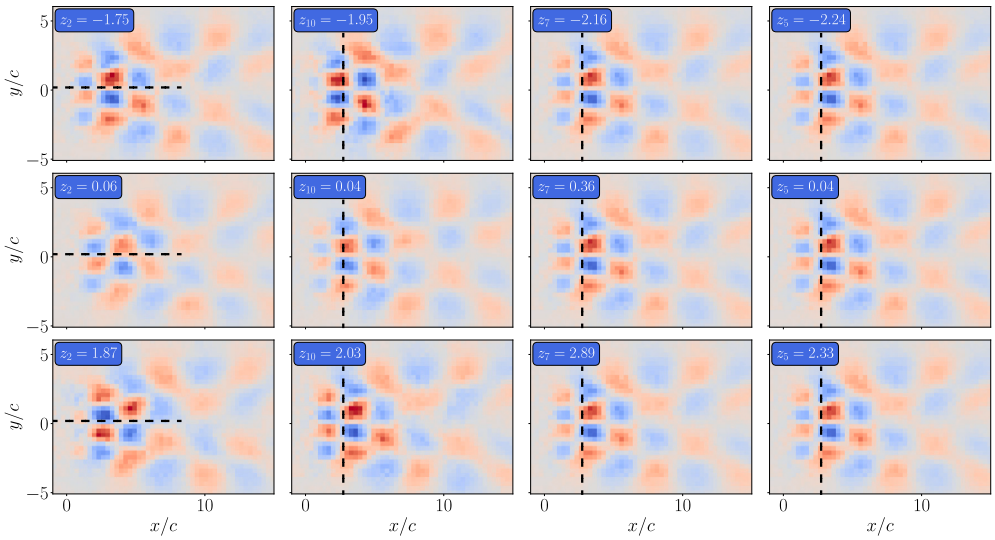

Fig. 15. Impact of single latent variables on the u-velocity in the reference snapshot of the verification case for the β-VAE using a latent space dimension of $m = 10$ and $\beta = 10^{-3}$. In each column, nine latent variables are fixed to the corresponding latent space representation of the snapshot and one variable takes equidistant values ranging from the minimum to the maximum value obtained for the validation set. Color code is the same for all examples in this figure, ranging from −1.66 (dark blue) to 1.59 (dark red).

supplemented by vertical shifts introduced through changing the value of the second highest ranked variable z_5. The influence of z_{10} on the encoder data is minor. Although the results are consistent with the data shown for $m = 2$, the reduction to two modes is not so obvious in this case.

Fig. 16 illustrates the latent trajectories corresponding to the four most active latent variables of the three autoencoder models when trained with a latent space dimension of 10. With regards to the UAE in the first row, the trajectories of the two most active latent variables generally evolve similarly as in Fig. 10, being slightly shifted in time, which can also be interpreted as a multiplication with −1. However, the amplitudes are smaller and the center for z_1 (blue curve) is also slightly displaced in negative direction. The trajectory amplitudes for the two less active latent variables are significantly smaller, leading to the smaller standard deviations in Fig. 11. A similar result can be observed for the OAE, although the difference between the amplitudes of the two most active and the two less active latent variables is, in accordance with Fig. 11, significantly less pronounced. For the β-VAE, a substantial difference can be observed between the trajectories for $\mathbf{z}$ and $\boldsymbol{\mu}$. The trajectories for $\boldsymbol{\mu}$ for the two most active variables are similar to the ones in Fig. 10, apart from a shift in time or a multiplication with −1. The two less active variables are zero for every time step, matching the standard deviations and KL divergences in Figs. 11 and 12. With respect to $\mathbf{z}$, similarly small differences in the amplitudes for the two most active variables occur as with respect to $\boldsymbol{\mu}$. Since the KL divergence is approximately zero for the less active variables, their trajectories only consist of Gaussian noise.

Fig. 17 depicts the evolution of the reconstruction and latent space losses over the course of the training of the three autoencoder models. In general, the KL divergence of the β-VAE is around one order of magnitude higher than the latent space losses of the UAE and the OAE. After passing through the initial plateau of values, the reconstruction loss of the UAE drops to a slightly higher level than the OAE, but lower than the level of the β-VAE.

3.1.5. Pruning

In contrast to separating active from inactive latent variables during a post-processing step after the training, inactive latent variables could also be identified and pruned during the training process. The ith latent variable is pruned by setting the ith row of the weight matrix and the ith component of the bias of the final layer in the encoder to zero. By pruning inactive latent variables during training, these variables do not take part in the reconstruction of the input anymore and the models learn to only use the active latent variables. We tested such an approach for the UAE with a latent space dimension of $m = 10$, in which an experiment-based threshold value of 0.07 for the standard deviation of the normalized latent variables was employed to identify inactive latent variables, starting from the 500th epoch. A normalized threshold value slightly above 0.05 can also be used to identify the active latent variables in a post-processing step.

The results were compared to a post-processing approach, where the two most active latent variables are identified after the training and the other latent variables are set to their mean, as for example suggested in [19]. We can confirm that the pruning approach successfully identified the two most active latent variables in several cases. However, no significant improvement of the reconstruction accuracy and the correlation coefficient was achieved compared to the post-processing approach. Further work on the pruning approach will be conducted in future research.

3.2. Industrial application: Aircraft ditching loads

The second case serves to validate the use of the autoencoder models for more complex problems. The particular application refers to the data-driven surrogate model of ditching load and deformation predictions. In a previous study [17], different combinations of CAE and LSTM networks well as a Koopman autoencoder [2] were assessed to predict the spatio–temporal evolution of ditching loads, with a *convolutional encoder–LSTM–convolutional decoder* combination yielding the best results. In this paper, the focus lies on the disentanglement and analysis of the latent variables.

The data set contains simulated ditching loads on the fuselage of a DLR-D150 aircraft, which is of similar size to an Airbus A320. The simulations were performed with the 3 degrees of freedom method *ditch* [46], which considers horizontal, vertical and pitch motion of the aircraft in response to hydrodynamic loads, aerodynamic loads, inertia forces and thrust. Due to the resulting symmetry, only one half of the fuselage is used. A detailed description of the data generation and processing is given in [17]. The training set contains data from 323 simulations, which differ in the horizontal and vertical initial velocities.

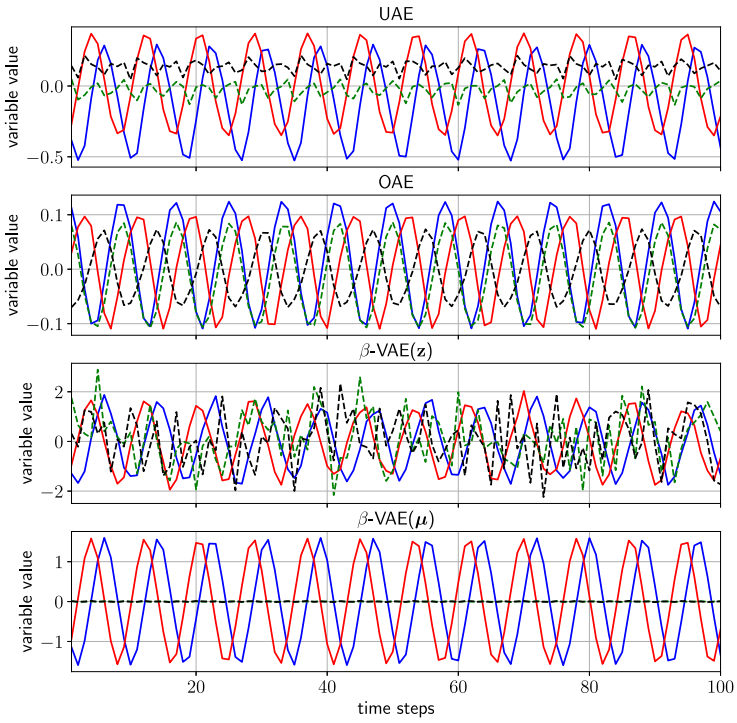

Fig. 16. Latent trajectories of the three autoencoder models for a latent space dimension of $m = 10$ and $\nu = 10^{-1}$, $\lambda = 10^{-4}$, $\beta = 10^{-3}$, respectively, over all time steps of the validation set for the verification case. Colors blue, red, green and black refer to the most active components in descending order, cf. Fig. 11.

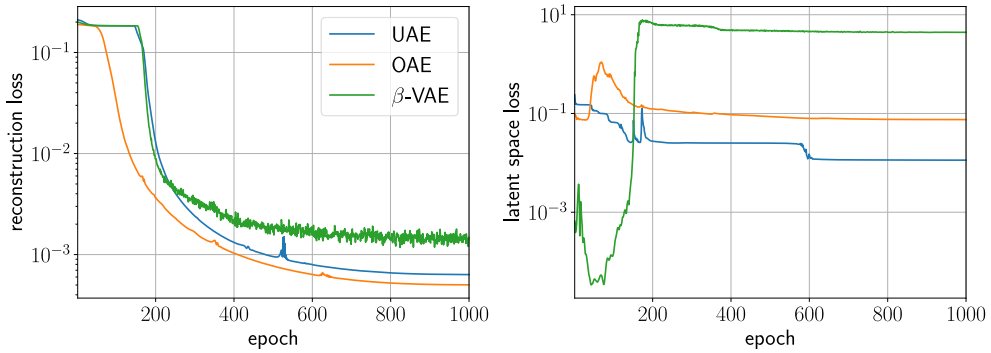

Fig. 17. Evolution of the loss functions on the validation set of the verification case with $\nu = 10^{-1}$, $\lambda = 10^{-4}$ and $\beta = 10^{-3}$ for the UAE, OAE, β-VAE, respectively.

A validation set containing data from 20 simulations is used to compare the models for different weighting values of the disentanglement contributions to the loss functions. A test set comprising 30 additional simulations is subsequently used to compare the performance when the models are combined with an LSTM network for spatio–temporal predictions. For each simulation, around 18–35 time step images of size 128×128, representing equidistantly spaced circumferential and longitudinal coordinates of the fuselage, are included in the data sets. The equidistant time step corresponds to 0.1 s.

The assessment of the autoencoder models is based on typical results for a temporal load series that were obtained from the trained models, see also [17]. An exemplary temporal development of the hull loads is shown in Fig. 18 and shows the five typical phases. The first impact phase usually results in an initially confined circumferential load area (initial phase; 1st step), followed by a widening towards a larger longitudinal and circumferential load area (early phase; 5th, 9th step). For the subsequent phases, positive load signatures are usually slender crescent-shaped and limited to a smaller spatial area (late phase; 13th, 17th step). Moreover, they are followed by a more pronounced under pressure region in downstream direction. The final loads of rear fuselage are often small in amplitude and area (final phase; 21st step).

3.2.1. Autoencoder configuration

The structure of the encoder and the decoder is based on the previous study [17] and outlined in Table 3. The baseline study employs a latent space dimension of $m = 10$, which lead to satisfactory results in the previous research [17], while keeping the latent space compact to allow for future research on interpretability. A LeakyReLU [47] activation function with slope parameter $\alpha = 0.01$ was employed for the hidden layers in the encoder and the decoder. The output layers of the encoder and the decoder are linear. Models are trained for 500 epochs using Adam with the default learning rate of $1 \cdot 10^{-3}$ and a minibatch size of 128.

In a recent study of the authors [17], it was demonstrated that POD reconstructions of the loads based on 100 modes still contain residual oscillations. Therefore, 100 modes seem to be too few to capture the high-frequency content. The focus in this work thus lies entirely on autoencoder-based dimension reduction, expecting that much fewer latent variables are required.

3.2.2. Influence of loss function weights

Similar to the verification study in Section 3.1.2 we compare the reconstruction losses on the validation set and use the determinant of

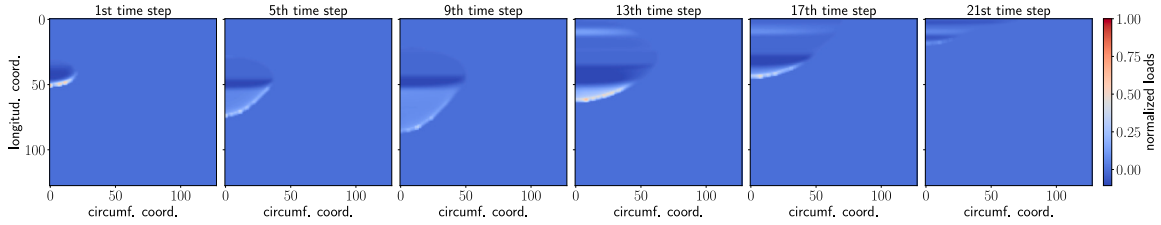

Fig. 18. Exemplary spatio–temporal evolution of simulated ditching loads.

Table 3
Structure of encoder and decoder. The kernel size is 3×3, and the stride is 2 for all Conv2d and ConvTranspose2d layers. For the β-VAE, two Linear(10) layers representing $\boldsymbol{\mu}$ and $\log(\boldsymbol{\sigma}^2)$ and the calculation of $\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$ replace the single Linear(10) layer in the encoder.

Encoder	Output shape	Decoder	Output shape
Input	(3, 128, 128, 1)	Linear(4096)	(4096)
Conv2d(out_channels = 8)	(64, 64, 8)	Unflatten(8, 8, 64)	(8, 8, 64)
Conv2d(out_channels = 16)	(32, 32, 16)	ConvTranspose2d(out_channels = 32)	(16, 16, 32)
Conv2d(out_channels = 32)	(16, 16, 32)	ConvTranspose2d(out_channels = 16)	(32, 32, 16)
Conv2d(out_channels = 64)	(8, 8, 64)	ConvTranspose2d(out_channels = 8)	(64, 64, 8)
Flatten()	(4096)	ConvTranspose2d(out_channels = 1)	(128, 128, 1)
Linear(10)	(10)		

the correlation matrix $\det(\mathbf{R})$ to simultaneously measure the disentanglement of the latent variables for different weights β, λ and ν of the disentanglement contribution to the loss function. Again, the presented results refer to studies without warm-up and both contributions to the loss function are used in every epoch with the corresponding value for ν, λ and β, respectively. To obtain some guidance on judging the results for the different β, λ and ν values, we additionally trained a standard convolutional autoencoder without any disentanglement constraints in the latent space, and obtained $\det(\mathbf{R}) = 0.014$ and a reconstruction error of $2.53 \cdot 10^{-6}$.

Fig. 19 depicts the influence of the weights on the reconstruction error (left) and the determinant of the correlation matrix (right). The two deterministic models again reveal a robust behavior. For the investigated range of values for ν and λ, the minimization of the reconstruction loss does not get stuck in the beginning, although higher values usually yield higher reconstruction errors. The UAE, represented by the blue symbols, achieves the lowest reconstruction error and returns $\det(\mathbf{R}) \approx 1$ except for $\nu = 10^{-6}$ and $\nu = 10^{-7}$, where $\det(\mathbf{R})$ is close to zero, suggesting that $\nu \leq 10^{-6}$ is too small to enforce uncorrelated latent variables. For the OAE, marked in orange, $\det(\mathbf{R})$ is mostly close to 0.9 and smaller than $\det(\mathbf{R})$ for the UAE when $\lambda \geq 10^{-4}$. Although $\det(\mathbf{R})$ values associated with $\lambda \leq 10^{-6}$ are higher for the OAE than for the corresponding UAE, the determinant only reaches values about 0.4 and 0.25. These values are quite small, indicating an undesirable correlation level between the latent variables. On the contrary, one can observe that the β-VAE represented by the green symbols, only minimizes the reconstruction loss when β is set smaller than $\beta = 10^{-5}$, which is the largest value, for which the minimization of the reconstruction loss does not get stuck in the beginning. For $\beta = 10^{-4}$, we also tested a warm-up phase, in which the β-value was gradually increased to 10^{-4} at the beginning of training. Without going into detail we note that we obtained virtually the same reconstruction error level as for the studies without warm-up. In all cases, the reconstruction error for the β-VAE exceeds the reconstruction error of the deterministic models. The values for $\det(\mathbf{R})$ are greater than 0.9 for $\beta = 10^{-5}$ and $\beta = 10^{-6}$. For $\beta = 10^{-7}$, it is below 0.7, indicating a small interval of suitable β-values that achieve both, a disentanglement of the latent variables and a satisfactory reconstruction capability.

Table 4 reports the average training and reconstruction times of the three autoencoder models displayed in Fig. 19. These times again refer to the use of a single NVIDIA GeForce RTX 3090 GPU. For comparison, training a POD on an Intel Xeon Silver 4314 CPU takes around 168 s.

Table 4
Training time and time taken to reconstruct high-dimensional data for the 16384 discrete spatial points from the latent space for the three autoencoder models on the application case using a single NVIDIA GeForce RTX 3090 GPU.

Model	Training time [s]	Decoder inference time [s]
UAE	335	0.00035
OAE	306	0.00036
β-VAE	320	0.00037

The training of the β-VAE [UAE] is around 5% [10%] slower than the training of the OAE. As for the verification case in Section 3.1.2, the time taken to apply the decoder is similar for all models.

3.2.3. Mode analysis

Based on the studies from Section 3.1.3, Figs. 20 and 21 depict the standard deviations of the latent variables (UAE and OAE) and the standard deviations of the mean values $\boldsymbol{\mu}$ (β-VAE) as well as the KL divergence of the β-VAE. The investigated three weight values for ν, λ and β were extracted from the results displayed in Fig. 19.

Considering the UAE on the left, we can observe that many latent variables appear to be active for all three investigated values of ν. For the large value $\nu = 10^{-3}$ displayed in blue, which promises the best disentanglement whilst still delivering a fair amount of reconstruction accuracy, we can structure the latent space into two higher-ranked contributions z_1, z_5 and two second-ranked contributions z_8, z_9. As regards the OAE, the situation is similar. Due to the results shown in Fig. 19, we discard the lower values and focus on the largest meaningful value $\lambda = 10^{-3}$ displayed in blue. The related latent space can also be grouped into a dominating mode z_2 and two second-ranked modes z_8, z_5 followed by z_3. For the β-VAE, Fig. 19 suggests to employ $\beta = 10^{-6}$ as a baseline value, where the four most important latent variables refer to z_7 (top rank) and z_9 slightly ahead of z_6, z_8, z_{10}, cf. red curves in Fig. 20. The KL divergence depicted in Fig. 21 suggests to assign z_6 to a lower priority. When considering the standard deviation and the KL divergence, for $\beta = 10^{-5}$, unlike for $\beta = 10^{-6}$ and $\beta = 10^{-7}$, only two variables appear to be active, which also matches the higher reconstruction error in Fig. 19 compared to the cases $\beta = 10^{-6}$ and $\beta = 10^{-7}$.

For each autoencoder model, we employ the four most active latent variables to assess the associated modes. In contrast to the periodic flow example, we set all other latent variables to zero during the assessment,

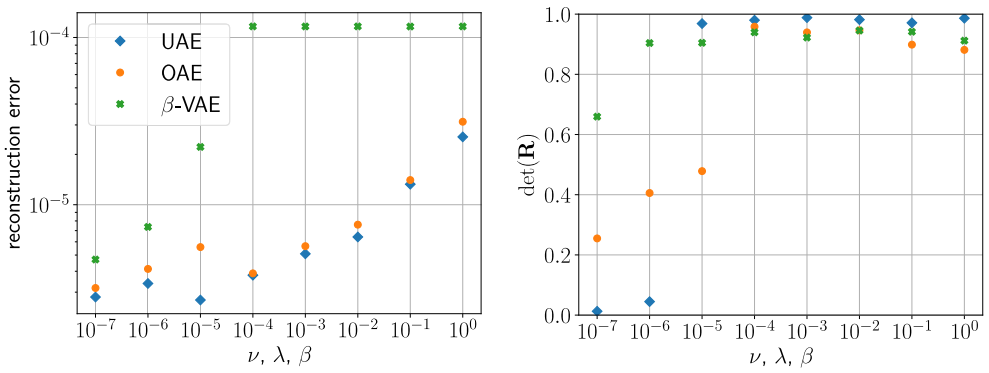

Fig. 19. Reconstruction errors on the validation set and resulting values of $\det(\mathbf{R})$ obtained by the different models for the ditching application using $m = 10$ latent variables.

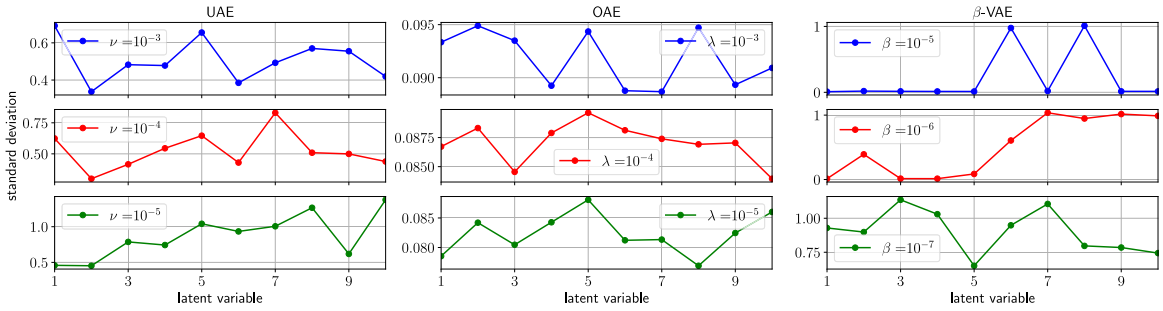

Fig. 20. Standard deviations of the latent variables over the validation set obtained in the application case for a latent space dimension of $m = 10$. The standard deviations for the β-VAE are computed over the means $\boldsymbol{\mu}$ obtained from the encoder. Results for three different weights of the disentanglement contribution to the loss function are shown corresponding to Fig. 19.

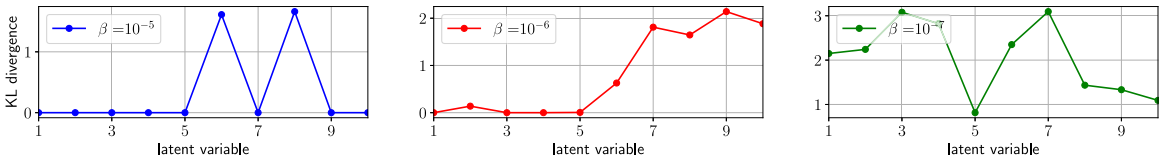

Fig. 21. KL divergence of the latent variables over the validation set for the β-VAE obtained in the application case for a latent space dimension of $m = 10$. Results for three different values of β in the loss function are shown corresponding to Fig. 19.

except the one we are analyzing. The reason is that the ditching data involves more different states and we believe that the choice of a suitable baseline state is not obvious. This also means that the studies do not consider a reference snapshot, since all but one latent variable are inactive.

Fig. 22 illustrates the response of the load to the most active latent variables for the UAE. Considering the most active variable z_1, we can see that negative z_1-values only have a small impact on the output. Positive values seem to be related to the late phase of the impact, cf. steps 13 and 17 in Fig. 18, featuring a typical slender crescent-type pressure load followed by a more pronounced suction load regime. The second most active variable z_5 is contributing to the initial impact phase for positive values, cf. step 1 in Fig. 18, with a confined small area being exposed to high pressure loads. Negative z_5-values resemble the subsequent early phase, cf. steps 5 and 9 in Fig. 18, in which a large area of the fuselage is exposed to smaller pressure loads. Negative values in z_8 are associated to the final phase of the impact, in which loads occur at the rear of the fuselage and begin to vanish, cf. step 21 in Fig. 18. For positive values, z_8 is contributing to the early impact phase, cf. steps 5 and 9 in Fig. 18. Small values of z_9 are related to the late phase and larger values to final phase of the ditching, cf. steps 13, 17 and 21 in Fig. 18.

The outputs generated by an investigation of the four most active latent variables obtained from the OAE are shown in Fig. 23. The OAE results are generally more noisy than the corresponding UAE results. Although it is evident that certain latent variables contribute to the loads occurring in a particular phase of the ditching event, for example, the spatial expansion of the load regime in the early impact phase captured by positive values of z_5, z_8 and a footprint of final phase loads represented by negative values of z_2, in most cases the patterns are not so clearly linked to a loading situation.

Looking at the modes associated with the four most active variables of the β-VAE in Fig. 24, we can see more distinct links with realistic load pattern again. The most active variable z_7 with respect to the standard deviation of the mean $\boldsymbol{\mu}$ is associated to the final phase of the impact for negative values and to the early phase for positive values. Typical load pattern for the late phase resemble the structures reported by z_9, z_8, whereas z_{10} supplements the load widening in circumferential direction. It is interesting to note, that the fifth highest ranked variable z_6 does not yield any meaningful output, while the fourth highest ranked variable z_8 does. The standard deviations of these two latent variables are fairly close, cf. Fig. 20. However, a slightly higher difference can be observed for the KL divergence.

The resulting modes of the ditching loads are in general harder to analyze compared to the periodic flow case, as the temporal dynamics are more complex and over 300 different simulations with different input parameters were used to train the models. However, distinct influences of different active latent variables on different phases of a

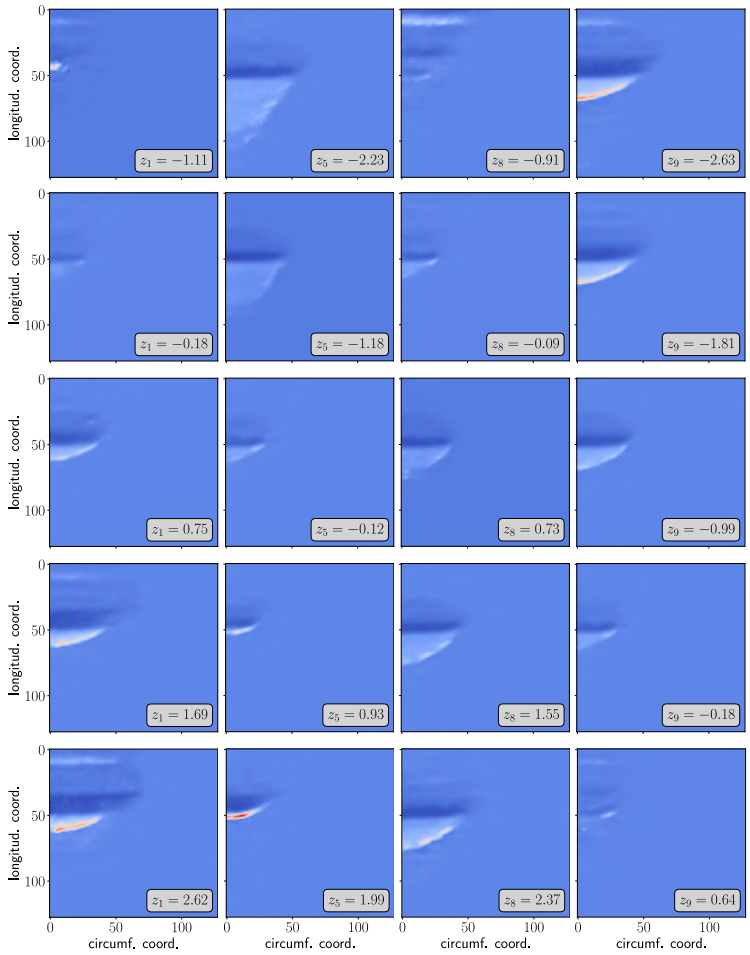

Fig. 22. Impact of four latent variables with the highest standard deviation on the loads obtained from the UAE for the application case using a latent space dimension of $m = 10$ and $\nu = 10^{-3}$. In each column, one latent variable takes equidistant values ranging from the minimum to the maximum obtained on that variable on the validation set while the other latent variables are set to zero. Color code is the same for all examples in this figure, ranging from lowest (dark blue) to highest (dark red) values.

ditching event can be observed when the latent variables are disentangled. One can verify that a latent variable is indeed associated to a specific impact phase by reconstructing a particular snapshot of this phase after setting this variable to 0. For this purpose, we reuse the second test case from [17], which is not included in the training set and simulated with the full-order model using a horizontal velocity of 80.75 m/s, a vertical velocity of 1.91 m/s and a pitch angle of 6°. As an example, we illustrate this for the two most active latent variables of the UAE, i.e., z_1, z_5, cf. Fig. 22. Fig. 25 shows snapshots for two particular time steps of the test case, their UAE-reconstruction using $m = 10$ latent variables and the corresponding UAE-reconstruction obtained from setting either the most active latent variable z_1 or the second most active z_5 to zero. The selected time steps correspond to impact phases associated to the two most active latent variables, i.e., one from the early and one from the late phase, cf. Fig. 22. Comparing the true values in the first column with the reconstructions using all $m = 10$ latent variables shown in the second column displays a satisfactory agreement. If $z_1 = 0$ is set, the reconstruction of the 16th time step depicted in the first row is considered insufficient. However, if $z_5 = 0$ is set, the reconstruction is again very accurate. A similar observation can be made for the 10th time step, which is shown in the second row. Removing the most active latent variable does not have a significant impact on the reconstruction quality. If $z_5 = 0$ is set, the reconstruction deteriorates significantly. It is evident that the corresponding latent variables have a significant influence on the quality of the reconstruction in the respective distinct phases.

Fig. 26 depicts the latent trajectories for the same test case illustrated in Fig. 25. Attention is again restricted to the four dominant modes of the three autoencoder models. Compared to the verification case, the evolution of the latent variables is more complex. In the UAE results shown at the top, all displayed latent variables have values close to zero for most time steps. Often, one or two latent variables temporarily become significantly larger or smaller than the others, e.g., z_1 (blue curve) on time step 16 (late phase, cf. Fig. 25), and z_5 (red curve) on time steps 1 (initial phase), 10–13 (early phase) and 18 (late phase). This observation, in combination with the positive [negative] peak of z_8 [z_9] (green [black] curve) on time step 14 [15], is in good agreement with the mode analysis with respect to Fig. 22. The positive value of z_5 at time step 18 can likely be explained by the fact that the longitudinal coordinates of areas that are exposed to hydrodynamic loads can be similar for the initial impact phase and the late phase. With regards to the OAE, it is more difficult to identify a general behavior for the latent variables and recognize characteristic aspects from the mode analysis in Fig. 23. For the β-VAE, the curves for $\mathbf{z}$ and $\boldsymbol{\mu}$ differ slightly. Similar to the UAE, one can observe a good agreement between the trajectories and the analysis of the modes that are displayed in Fig. 24. For example, the maximum values for z_7 (blue curve) are at time steps 10–13 (early phase) and the minimum values are at time steps 20 and 21 (late phase), along with the positive and negative peaks of z_9 (red curve) at time steps 16 and 19, respectively.

Fig. 27 shows the evolution of the reconstruction and latent space losses over the course of the training of the three autoencoder models.

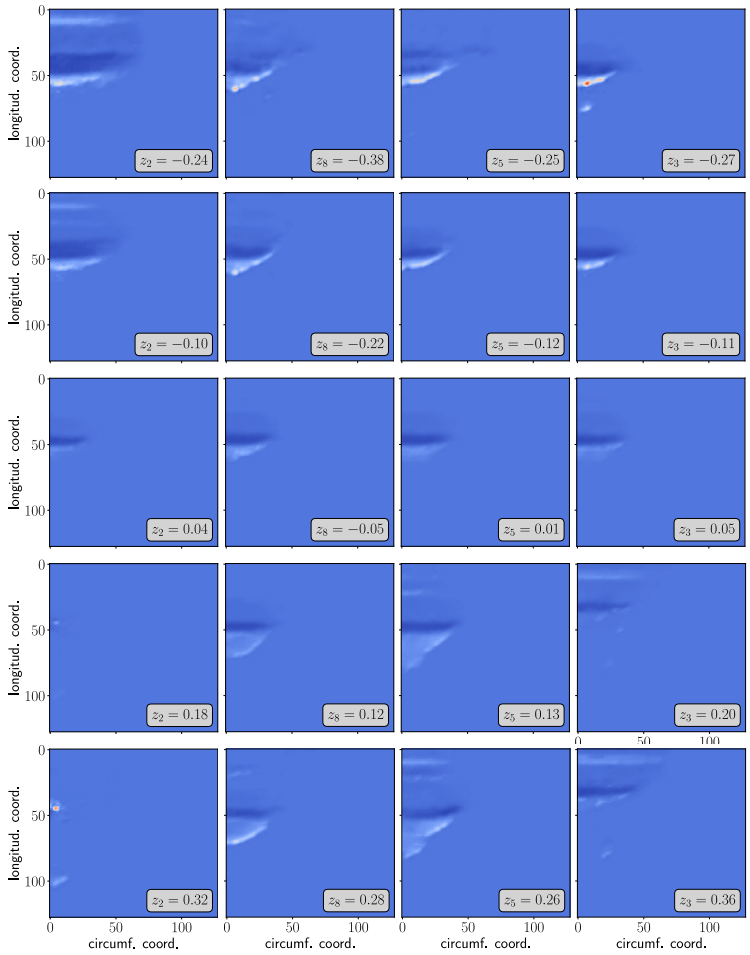

Fig. 23. Impact of the four latent variables with the highest standard deviation on the loads obtained from the OAE for the application case using a latent space dimension of $m = 10$ and $\lambda = 10^{-3}$. In each column, one latent variable takes equidistant values ranging from the minimum to the maximum obtained on that variable on the validation set while the other latent variables are set to zero. Color code is the same for all examples in this figure, ranging from lowest (dark blue) to highest (dark red) values.

The KL divergence of the β-VAE is around two [three] orders of magnitude higher than the latent space loss of the UAE [OAE]. After passing through the initial plateau, the reconstruction loss of the UAE reduces to a similar level like the OAE, which is lower than the level of the β-VAE. Note that accounting for the difference in the ratios of reconstruction loss and latent space loss for the β-VAE and the deterministic models during training through scaling would only result in a shift of the tested values in Fig. 19, and the interval of suitable values would still be smaller for the β-VAE.

3.2.4. Temporal predictions

To assess the performance of the different autoencoder models for time series predictions, we perform temporal predictions in the latent space with an LSTM network. The employed LSTM structure closely follows the structure used in a previous publication of the authors [17], where different combinations of CAE and LSTM networks have been compared for the spatio–temporal prediction of ditching loads. For all autoencoder models, the same LSTM structure is used, that is, two LSTM layers with a hidden state of dimension 100 and an output layer containing 10 neurons to predict the next time step in the latent space. A time delay of three input time steps is used. Moreover, the autoencoders use the same values for ν, λ and β that are used in combination with the generation of modes in Section 3.2.3, i.e., $\nu = \lambda = 10^{-3}$ and $\beta = 10^{-6}$. Mind that the autoencoder and LSTM models are trained separately to first construct the disentangled latent space, and subsequently train the LSTM inside the disentangled latent space. The LSTM models are trained for 500 epochs using Adam with the default learning rate of $1 \cdot 10^{-3}$ and a minibatch size of 512.

The predictive quality of the autoencoder-LSTM combinations are assessed as in [17], i.e., each autoencoder model is trained five times to account for randomness in the training process and for each trained version, one LSTM network is trained. To this end, all 30 cases from the test set are employed. For each trained version, predictions on all test cases are made and normalized root mean squared errors (RMSEs) are computed for each test case. These are then averaged over the five trained versions on a particular test case. Subsequently this quantity is averaged in time for each test case and finally the average over the 30 test cases is computed. The resulting errors are listed in Table 5. For comparison, the table also includes the results for a standard CAE without any disentanglement constraints in the latent space. For the β-VAE, LSTM networks are trained for each the sampled $\mathbf{z}$-vectors and the mean $\boldsymbol{\mu}$-vectors to assess the respective influence on the performance.

The standard CAE and the β-VAE using the means $\boldsymbol{\mu}$ yield the lowest average errors. These are of equal magnitude and approximately correspond to the lowest errors of a CAE-LSTM model in [17]. The UAE-LSTM model yields slightly worse prediction performance. Both the OAE and the β-VAE with the sampled $\mathbf{z}$ vectors yield significantly larger errors, which are almost 10% worse than the worst combination assessed in [17]. The associated computational cost of the LSTM refers to approximately $4 \cdot 10^{-4}$ seconds per predicted time step. Without going into detail, we note that the OAE also performed significantly worse

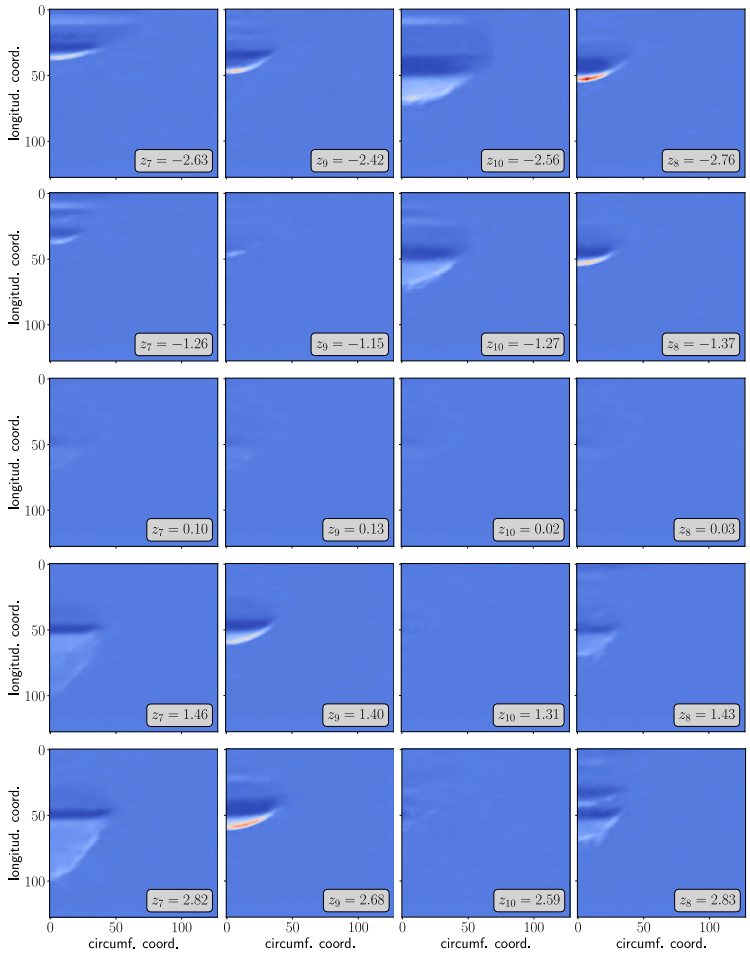

Fig. 24. Impact of the four latent variables with the highest standard deviation on the loads obtained from the β-VAE for the application case using a latent space dimension of $m = 10$ and $\beta = 10^{-6}$. In each column, one latent variable takes equidistant values ranging from the minimum to the maximum obtained on that variable on the validation set while the other latent variables are set to zero. Color code is the same for all examples in this figure, ranging from lowest (dark blue) to highest (dark red) values.

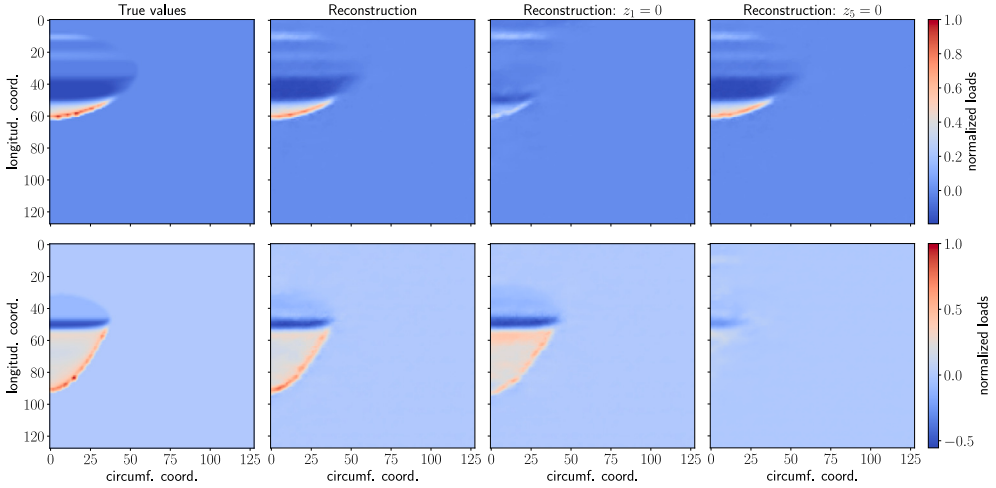

Fig. 25. UAE reconstructions of the 16th (first row) and the 10th (second row) time step of a particular ditching case using a latent space dimension of $m = 10$ and $\nu = 10^{-3}$. The suppressed variables in column three and four, i.e., z_1 and z_5, correspond to the most active and second most active latent variable, respectively.

than the UAE and the β-VAE in initial experiments for the temporal prediction on the periodic flow dataset.

Regarding β-VAE, Kang et al. [19] noted that considering $\det(\mathbf{R})$ using all latent variables is not sensible when the latent space is sparse, and suggested to consider combinations of a smaller number of latent variables. Fig. 28 displays the value for $\det(\mathbf{R})$ using only the $2 \leq i \leq 10$ most active latent variables, averaged over the five trained versions of the autoencoder models used for the temporal predictions. Though

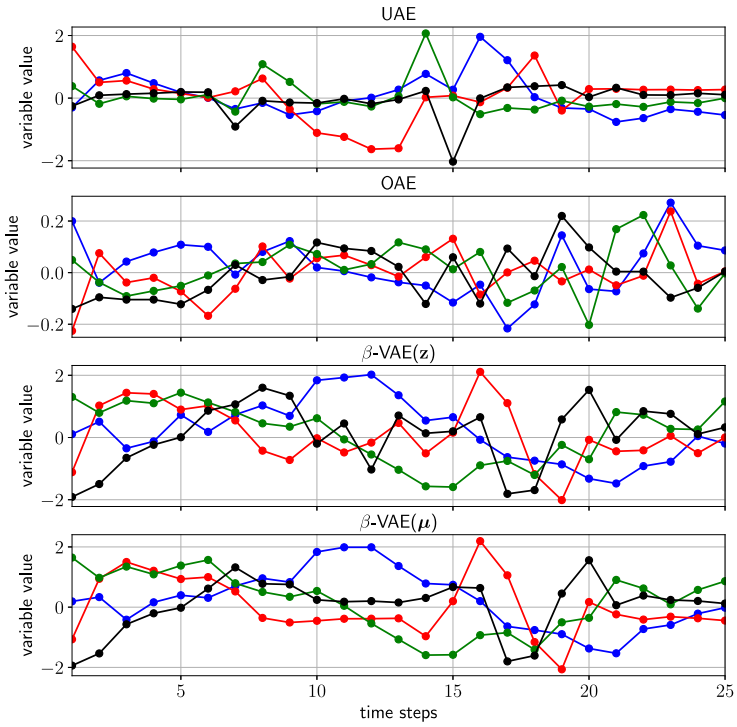

Fig. 26. Latent trajectories of the three autoencoder models for a latent space dimension of $m = 10$ and $\nu = 10^{-3}$, $\lambda = 10^{-3}$, $\beta = 10^{-6}$, respectively, for a particular ditching case. Colors blue, red, green and black refer to the most active components in descending order, cf. Fig. 20.

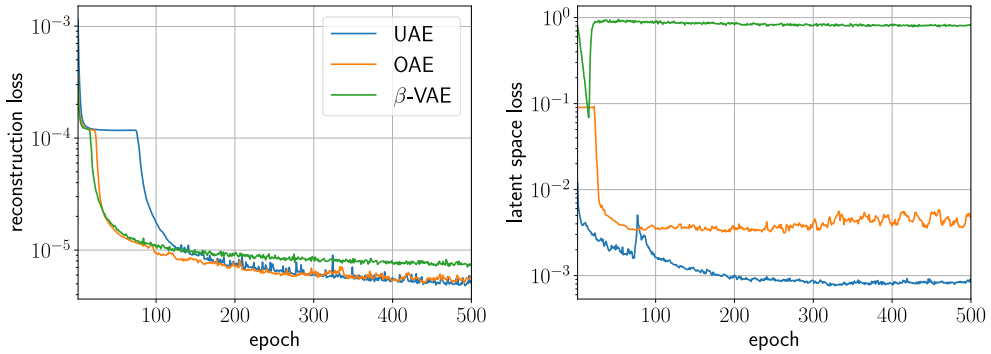

Fig. 27. Loss function evolution on the validation set of the application case with $\nu = 10^{-3}$, $\lambda = 10^{-3}$ and $\beta = 10^{-6}$ for the UAE, OAE, β-VAE, respectively.

Table 5
Average error obtained for the spatio–temporal prediction on all 30 test cases. All models are combined with the same LSTM network.

Model	Average error
CAE	0.020
UAE	0.022
OAE	0.027
β-VAE ($\mathbf{z}$)	0.027
β-VAE ($\boldsymbol{\mu}$)	0.020

the β-VAE returns a slightly lower average error than the UAE when the means $\boldsymbol{\mu}$ are used for the temporal prediction, the UAE latent variables achieve less correlation than the β-VAE means. There is also a noticeable difference between the curves for the sampled $\mathbf{z}$ and the means $\boldsymbol{\mu}$ for the β-VAE, as the value for $\det(\mathbf{R})$ for the means decreases rapidly when more than seven variables are used. Note that multiple latent variables have KL divergences close to zero, cf. Fig. 21, and the means are therefore approximately zero for every example. Similar observations regarding differences between mean and sampled values have previously been made, e.g., [48,49].

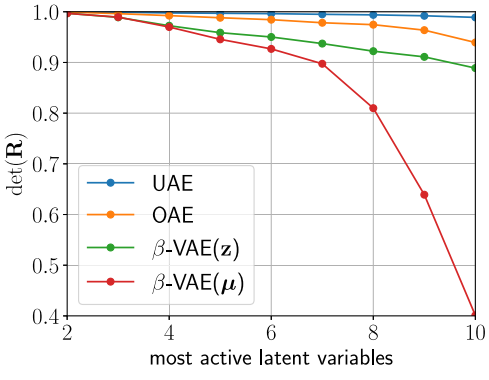

Fig. 28. Average values for $\det(\mathbf{R})$ on the validation set using different amounts of the most active latent variables for the five trained versions of each model used for temporal predictions.

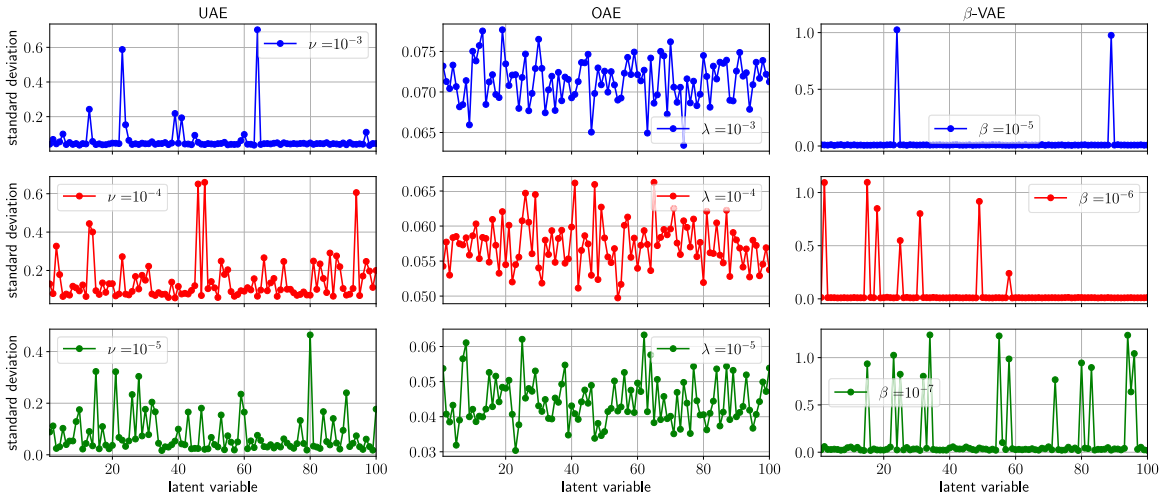

Fig. 29. Standard deviations of the latent variables over the validation set for all three autoencoder models using a latent space dimension of $m = 100$. The standard deviations for the β-VAE are computed over the means $\boldsymbol{\mu}$ obtained from the encoder. Each model is trained with three different weights of the disentanglement contribution to the loss function.

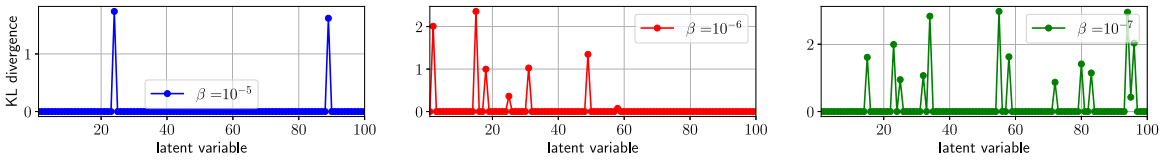

Fig. 30. KL divergence for the latent variables over the validation set of the β-VAE using a latent space dimension of $m = 100$. The model is trained with three different values of β in the loss function.

3.2.5. *Physics awareness for higher latent space dimension*

Similar to Section 3.1.4, we train the models with a latent space dimension of 100 to see, how well the models work at finding a smaller number of active latent variables. Again, we use different weighting values for the contribution of the disentanglement part to the loss function, i.e., $\nu = \lambda = 10^{-3}, 10^{-4}, 10^{-5}$ as well as $\beta = 10^{-5}, 10^{-6}, 10^{-7}$ to analyze the impact of the loss term in the latent space.

Fig. 29 shows the standard deviations of the different latent variables for all three autoencoder models using the different weighting values. We can observe a similar behavior as for the periodic flow example in Fig. 11. Looking at the UAE, we can identify two distinct active latent variables in combination with the most pronounced disentanglement $\nu = 10^{-3}$. Four more latent variables have a noticeably higher standard deviation than the remaining ones, from which four further variables stand out. By reducing the weight to $\nu = 10^{-4}$ and $\nu = 10^{-5}$, the number of active latent variables increases significantly. Observing the behavior of the OAE model shown in the second column, the standard deviations of the latent variables are again generally much closer to each other for $\lambda = 10^{-3}$. For $\lambda = 10^{-4}$ and $\lambda = 10^{-5}$, the differences between the standard deviations become higher, but are never as pronounced as for the UAE. Considering the β-VAE in the right column, we can observe that an increasing amount of latent variables is active for a decreasing value of β. For $\beta = 10^{-7}$, there are more than 10 distinct latent variables with a clearly higher standard deviation, compared to only two active latent variables for $\beta = 10^{-5}$. The corresponding KL divergences are depicted in Fig. 30. As expected, higher KL divergences are returned for lower values of β. The highest values are obtained for the latent variables with the highest standard deviation in Fig. 29.

We compute $\det(\mathbf{R})$ for the twenty most active latent variables to verify their independence. The corresponding results are illustrated in Fig. 31. Analyzing the UAE behavior for $\nu = 10^{-3}$, one finds that the five most active latent variables are all approximately at $\det(\mathbf{R}) \approx 1$, and $\det(\mathbf{R})$ decreases only slightly when using additional active variables. For the 10 most active latent variables, $\det(\mathbf{R})$ is still greater than 0.9. When reducing ν, the value of $\det(\mathbf{R})$ decreases more rapidly, starting with five ($\nu = 10^{-4}$) or even three ($\nu = 10^{-5}$) variables. The OAE gives similar results to the UAE for $\lambda = 10^{-3}$. Using $\lambda = 10^{-4}$ and $\lambda = 10^{-5}$ induces a more rapid decrease in $\det(\mathbf{R})$ when increasing the amount of latent variables. When $\mathbf{z}$ is used to compute $\mathbf{R}$ for the β-VAE, $\beta = 10^{-5}$ and $\beta = 10^{-6}$ return similar values of $\det(\mathbf{R})$ as $\nu = \lambda = 10^{-3}$ for the UAE and OAE, respectively, although the five most active latent variables yield slightly smaller values than the five most active for the UAE. From the seventh ranked variable, $\det(\mathbf{R})$ deviates from the higher values of β for $\beta = 10^{-7}$ and takes values smaller than ≈ 0.83. When $\boldsymbol{\mu}$ is used to compute $\mathbf{R}$, the behavior is again different. For $\beta = 10^{-5}$, the value of $\det(\mathbf{R})$ decreases already when three latent variables are considered. Mind that only two latent variables have KL divergences significantly larger than zero in Fig. 30. Similar behavior can be observed for $\beta = 10^{-6}$ and $\beta = 10^{-7}$, although $\det(\mathbf{R})$ remains over 0.9 for $\beta = 10^{-6}$ for up to seven latent variables.

In Fig. 32, the modes for the four most active latent variables of the UAE using $\nu = 10^{-3}$ are displayed in combination with $m = 100$. The modes are generated exactly as described in Section 3.2.3. For the two most active latent variables z_{64} and z_{23} results agree with the modes associated with z_1 and z_5 in the $m = 10$ configuration, cf. Fig. 22. The main differences are the flipped order of the first column, which is again related to the late phase, as well as differences in the intensities of the loads. The two following latent variables of the $m = 100$ configuration z_{13} and z_{39}, however, have a smaller impact on the output. This could also be expected from the different standard deviations depicted in Fig. 29. Without going into detail, we report that the β-VAE also returned reasonable leading modes for $\beta = 10^{-7}$ and $m = 100$.

4. Conclusion

The paper compares the performance of two deterministic autoencoder models and the β-variational autoencoder for the dimension

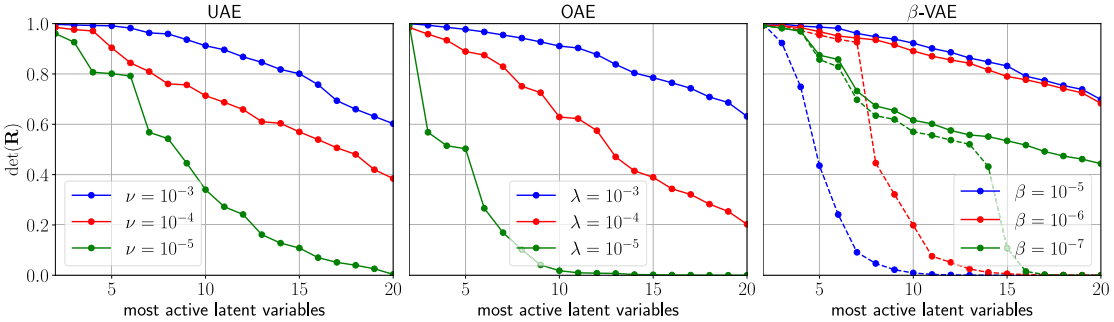

Fig. 31. Values for $\det(\mathbf{R})$ on the validation set using different amounts of the most active latent variables. For the β-VAE, $\mathbf{R}$ has been computed either on the sampled $\mathbf{z}$ (solid lines) or the means $\boldsymbol{\mu}$ (dashed line).

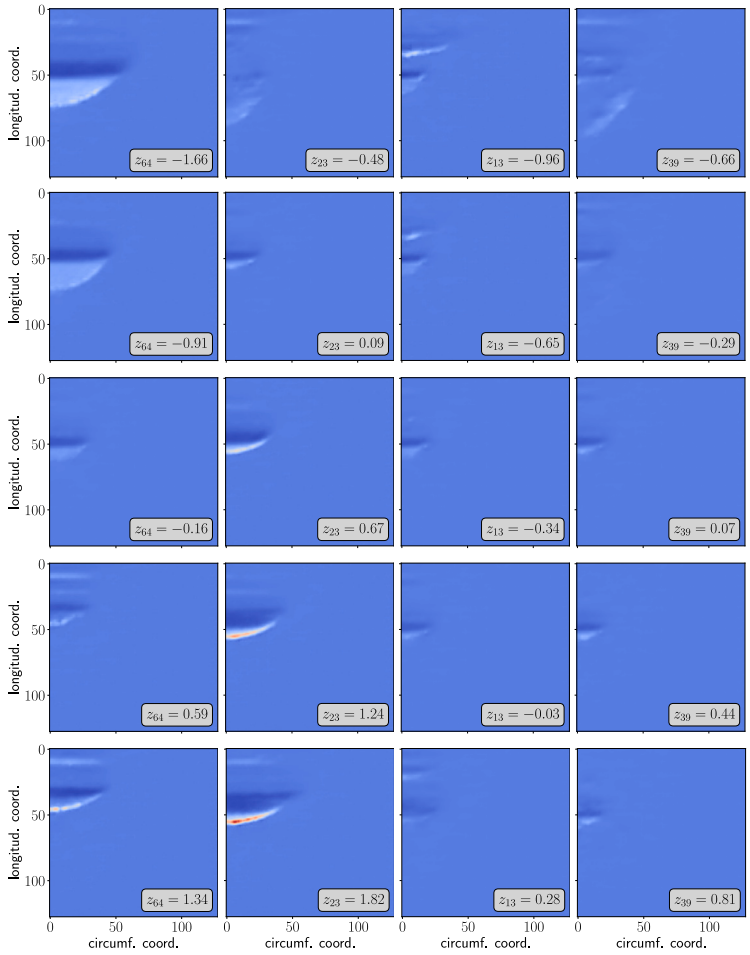

Fig. 32. Impact of four latent variables with the highest standard deviation on the loads obtained from the UAE trained with a latent space dimension of $m = 100$. In each column, one latent variable takes equidistant values ranging from the minimum to the maximum obtained on that variable on the validation set while the other latent variables are set to zero. Color code is the same for all examples in this figure, ranging from lowest (dark blue) to highest (dark red) values.

reduction of two-mode periodic benchmark flow data and multi-mode aircraft ditching load data. Emphasis is placed on the disentanglement of the latent variables and the analysis of the resulting modes. Considering the autoencoder reconstruction accuracy and the level of disentanglement in the latent space, the uncorrelated autoencoder (UAE) outperformed the β-VAE, while being deterministic and easier to train with respect to the choice of the hyperparameter that balances the reconstruction loss and latent space loss in both test cases. For temporal predictions, however, the β-VAE returned slightly lower errors than the UAE, but the UAE latent variables were much less correlated. If only reconstruction accuracy and disentanglement of latent variables are desired in a surrogate model, but not specifically a normally distributed latent space, the UAE may more easily provide satisfactory results. Similarly to β-VAE, the UAE can identify a small and limited number of truly active latent variables when the model is trained with a larger latent space dimension than required. Such active latent variables can be easily identified by their standard deviation. The second deterministic autoencoder model tested, the orthogonal autoencoder (OAE), was also able to return small correlation coefficients between the latent variables. However, it was not able to identify the active latent variables in this study when trained with a higher latent space dimension, and performed worse than the UAE and the β-VAE on the

temporal predictions. The analysis of the ditching case provided a better understanding of how the different latent variables contribute to the reconstruction. It was observed that different latent variables concentrate at different temporal phases of the impact. This can be of importance in future work, when deformation-induced load changes are to be included in the model. Future efforts will also focus on the application of disentangled latent spaces to the analysis of more complex 3D flow fields, e.g., [50]. This will likely require larger latent spaces. Commonly used training strategies for β-VAE, such as latent space loss warm-up phases, should be revisited in combination with the UAE since they could also help to train the UAE for significantly more complex problems.

CRediT authorship contribution statement

Henning Schwarz: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Pyei Phyo Lin:** Writing – review & editing, Methodology, Conceptualization. **Jens-Peter M. Zemke:** Writing – review & editing, Writing – original draft, Methodology, Conceptualization. **Thomas Rung:** Writing – review & editing, Writing – original draft, Supervision, Resources, Project administration, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

H.S., P.P.L., and T.R. acknowledge support by the German Federal Ministry for Economic Affairs and Climate Action under aegis of the “Luftfahrtforschungsprogramm LuFo VI” project HYMNE (grant 20E2218A) and by the Clean Aviation project FASTER-H2, funded by the European Union under Grant Agreement No. 101101978. The views and opinions expressed are solely those of the author(s) and do not necessarily reflect those of the European Union or Clean Aviation. Neither the European Union nor Clean Aviation can be held responsible for them. This paper is a contribution to the research training group RTG 2583 on “Modeling, Simulation and Optimization of Fluid Dynamic Applications” funded by the Deutsche Forschungsgemeinschaft (DFG). The authors acknowledge the support of TU Hamburg’s Machine Learning in Engineering (MLE) initiative.

Data availability

The periodic flow dataset was created by Solera-Rico et al. [16] and is publicly available under [40].

References

[1] Lumley JL. The structure of inhomogeneous turbulent flows. Atmospheric Turbul Radio Wave Propag 1967;166–78.
[2] Lusch B, Kutz JN, Brunton SL. Deep learning for universal linear embeddings of nonlinear dynamics. Nat Commun 2018;9(4950).
[3] Swischuk R, Mainini L, Peherstorfer B, Willcox K. Projection-based model reduction: Formulations for physics-based machine learning. Comput & Fluids 2019;179:704–17. http://dx.doi.org/10.1016/j.compfluid.2018.07.021.
[4] Eivazi H, Veisi H, Naderi MH, Esfahanian V. Deep neural networks for nonlinear model order reduction of unsteady flows. Phys Fluids 2020;32(10):105104. http://dx.doi.org/10.1063/5.0020526.
[5] Agostini L. Exploration and prediction of fluid dynamical systems using auto-encoder technology. Phys Fluids 2020;32(6):067103. http://dx.doi.org/10.1063/5.0012906.
[6] Ganti H, Khare P. Data-driven surrogate modeling of multiphase flows using machine learning techniques. Comput & Fluids 2020;211:104626. http://dx.doi.org/10.1016/j.compfluid.2020.104626, URL https://www.sciencedirect.com/science/article/pii/S0045793020301985.
[7] Wu P, Gong S, Pan K, Qiu F, Feng W, Pain C. Reduced order model using convolutional auto-encoder with self-attention. Phys Fluids 2021;33(7):077107. http://dx.doi.org/10.1063/5.0051155.
[8] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput 1997;9.8:1735–80. http://dx.doi.org/10.1162/neco.1997.9.8.1735.
[9] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I. Attention is all you need. In: Advances in neural information processing systems. 2017, URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.
[10] Brunton SL, Proctor JL, Kutz JN. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. Proc Natl Acad Sci 2016;113(15):3932–7.
[11] Koopman BO. Hamiltonian systems and transformation in Hilbert space. Proc Natl Acad Sci USA 1931;17:315–8.
[12] Gupta R, Jaiman R. A hybrid partitioned deep learning methodology for moving interface and fluid–structure interaction. Comput & Fluids 2022;233:105239. http://dx.doi.org/10.1016/j.compfluid.2021.105239, URL https://www.sciencedirect.com/science/article/pii/S0045793021003479.
[13] Hemmasian A, Barati Farimani A. Reduced-order modeling of fluid flows with transformers. Phys Fluids 2023;35(5):057126. http://dx.doi.org/10.1063/5.0151515, arXiv:https://pubs.aip.org/aip/pof/article-pdf/doi/10.1063/5.0151515/17720097/057126_1_5.0151515.pdf.
[14] Ando K, Onishi K, Bale R, Kuroda A, Tsubokura M. Nonlinear reduced-order modeling for three-dimensional turbulent flow by large-scale machine learning. Comput & Fluids 2023;266:106047. http://dx.doi.org/10.1016/j.compfluid.2023.106047, URL https://www.sciencedirect.com/science/article/pii/S0045793023002724.
[15] Zhang B. Airfoil-based convolutional autoencoder and long short-term memory neural network for predicting coherent structures evolution around an airfoil. Comput & Fluids 2023;258:105883. http://dx.doi.org/10.1016/j.compfluid.2023.105883, URL https://www.sciencedirect.com/science/article/pii/S0045793023001081.
[16] Solera-Rico A, Vila CS, Gómez-López M, Wang Y, Almashjary A, Dawson S, Vinuesa R. β-variational autoencoders and transformers for reduced-order modelling of fluid flows. Nat Commun 2024;15. http://dx.doi.org/10.1038/s41467-024-45578-4.
[17] Schwarz H, Überrück M, Zemke J-PM, Rung T. Machine learning based prediction of ditching loads. AIAA J 2025;63(5):1835–54. http://dx.doi.org/10.2514/1.J064086.
[18] Eivazi H, Clainche SL, Hoyas S, Vinuesa R. Towards extraction of orthogonal and parsimonious non-linear modes from turbulent flows. Expert Syst Appl 2022;202:117038. http://dx.doi.org/10.1016/j.eswa.2022.117038, URL https://www.sciencedirect.com/science/article/pii/S0957417422004535.
[19] Kang Y-E, Yang S, Yee K. Physics-aware reduced-order modeling of transonic flow via β-variational autoencoder. Phys Fluids 2022;34(7):076103. http://dx.doi.org/10.1063/5.0097740.
[20] Higgins I, Matthey L, Pal A, Burgess CP, Glorot X, Botvinick MM, et al. beta-VAE: Learning basic visual concepts with a constrained variational framework. In: International conference on learning representations. 2016.
[21] Kingma DP, Welling M. Auto-encoding variational Bayes. 2014, CoRR arXiv:1312.6114.
[22] Wang W, Yang D, Chen F, Pang Y, Huang S, Ge Y. Clustering with orthogonal AutoEncoder. IEEE Access 2019;7:62421–32. http://dx.doi.org/10.1109/ACCESS.2019.2916030.
[23] Cacciarelli D, Kulahci M. A novel fault detection and diagnosis approach based on orthogonal autoencoders. Comput Chem Eng 2022;163:107853. http://dx.doi.org/10.1016/j.compchemeng.2022.107853, URL https://www.sciencedirect.com/science/article/pii/S0098135422001910.
[24] Chen RTQ, Li X, Grosse RB, Duvenaud DK. Isolating sources of disentanglement in variational autoencoders. In: Bengio S, Wallach H, Larochelle H, Grauman K, Cesa-Bianchi N, Garnett R, editors. Advances in neural information processing systems, vol. 31. Curran Associates, Inc.; 2018, URL https://proceedings.neurips.cc/paper_files/paper/2018/file/1ee3dfcd8a0645a25a35977997223d22-Paper.pdf.
[25] Carbonneau M-A, Zaïdi J, Boilard J, Gagnon G. Measuring disentanglement: A review of metrics. IEEE Trans Neural Netw Learn Syst 2024;35(7):8747–61. http://dx.doi.org/10.1109/TNNLS.2022.3218982.
[26] Jacobsen C, Duraisamy K. Disentangling generative factors of physical fields using variational autoencoders. Front Phys 2022;10. http://dx.doi.org/10.3389/fphy.2022.890910, URL https://www.frontiersin.org/journals/physics/articles/10.3389/fphy.2022.890910.
[27] Wang Y, Solera-Rico A, Sanmiguel Vila C, Vinuesa R. Towards optimal β-variational autoencoders combined with transformers for reduced-order modelling of turbulent flows. Int J Heat Fluid Flow 2024;105:109254. http://dx.doi.org/10.1016/j.ijheatfluidflow.2023.109254, URL https://www.sciencedirect.com/science/article/pii/S0142727X23001534.

[28] Pache R, Rung T. Data-driven surrogate modeling of aerodynamic forces on the superstructure of container vessels. Eng Appl Comput Fluid Mech 2022;16:746–63. http://dx.doi.org/10.1080/19942060.2022.2044383.

[29] Lazzara M, Chevalier M, Colombo M, Garay Garcia J, Lapeyre C, Teste O. Surrogate modelling for an aircraft dynamic landing loads simulation using an LSTM AutoEncoder-based dimensionality reduction approach. Aerosp Sci Technol 2022;126:107629. http://dx.doi.org/10.1016/j.ast.2022.107629, URL https://www.sciencedirect.com/science/article/pii/S1270963822003030.

[30] Dias Ribeiro M, Stradtner M, Bekemeyer P. Unsteady reduced order model with neural networks and flight-physics-based regularization for aerodynamic applications. Comput & Fluids 2023;264:105949. http://dx.doi.org/10.1016/j.compfluid.2023.105949, URL https://www.sciencedirect.com/science/article/pii/S0045793023001743.

[31] Bowman S, Vilnis L, Vinyals O, Dai A, Jozefowicz R, Bengio S. Generating sentences from a continuous space. In: Proceedings of the 20th SIGNLL conference on computational natural language learning. 2016, p. 10–21.

[32] Burda Y, Grosse R, Salakhutdinov R. Importance weighted autoencoders. 2016, arXiv:1509.00519. URL https://arxiv.org/abs/1509.00519.

[33] Sønderby CK, Raiko T, Maaløe L, Sønderby SK, Winther O. Ladder variational autoencoders. In: Lee D, Sugiyama M, Luxburg U, Guyon I, Garnett R, editors. Advances in neural information processing systems, vol. 29. Curran Associates, Inc.; 2016, URL https://proceedings.neurips.cc/paper_files/paper/2016/file/6ae07dcb33ec3b7c814df797cbda0f87-Paper.pdf.

[34] Kim M, Choi H-C. Uncorrelated feature encoding for faster image style transfer. Neural Netw 2021;140:148–57. http://dx.doi.org/10.1016/j.neunet.2021.03.007, URL https://www.sciencedirect.com/science/article/pii/S0893608021000873.

[35] Savargaonkar M, Oyewole I, Chehade A, Hussein AA. Uncorrelated sparse autoencoder with long short-term memory for state-of-charge estimations in lithium-ion battery cells. IEEE Trans Autom Sci Eng 2024;21(1):15–26. http://dx.doi.org/10.1109/TASE.2022.3222759.

[36] Lucas J, Tucker G, Grosse R, Norouzi M. Understanding posterior collapse in generative latent variable models. In: 7th international conference on learning representations. ICLR, 2019.

[37] van den Oord A, Vinyals O, Kavukcuoglu K. Neural discrete representation learning. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, Garnett R, editors. Advances in neural information processing systems, vol. 30. Curran Associates, Inc.; 2017, URL https://proceedings.neurips.cc/paper_files/paper/2017/file/7a98af17e63a0ac09ce2e96d03992fbc-Paper.pdf.

[38] Davidson TR, Falorsi L, Cao ND, Kipf T, Tomczak JM. Hyperspherical variational auto-encoders. 2022, arXiv:1804.00891. URL https://arxiv.org/abs/1804.00891.

[39] Burgess CP, Higgins I, Pal A, Matthey L, Watters N, Desjardins G, et al. Understanding disentangling in β-VAE. 2018, arXiv:1804.03599. URL https://arxiv.org/abs/1804.03599.

[40] Solera-Rico A, Sanmiguel Vila C, Gómez MÁ, Wang Y, Almashjary A, Dawson S, et al. Dataset of "beta-variational autoencoders and transformers for reduced-order modelling of fluid flows". 2024, http://dx.doi.org/10.5281/zenodo.10501216.

[41] Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, et al. PyTorch: An imperative style, high-performance deep learning library. In: et al. HW, editor. Advances in neural information processing systems 32. Curran Associates, Inc.; 2019, p. 8024–35, URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[42] Asztalos KJ, Almashjary A, Dawson STM. Galerkin spectral estimation of vortex-dominated wake flows. Theor Comput Fluid Dyn 2024;38:801–23. http://dx.doi.org/10.1007/s00162-023-00670-1.

[43] Clevert D-A, Unterthiner T, Hochreiter S. Fast and accurate deep network learning by exponential linear units (ELUs). 2016, arXiv:1511.07289. URL https://arxiv.org/abs/1511.07289.

[44] Kingma DP, Ba J. Adam: A method for stochastic optimization. In: Proceedings of the 3rd international conference on learning representations. ICLR, 2015.

[45] Smith LN, Topin N. Super-convergence: Very fast training of neural networks using large learning rates. 2018, arXiv:1708.07120.

[46] Bensch L, Shigunov V, Söding H. Computational method to simulate planned ditching of a transport airplane. In: Bathe K, editor. Computational fluid and solid mechanics 2003. Oxford: Elsevier Science Ltd; 2003, p. 1251–4. http://dx.doi.org/10.1016/B978-008044046-0.50307-9, URL https://www.sciencedirect.com/science/article/pii/B9780080440460503079.

[47] Maas AL, Hannun AY, Ng AY. Rectifier nonlinearities improve neural network acoustic models. In: Proceedings of the 30th international conference on machine learning, vol. 28, 2013.

[48] Locatello F, Bauer S, Lucic M, Raetsch G, Gelly S, Schölkopf B, et al. Challenging common assumptions in the unsupervised learning of disentangled representations. In: Chaudhuri K, Salakhutdinov R, editors. Proceedings of the 36th international conference on machine learning. Proceedings of machine learning research, vol. 97, PMLR; 2019, p. 4114–24, URL https://proceedings.mlr.press/v97/locatello19a.html.

[49] Bonheme L, Grzes M. Be more active! understanding the differences between mean and sampled representations of variational autoencoders. J Mach Learn Res 2023;24(324):1–30, URL http://jmlr.org/papers/v24/21-1145.html.

[50] Ashton N, Maddix DC, Gundry S, Shabestari PM. AhmedML: High-fidelity computational fluid dynamics dataset for incompressible, low-speed bluff body aerodynamics. 2024, arXiv:2407.20801. URL https://arxiv.org/abs/2407.20801.