Physics of Fluids

Semi-conditional variational auto-encoder for flow reconstruction and uncertainty quantification from limited observations

Cite as: Phys. Fluids **33**, 017119 (2021); https://doi.org/10.1063/5.0025779
Submitted: 19 August 2020 • Accepted: 19 November 2020 • Published Online: 28 January 2021

Kristian Gundersen, Anna Oleynik, Nello Blaser, et al.

View Online Export Citation CrossMark

ARTICLES YOU MAY BE INTERESTED IN

Physics guided machine learning using simplified theories
Physics of Fluids **33**, 011701 (2021); https://doi.org/10.1063/5.0038929

Data-driven recovery of hidden physics in reduced order modeling of fluid flows
Physics of Fluids **32**, 036602 (2020); https://doi.org/10.1063/5.0002051

Exploration and prediction of fluid dynamical systems using auto-encoder technology
Physics of Fluids **32**, 067103 (2020); https://doi.org/10.1063/5.0012906

Semi-conditional variational auto-encoder for flow reconstruction and uncertainty quantification from limited observations

Cite as: Phys. Fluids 33, 017119 (2021); doi: 10.1063/5.0025779
Submitted: 19 August 2020 • Accepted: 19 November 2020 •
Published Online: 28 January 2021

Kristian Gundersen,[1,a)] Anna Oleynik,[1,b)] Nello Blaser,[2,c)] and Guttorm Alendal[1,d)]

AFFILIATIONS

[1] Department of Mathematics, University of Bergen, 5020 Bergen, Norway
[2] Department of Informatics, University of Bergen, 5020 Bergen, Norway

[a)] **Author to whom correspondence should be addressed:** Kristian.Gundersen@uib.no
[b)] Anna.Oleynik@uib.no
[c)] Nello.Blaser@uib.no
[d)] Guttorm.Alendal@uib.no

ABSTRACT

We present a new data-driven model to reconstruct nonlinear flow from spatially sparse observations. The proposed model is a version of a Conditional Variational Auto-Encoder (CVAE), which allows for probabilistic reconstruction and thus uncertainty quantification of the prediction. We show that in our model, conditioning on measurements from the complete flow data leads to a CVAE where only the decoder depends on the measurements. For this reason, we call the model semi-conditional variational autoencoder. The method, reconstructions, and associated uncertainty estimates are illustrated on the velocity data from simulations of 2D flow around a cylinder and bottom currents from a simulation of the southern North Sea by the Bergen Ocean Model. The reconstruction errors are compared to those of the Gappy proper orthogonal decomposition method.

I. INTRODUCTION

Reconstruction of non-linear dynamic processes from sparse observations traditionally requires knowledge of the processes and the governing equations to be able to generalize to a wider area around, in-between, and beyond the measurements. Alternatively, it is possible to learn the underlying processes or equations based on the data itself, the so-called data-driven methods. In geophysics and environmental monitoring, measurements are often only available at sparse locations. For instance, within the field of meteorology, atmospheric pressures, temperatures, and wind are only measured at a limited number of stations. Producing accurate and general weather predictions requires methods that both forecast for the future and also reconstruct where no data are available. Within oceanography, one faces the same problem that *in situ* information about the ocean dynamics is only available at sparse locations such as buoys or sub-sea sensors.

Both the weather and ocean currents can be approximated with models that are governed by physical laws, e.g., the Navier–Stokes equation. However, it is of crucial importance to incorporate observations in order to obtain accurate reliable reconstructions and forecasts.

Reconstruction and inference based on sparse observations are important in numerous applications.[1–6] Bolton and Zanna[3] used Convolutional Neural Networks (CNNs) to hindcast ocean models, and Yeo[7] reconstructed time series from nonlinear dynamics based on sparse observations. Oikonomo *et al.*[8] proposed a method for filling data gaps in groundwater level observations, and Kong *et al.*[2] used reconstruction techniques to model the characteristics of cartridge valves.

The above mentioned applications and methods are just some of the many examples of reconstruction of a dynamic process based on limited information. Here, we focus on the reconstruction of flow. This problem can be formulated as follows: Let $\boldsymbol{w} \in \mathbb{R}^d$,

$d \in \mathbb{N}$, represent a state vector of the flow containing, for example, velocity, pressure, and temperature. Here, we will focus on incompressible unsteady fluid flows in 2D, so $\boldsymbol{w} = (u, v) \in \mathbb{R}^2$, where u and v are the two velocity components. The velocity vector, $\boldsymbol{w}$, is typically obtained from computational fluid dynamic simulations on a meshed spatial domain, $\mathcal{P}$, at discrete times, $\mathcal{T} = \{t_1, \dots, t_K\}$.

Let $\mathcal{P} = \{p_1, \dots, p_N\}$ consist of N grid points p_n, $n = 1, \dots, N$. Then, the state of the flow, $\boldsymbol{w}$, evaluated on $\mathcal{P}$ at a time $t_i \in \mathcal{T}$, can be represented as a vector $\boldsymbol{x}^{(i)} \in \mathbb{R}^{2N}$,

$$\boldsymbol{x}^{(i)} = \left(u(p_1, t_i), \dots, u(p_N, t_i), v(p_1, t_i), \dots, v(p_N, t_i)\right)^T. \quad (1)$$

The collection of $\boldsymbol{x}^{(i)}$, $i = 1, \dots, K$, constitutes the dataset $\boldsymbol{X}$. In order to account for incompressibility, we introduce a discrete divergence operator L_{div}, which is given by a $N \times 2N$ matrix associated with a finite difference scheme, and

$$(L_{div}\, \boldsymbol{x})_k \approx (\nabla \cdot w)(p_k) = 0. \quad (2)$$

Furthermore, we assume that the state is measured only at specific points in $\mathcal{P}$, that is, at $\mathcal{Q} = \{q_1, \dots, q_M\} \subset \mathcal{P}$ with typically $M \ll N$. Hence, there is $\boldsymbol{M} = \{\boldsymbol{m}^{(i)} \in \mathbb{R}^{2M} : \boldsymbol{m}^{(i)} = C\boldsymbol{x}^{(i)}, \forall \boldsymbol{x}^{(i)} \in \boldsymbol{X}\}$, where $\boldsymbol{C} \in \mathbb{R}^{2M \times 2N}$ represents a sampling matrix. More specifically, $\boldsymbol{C}$ is a two block matrix,

$$\boldsymbol{C} = \begin{pmatrix} \boldsymbol{C}_{1/2} & O \\ O & \boldsymbol{C}_{1/2} \end{pmatrix},$$

$$(\boldsymbol{C}_{1/2})_{ij} = \begin{cases} 1, & \text{if } q_i = p_j \\ 0, & \text{otherwise} \end{cases}, \quad i = 1, \dots, N \quad j = 1, \dots, M,$$

and $\boldsymbol{O} \in \mathbb{R}^{M \times N}$ is the zero matrix. The problem of reconstructing fluid flow $\boldsymbol{x}^{(i)} \in \boldsymbol{X}$ from $\boldsymbol{m}^{(i)} \in \boldsymbol{M}$ is presented as a schematic plot in Fig. 1.

There have been a wide range of methods for solving the problem, e.g., Refs. 9–14. In particular, the use of proper orthogonal decomposition (POD)[9] techniques has been popular. It is a traditional dimensional reduction technique where, based on a dataset, a number of basis functions are constructed. The key idea is that linear combinations of these basis functions can reconstruct the original data within some error margin, efficiently reducing the dimension of the problem. In a modified version of the POD, the Gappy POD (GPOD),[10] the aim is to fill the gap in-between sparse measurements. Given a POD basis, one can minimize the L_2-error with the measurements and find a linear combination of the POD-basis that complements the measurements. If the basis is not known, an iterative scheme can be formulated to optimize the basis based on the measurements. The original application of GPOD[10] was related to reconstruction of human faces, and it has later been applied to fluid flow reconstruction.[4,15–17] This motivates us to use this method for comparison in our study.

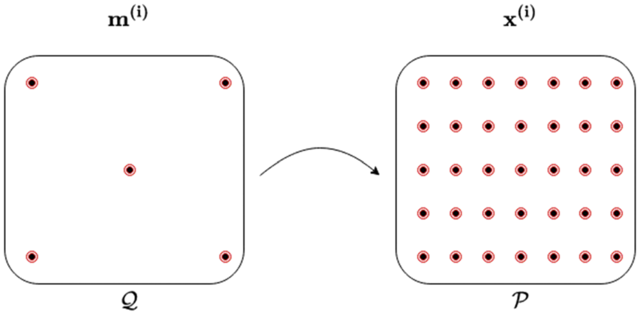

FIG. 1. Sketch of reconstruction of $\boldsymbol{x}^{(i)}$ from $\boldsymbol{m}^{(i)}$. The dots on the right-hand side represent the grid $\mathcal{P}$, and those on the left-hand side represent the measurement locations $\mathcal{Q}$.

A similar approach is the Compressed Sensing (CS) technique.[11] As for the GPOD method, the aim is to solve a linear system; in the CS case, it will be a under-determined linear system. Hence, there is a need for additional information about the system to be able to solve it; typically, this can be a condition/constraint related to the smoothness of the solution. The core difference between CS and GPOD is, however, the sparsity constraint. That is, instead of minimizing the L_2-norm, we minimize the L_1-norm. Minimizing the L_1-norm favors sparse solutions, i.e., solutions with a small number of nonzero coefficients.

Another reconstruction approach is Dynamical Mode Decomposition (DMD).[12] Instead of using principal components in the spatial domain, DMD seeks to find modes or representations that are associated with a specific frequency in the data, i.e., modes in the temporal domain. Again, the goal is to find a solution to an undetermined linear system and reconstruct based on the measurements by minimizing the deviance from the observed values.

During the past decade, data-driven methods have become popular, partly because of the growth and availability of data, but also driven by new technology and improved hardware. Modeling non-linear relationships with linear approximations is one of the fundamental limitation of the DMD, CS, and GPOD methods. There have been efforts that involve neural networks and other machine learning methods to solve the flow reconstruction and decomposition problem. For example, Pawar and San[18] used data assimilation to correct a physics-based model coupled with a neural network as a surrogate to resolved flow dynamics in multiscale systems. Other examples include the work of Erichson *et al.*[19] that used shallow neural networks to learn the input-to-output mapping between sensor measurements and flow fields for a deterministic reconstruction of the data and that of Pérez *et al.*[20] that reconstructed three-dimensional flow fields from two-dimensional data through higher order dynamic mode decomposition,[21] while Gao *et al.*[22] used an attentional generative adversarial neural network to reconstruct and interpolate air pollution data from Beijing. Probabilistic reconstruction of flow fields is not that widespread; however, Maulik *et al.*[23] recently developed a probabilistic neural network for fluid flow surrogate modeling and data recovery through Mixture Density Networks (MDN). MDNs can account for complex multi-modal conditional distributions but are, in general, difficult to optimize. In addition, MDNs have tendency to overfit the training data and are prone to mode collapse.[24] These are examples of recent initiatives that seek to either reconstruct or decompose flow fields based on the input from limited information with data-driven methods.

Another promising and interesting method where the neural networks are informed with a physical law, the so-called Physic-Informed Neural Networks (PINNs),[14] has been developed during the past few years. In PINNs, the reconstruction is typically informed by a Partial Differential Equation (PDE) (e.g., Navier Stokes equation for fluid flow), and thus, the neural network can learn to fill the gap between measurements that are in compliance with this equation. This is what Raissi *et al.*[25] have shown for the benchmark examples such as flow around a 2D and 3D cylinder.

Although PINNs are showing encouraging results, we have yet to see applications to complex systems such as atmospheric or oceanographic systems, where other aspects have to be accounted for in large scale oceanic circulation models that are driven by forcing, such as tides, bathymetry, and river-influx. That being said, these problems may be resolved through PINNs in the future.

Popular non-linear data-driven approaches for reconstruction of fluid flow are different variations of autoencoders.[13,26,27] An autoencoder[28] is a special configuration of an artificial neural network that first encodes the data by gradually decreasing the size of the hidden layers. Through this process, the data are represented in a lower dimensional space. A second neural network then takes the output of the encoder as the input and decodes the representation back to its original shape. These two neural networks together constitute an autoencoder.

Principal Component Analysis (PCA)[29] also represent the data in a different and more compact space. However, PCA reduces the dimension of the data by finding orthogonal basis functions or principal components through singular value decomposition. In fact, it has been showed with a linear activation function, PCA and autoencoders produce the same basis functions.[30] The probabilistic version of the autoencoder is called Variational Auto-Encoder (VAE).[31] Conditional Variational Auto-Encoders (CVAEs)[32] are conditional probabilistic autoencoders, that is, the model is dependent on some additional information such that it is possible to create representations that are dependent on this information.

Here, we address the mentioned problem from a probabilistic point of view. Let $\boldsymbol{x}: \mathcal{P} \to \mathbb{R}^{2N}$ and $\boldsymbol{m}: \mathcal{Q} \to \mathbb{R}^{2M}$ be two multivariate random variables associated with the flow on $\mathcal{P}$ and $\mathcal{Q}$, respectively. Then, the datasets $\boldsymbol{X}$ and $\boldsymbol{M}$ consist of the realizations of $\boldsymbol{x}$ and $\boldsymbol{m}$, respectively. Using $\boldsymbol{X}$ and $\boldsymbol{M}$, we intend to approximate the probability distribution $p(\boldsymbol{x}|\boldsymbol{m})$. This would allow us not only to predict $\boldsymbol{x}^{(i)}$ given $\boldsymbol{m}^{(i)}$ but also to estimate an associated uncertainty. We use a variational autoencoder to approximate $p(\boldsymbol{x}|\boldsymbol{m})$. The method we use is a Bayesian Neural Network (BNN)[33] approximated through variational inference,[34,35] which we have called *Semi-Conditional Variational Auto-Encoder* (SCVAE). A detailed description of the SCVAE method for reconstruction and associated uncertainty quantification is given in Sec. III B.

Here, we focus on fluid flow, being the main driving mechanism behind transport and dilution of tracers in marine waters. The world's oceans are under tremendous stress,[36] UN has declared 2021–2030 as the ocean decade,[80] and an ecosystem based Marine Spatial Planning initiative has been launched by the Intergovernmental Oceanographic Commission of UNESCO (IOC).[37]

Local and regional current conditions determine transport of tracers in the ocean.[38,39] Examples are accidental release of radioactive, biological, or chemical substances from industrial complexes, e.g., organic waste from fish farms in Norwegian fjords,[40] plastic,[41] or other contaminants that might have adverse effects on marine ecosystems.[42]

The ability to predict the environmental impact of a release, i.e., concentrations as a function of distance and direction from the source, requires reliable current conditions.[43,44] Subsequently, these transport predictions support design of marine environmental monitoring programs.[45–48] The aim here is to model current conditions in a probabilistic manner using SCVAEs. This allows for predicting footprints in a Monte Carlo framework, providing simulated data for training networks used for analyzing environmental time series.[49]

We will compare results with the GPOD method.[50] We are aware that there are recent methods (e.g., PINNS and traditional autoencoder) that may perform better on the specific datasets than the GPOD; however, the simplicity, versatility, and, not least, popularity of GPOD[5,51,52] make it a great method for comparison.

The reminder of this manuscript is outlined as follows: Sec. II presents a motivating example for the SCVAE-method in comparison with the GPOD-method. In Sec. III, we review both the VAE and CVAE methods and present the SCVAE. Results from experiments on two different datasets are presented in Sec. IV. Section V summarizes and discusses the method, experiments, drawbacks, benefits, potential extensions, and further work.

II. A MOTIVATING EXAMPLE

Here, we illustrate the performance of the proposed method vs the GPOD method in order to give a motivation for this study. We use data from simulations of a two dimensional viscous flow around a cylinder at the Reynolds number of 160 obtained from Ref. 53. The simulations were performed by Weinkauf and Theisel[54] with the Gerris Flow Solver software.[55] The dataset consists of a two velocity components, u and v, on an uniform 400 × 50 × 1001 grid of the $[-0.5, 7.5] \times [-0.5, 0.5] \times [15, 23]$ spatial–temporal domain. The simulations are run from $t = 0$ to $t = 23$, but velocities are only extracted from $t = 15$ to $t = 23$. In particular, we have 400 points in the horizontal direction, 50 points in the vertical direction, and 1001 points in time. The cylinder has a diameter of 0.125 and is centered at the origin (see Fig. 2).

The left vertical boundary (inlet) has the Dirichlet boundary condition, $u = 1$ and $v = 0$. The homogeneous Neumann boundary condition is imposed at the right boundary (outlet) and with homogeneous Dirichlet conditions on the remaining boundaries. At the start of the simulation, $t = 0$, both velocities were equal to zero. We plot a snapshot of the velocities in Fig. 2.

In the experiment below, we extract the data downstream from the cylinder, that is, from grid points 40–200 in the horizontal direction, and keep all grid points in the vertical direction. Hence, $\mathcal{P}$ contains $N = 8000$ points, 160 points in the horizontal direction, and 50 in the vertical direction. The temporal resolution is kept as before, that is, the number of time steps in $\mathcal{T}$ is $K = 1001$.

For validation purposes, the dataset was split into a train, validation, and test dataset. For both the SCVAE and the GPOD, the goal was to minimize the L_2 error between the true and the modeled flow state. The restriction of the GPOD is that the number of components r can be at most $2M$. To deal with this problem and to account for the flow incompressibility, we added the regularization term $\lambda \| L_{div} x^{(i)} \|$, $\lambda > 0$, to the objective function (see Appendix A). For the GPOD method, the parameters r and λ are optimized on the validation dataset in order to have the smallest mean error. We give more details about objective functions for the SCVAE in Sec. III B. For now, we mention that there are two versions, where one version uses an additional divergence regularization term similar to GPOD.

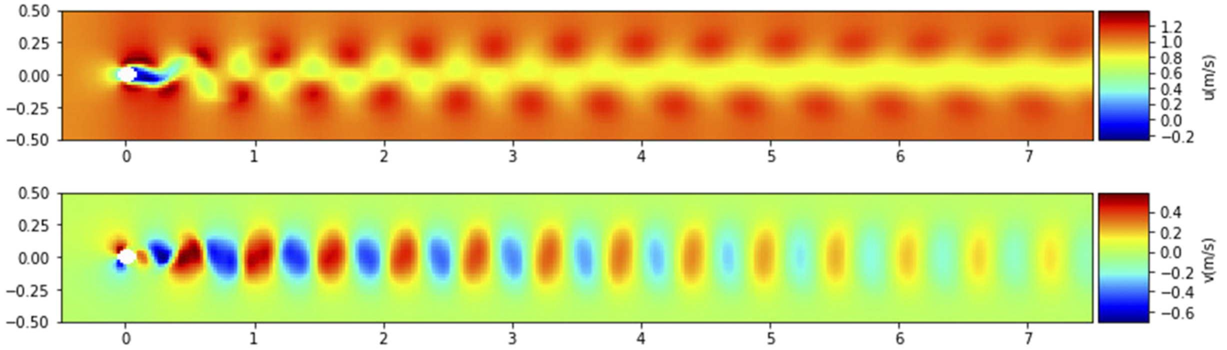

FIG. 2. A spatial snapshot at the time $t \approx 19$ (a time step of 500) of the flow field from the original 2D flow around a cylinder dataset[53] with u and v components presented in the upper and lower panels, respectively.

In Fig. 3, we plot the mean of the relative L_2 error calculated on the test data for both methods with and without the div-regularization (in-compressible fluid). The results are presented for three, four, and five measurement locations, that is, $M = 3, 4, 5$. For each of these three cases, we selected 20 randomly different configurations of M. In particular, we created 20 subgrids $\mathcal{Q}$, each containing five randomly sampled spatial grid points. Next, we removed one and then two points from each of the 20 subgrids $\mathcal{Q}$ to create new subgrids of 4 and 3 measurements, respectively. The figure is a box-and-whisker plot of these 20 configurations, where outliers are defined as data points that are outside 1.5 times the inter quartile range from the upper and lower quantiles.

As shown in Fig. 3, both methods perform well with 5 measurements. The resulting relative errors have comparable mean and variance. When reducing the number of observations, the SCVAE method maintains low errors, while the GPOD error increases. The SCVAE seems to benefit from the additional regularization of minimizing the divergence in terms of lower error and less variation in the error estimates. The effect is more profound with fewer measurements.

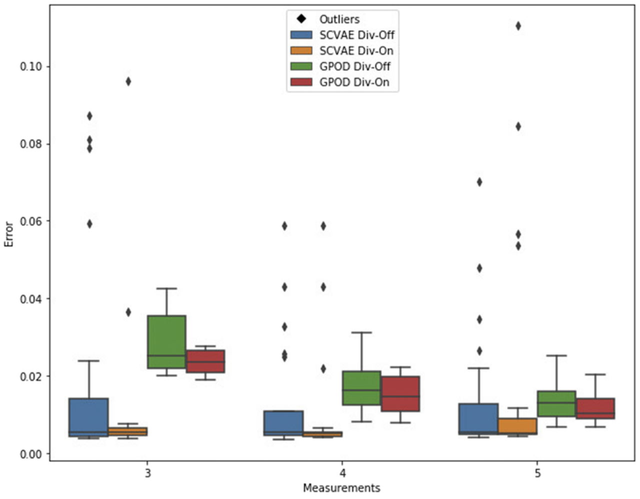

FIG. 3. The mean relative error for two reconstruction methods. The orange and blue label correspond to the SCVAE with (div-on) and without (div-off) additional divergence regularization. The green and red labels correspond to the GPOD method. The figure is a box-and-whisker plot of the 20 configurations, where outliers are defined as data points that are outside 1.5 times the interquartile range from the upper and lower quantiles (25/75).

The key benefit of the SCVAE is that its predictions are optimal for the given measurement locations. In contrast, the POD based approaches and particularly the GPOD create a set of basis functions (principal components) based on the training data independently of the measurements. While this has an obvious computational advantage, the number of principle components for complex flows can be high and, as a result, many more measurements are needed.[6,50,56] There are a number of algorithms that aim to optimize the measurement locations to achieve the best performance of the POD based methods (see Refs. 17, 50, and 51). In practice, however, the locations are often fixed and other approaches are needed. The results in Fig. 3 suggest that the SCVAE could be one of these approaches.

III. METHODS

Before we introduce the model used for reconstruction of flows, we give a brief introduction to VAEs and CVAEs. For a detailed introduction, see Ref. 57. VAEs are neural network models that has been used for learning structured representations in a wide variety of applications, e.g., image generation,[58] interpolation between sentences,[59] and compressed sensing.[26]

A. Variational auto-encoders (VAE)

Let us assume that the dataset $\boldsymbol{X}$ is generated by a random process that involves an unobserved continuous random variable $\boldsymbol{z}$. The process consists of two steps: (i) a value $\boldsymbol{z}^{(i)}$ is sampled from a prior $p_{\theta^*}(\boldsymbol{z})$ and (ii) $\boldsymbol{x}^{(i)}$ is generated from a conditional distribution, $p_{\theta^*}(\boldsymbol{x}|\boldsymbol{z})$. In the case of flow reconstruction, $\boldsymbol{z}$ could be unknown boundary or initial conditions, tidal and wind forcing, etc. However, generally, $\boldsymbol{z}$ is just a convenient construct to represent $\boldsymbol{X}$, rather than a physically explained phenomena. Hence, it is, for convenience, assumed that $p_{\theta^*}(\boldsymbol{z})$ and $p_{\theta^*}(\boldsymbol{x}|\boldsymbol{z})$ come from parametric families of distributions $p_\theta(\boldsymbol{z})$ and $p_\theta(\boldsymbol{x}|\boldsymbol{z})$, and their density functions are differentiable almost everywhere with respect to both $\boldsymbol{z}$ and θ. A probabilistic autoencoder is a neural network that is trained to represent its input $\boldsymbol{X}$ as $p_\theta(\boldsymbol{x})$ via *latent representation* $\boldsymbol{z} \sim p_\theta(\boldsymbol{z})$, that is,

$$p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z} = \int p_\theta(\boldsymbol{x}|\boldsymbol{z}) p_\theta(\boldsymbol{z}) d\boldsymbol{z}. \tag{3}$$

As $p_\theta(\boldsymbol{z})$ is unknown and observations $\boldsymbol{z}^{(i)}$ are not accessible, we must use $\boldsymbol{X}$ in order to generate $\boldsymbol{z} \sim p_\theta(\boldsymbol{z}|\boldsymbol{x})$. That is, the network can be viewed as consisting of two parts: an *encoder* $p_\theta(\boldsymbol{z}|\boldsymbol{x})$ and a *decoder* $p_\theta(\boldsymbol{x}|\boldsymbol{z})$. Typically, the true posterior distribution $p_\theta(\boldsymbol{z}|\boldsymbol{x})$ is intractable but could be approximated with variational inference.[34,35] That is, we define the so-called recognition model $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ with variational parameters ϕ, which aims to approximate $p_\theta(\boldsymbol{z}|\boldsymbol{x})$. The recognition model is often parameterized as a Gaussian. Thus, the problem of estimating $p_\theta(\boldsymbol{z}|\boldsymbol{x})$ is reduced to finding the best possible estimate for ϕ, effectively turning the problem into an optimization problem.

An autoencoder that uses a recognition model is called Variational Auto-Encoder (VAE). In order to get good prediction, we need to estimate the parameters ϕ and θ. The marginal likelihood is equal to the sum over the marginal likelihoods of the individual samples, that is, $\sum_{i=1}^{K} \log p_\theta(\boldsymbol{x}^{(i)})$. Therefore, we further on present estimates for an individual sample. The Kullback–Leibler divergence between two probability distributions $q_\phi(\boldsymbol{z}|\boldsymbol{x}^{(i)})$ and $p_\theta(\boldsymbol{z}|\boldsymbol{x}^{(i)})$, defined as

$$D_{KL}\Big[q_\phi\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\Big) \| p_\theta\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\Big)\Big] = \int q_\phi\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\Big) \log\left(\frac{q_\phi\big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\big)}{p_\theta\big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\big)}\right) d\boldsymbol{z},$$

can be interpreted as a measure of distinctiveness between these two distributions.[60] It can be shown (see Ref. 57) that

$$\log p_\theta\Big(\boldsymbol{x}^{(i)}\Big) = D_{KL}\Big[q_\phi\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\Big) \| p_\theta\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\Big)\Big] + \mathcal{L}\Big(\theta, \phi; \boldsymbol{x}^{(i)}\Big), \tag{4}$$

where

$$\mathcal{L}\Big(\theta, \phi; \boldsymbol{x}^{(i)}\Big) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x}^{(i)})}\Big[-\log q_\phi\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\Big) + \log p_\theta\Big(\boldsymbol{x}^{(i)}, \boldsymbol{z}\Big)\Big].$$

Since KL-divergence is non-negative, we have $\log p_\theta(\boldsymbol{x}^{(i)}) \geq \mathcal{L}(\theta, \phi; \boldsymbol{x}^{(i)})$ and $\mathcal{L}(\theta, \phi; \boldsymbol{x}^{(i)})$ is called Evidence Lower Bound (ELBO) for the marginal likelihood $\log p_\theta(\boldsymbol{x}^{(i)})$. Thus, instead of maximizing the marginal probability, one can instead maximize its variational lower bound to which we also refer to as an objective function. It can be further shown that the ELBO can be written as

$$\mathcal{L}\Big(\theta, \phi; \boldsymbol{x}^{(i)}\Big) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x}^{(i)})}\Big[\log p_\theta\Big(\boldsymbol{x}^{(i)}|\boldsymbol{z}\Big)\Big] - D_{KL}\Big[q_\phi\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\Big) \| p_\theta(\boldsymbol{z})\Big]. \tag{5}$$

Reformulating the traditional VAE framework as a constraint optimization problem, it is possible to obtain the β-VAE[61] objective function if $p_\theta(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$,

$$\mathcal{L}\Big(\theta, \phi; \boldsymbol{x}^{(i)}\Big) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x}^{(i)})}\Big[\log p_\theta\Big(\boldsymbol{x}^{(i)}|\boldsymbol{z}\Big)\Big] - \beta D_{KL}\Big[q_\phi\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\Big) \| p_\theta(\boldsymbol{z})\Big], \tag{6}$$

where $\beta > 0$. Here, β is a regularization coefficient that constrains the capacity of the latent representation $\boldsymbol{z}$. The $\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x}^{(i)})}\big[\log p_\theta(\boldsymbol{x}^{(i)}|\boldsymbol{z})\big]$ can be interpreted as the reconstruction term, while the KL-term can be interpreted $\beta D_{KL}[q_\phi(\boldsymbol{z}|\boldsymbol{x}^{(i)}) \| p_\theta(\boldsymbol{z})]$ as a regularization term.

Conditional Variational Auto-Encoders[32] (CVAE) are similar to VAEs but differ by conditioning on an additional property of the data (e.g., a label or class), here denoted $\boldsymbol{c}$. Conditioning both the recognition model and the true posteriori on both $\boldsymbol{x}^{(i)}$ and $\boldsymbol{c}$ results in the CVAE ELBO,

$$\mathcal{L}\Big(\theta, \phi; \boldsymbol{x}^{(i)}, \boldsymbol{c}\Big) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x}^{(i)}, \boldsymbol{c})}\Big[\log p_\theta\Big(\boldsymbol{x}^{(i)}|\boldsymbol{z}, \boldsymbol{c}\Big)\Big] - D_{KL}\Big[q\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}, \boldsymbol{c}\Big) \| p_\theta(\boldsymbol{z}|\boldsymbol{c})\Big]. \tag{7}$$

In the decoding phase, CVAE allows for conditional probabilistic reconstruction and permits sampling from the conditional distribution $p_\theta(\boldsymbol{z}|\boldsymbol{c})$, which has been useful for generative modeling of data with known labels (see Ref. 32). Here, we investigate a special case of the CVAE when $\boldsymbol{c}$ is a partial observation of $\boldsymbol{x}$. We call this the Semi-Conditional Variational Auto-Encoder (SCVAE).

B. Semi-conditional variational auto-encoder

The SCVAE takes the input data $\boldsymbol{X}$, conditioned on $\boldsymbol{M}$, and approximates the probability distribution $p_\theta(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{m})$. Then, we can generate $\boldsymbol{x}^{(i)}$ based on the observations $\boldsymbol{m}^{(i)}$ and latent representation $\boldsymbol{z}$. As $\boldsymbol{m}^{(i)} = C\boldsymbol{x}^{(i)}$, where C is a non-stochastic sampling matrix, we have

$$p_\theta\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}, \boldsymbol{m}^{(i)}\Big) = p_\theta\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\Big), \text{ and } q_\phi\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}, \boldsymbol{m}^{(i)}\Big) = q_\phi\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\Big).$$

Therefore, from Eq. (7), the ELBO for SCVAE is

$$\log p_\theta\Big(\boldsymbol{x}^{(i)}|\boldsymbol{m}^{(i)}\Big) \geq \mathcal{L}\Big(\theta, \phi; \boldsymbol{x}^{(i)}, \boldsymbol{m}^{(i)}\Big) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x}^{(i)})}\Big[\log p_\theta\Big(\boldsymbol{x}^{(i)}|\boldsymbol{z}, \boldsymbol{m}^{(i)}\Big)\Big] - D_{KL}\Big[q_\phi\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\Big) \| p_\theta\Big(\boldsymbol{z}|\boldsymbol{m}^{(i)}\Big)\Big], \tag{8}$$

where $p_\theta(\boldsymbol{z}|\boldsymbol{m}^{(i)}) = \mathcal{N}(\boldsymbol{0}, \mathbf{I})$. Similarly, as for the β-VAE,[61] we can obtain a relaxed version of Eq. (8) by maximizing the parameters $\{\phi, \theta\}$ of the expected log-likelihood $\mathbb{E}_{q_\phi(\cdot)}[\log p_\theta(\boldsymbol{x}^{(i)}|\boldsymbol{m}^{(i)}, \boldsymbol{z})]$ and treat it as a constrained optimization problem,

$$\max_{\phi,\theta} \mathbb{E}_{q_\phi(\cdot)}\Big[\log p_\theta\Big(\boldsymbol{x}^{(i)}|\boldsymbol{m}^{(i)}, \boldsymbol{z}\Big)\Big] \quad \text{subject to} \quad D_{KL}\Big(q_\phi\Big(\boldsymbol{z}|\boldsymbol{m}^{(i)}, \boldsymbol{x}^{(i)}\Big) \| p_\theta\Big(\boldsymbol{z}|\boldsymbol{m}^{(i)}\Big)\Big) \leq \epsilon, \tag{9}$$

where $\epsilon > 0$ is small. The subscript $q_\phi(\cdot)$ is short for $q_\phi(\boldsymbol{z}|\boldsymbol{m}^{(i)}, \boldsymbol{x}^{(i)})$. Since $\boldsymbol{m}^{(i)}$ is dependent on $\boldsymbol{x}^{(i)}$, we have $q_\phi(\boldsymbol{z}|\boldsymbol{m}^{(i)}, \boldsymbol{x}^{(i)}) = q_\phi(\boldsymbol{z}|\boldsymbol{x}^{(i)})$. Equation (9) can be expressed as a Lagrangian under the Karush–Kuhn–Tucker (KKT) conditions.[62,63] Hence,

$$\mathcal{F}\Big(\theta, \phi, \beta, \alpha, \boldsymbol{x}^{(i)}, \boldsymbol{m}^{(i)}\Big) = \mathbb{E}_{q_\phi(\cdot)}\Big[\log p_\theta\Big(\boldsymbol{x}^{(i)}|\boldsymbol{m}^{(i)}, \boldsymbol{z}\Big)\Big] + \beta\Big(D_{KL}\Big(q_\phi\Big(\boldsymbol{z}|\boldsymbol{x}^{(i)}\Big) \| p_\theta\Big(\boldsymbol{z}|\boldsymbol{m}^{(i)}\Big) - \epsilon\Big)\Big). \tag{10}$$

According to the complementary slackness KKT condition $\beta \geq 0$, we can rewrite Eq. (10) as

$$\begin{aligned}\mathcal{F}\left(\theta,\phi,\beta,\boldsymbol{x}^{(i)},\boldsymbol{m}^{(i)}\right) &\geq \mathcal{L}\left(\theta,\phi,\boldsymbol{x}^{(i)},\boldsymbol{m}^{(i)}\right)\\ &= \mathbb{E}_{q_\phi(\cdot)}\left[\log p_\theta\left(\boldsymbol{x}^{(i)}|\boldsymbol{m}^{(i)},\boldsymbol{z}\right)\right]\\ &\quad + \beta D_{KL}\left(q_\phi\left(\boldsymbol{z}|\boldsymbol{x}^{(i)}\right)\|p_\theta\left(\boldsymbol{z}|\boldsymbol{m}^{(i)}\right)\right). \qquad (11)\end{aligned}$$

The objective functions in Eqs. (8) and (11), later in Eq. (13), show that if conditioning on a feature that is a known function of the original data, such as measurements, we do not need to account for them in the encoding phase. The measurements are then coupled with the encoded data in the decoder. We sketch the main components of the SCVAE in Fig. 4.

In order to preserve some physical properties of the data $\boldsymbol{X}$, we can condition yet on another feature. Here, we utilize the incompressibility property of the fluid, i.e., $\boldsymbol{d}^{(i)} = L_{div}\boldsymbol{x}^{(i)} \approx 0$ [see Eq. (2)]. We intend to maximize a log-likelihood under an additional constrain $\boldsymbol{d}^{(i)}$ compared to Eq. (9). That is,

$$\max_{\phi,\theta} \mathbb{E}_{q_\phi(\cdot)}\left[\log p_\theta\left(\boldsymbol{x}^{(i)}|\boldsymbol{m}^{(i)},\boldsymbol{z}\right)\right]$$

subject to

$$D_{KL}\left(q_\phi\left(\boldsymbol{z}|\boldsymbol{x}^{(i)}\right)\|p_\theta\left(\boldsymbol{z}|\boldsymbol{m}^{(i)},\boldsymbol{d}^{(i)}\right)\right) \leq \epsilon$$

and

$$-\mathbb{E}_{q_\phi(\cdot)}\left[\log p_\theta\left(\boldsymbol{d}^{(i)}|\boldsymbol{m}^{(i)},\boldsymbol{z}\right)\right] \leq \delta, \qquad (12)$$

where $\epsilon, \delta > 0$ are small. Equation (12) can expressed as a Lagrangian under the Karush–Kuhn–Tucker (KKT) conditions as before, and as a consequence of the complementary slackness condition $\lambda, \beta \geq 0$, we can obtain the objective function

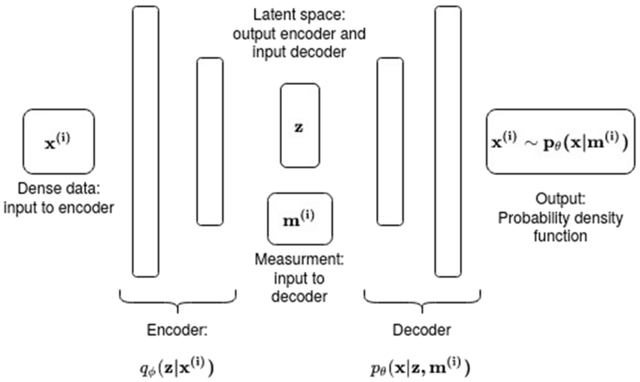
Latent space: output encoder and input decoder
$\mathbf{z}$
$\mathbf{x}^{(i)}$
Dense data: input to encoder
$\mathbf{m}^{(i)}$
Measurment: input to decoder
$\mathbf{x}^{(i)} \sim \mathbf{p}_\theta(\mathbf{x}|\mathbf{m}^{(i)})$
Output: Probability density function
Encoder:
$q_\phi(\mathbf{z}|\mathbf{x}^{(i)})$
Decoder
$p_\theta(\mathbf{x}|\mathbf{z},\mathbf{m}^{(i)})$

FIG. 4. The figure shows a sketch of the model used to estimate $p_\theta(\boldsymbol{x}|\boldsymbol{m}^{(i)})$.

$$\begin{aligned}&\mathcal{F}\left(\theta,\phi,\beta,\alpha,\boldsymbol{x}^{(i)},\boldsymbol{m}^{(i)},\boldsymbol{d}^{(i)}\right)\\ &\quad\geq \mathcal{L}\left(\theta,\phi,\boldsymbol{x}^{(i)},\boldsymbol{m}^{(i)},\boldsymbol{d}^{(i)}\right)\\ &\quad= \mathbb{E}_{q_\phi(\cdot)}\left[\log p_\theta\left(\boldsymbol{x}^{(i)}|\boldsymbol{m}^{(i)},\boldsymbol{z}\right)\right]\\ &\qquad + \lambda\,\mathbb{E}_{q_\phi(\cdot)}\left[\log p_\theta\left(\boldsymbol{d}^{(i)}|\boldsymbol{m}^{(i)},\boldsymbol{z}\right)\right]\\ &\qquad - \beta D_{KL}\left(q_\phi\left(\boldsymbol{z}|\boldsymbol{x}^{(i)}\right)\|p_\theta\left(\boldsymbol{z}|\boldsymbol{m}^{(i)},\boldsymbol{d}^{(i)}\right)\right), \qquad (13)\end{aligned}$$

where $p(\boldsymbol{z}|\boldsymbol{m}^{(i)},\boldsymbol{d}^{(i)}) = \mathcal{N}(\boldsymbol{0},\mathbf{I})$. For convenience of notation, we refer to the objective function [Eq. (11)] as the case with $\lambda = 0$ and the objective function [Eq. (13)] as the case with $\lambda > 0$. Observe that under the Gaussian assumptions on the priors, Eq. (13) is equivalent to Eq. (11) if $\lambda = 0$. Thus, from now on, we will refer to it as a special case of Eq. (13) and denote as $\mathcal{L}_0$. Optimizing Eq. (13) can be interpreted as a multi task learning,[64] i.e., we optimize the network to solve more than one task at a time. Multitask learning reduces the risk of overfitting[65] and produces more consistent results.[64]

Similar to Ref. 31, we obtain $q_\phi(\boldsymbol{z}|\boldsymbol{x}^{(i)}) = \mathcal{N}(\boldsymbol{\mu}^{(i)}, \mathrm{diag}(\boldsymbol{\sigma}^{(i)})^2)$, that is, $\phi = \{\boldsymbol{\mu}^{(i)}, \boldsymbol{\sigma}^{(i)}\}$. This allows us to express the KL-divergence terms in a closed form and avoid issues related to differentiability of the ELBOs. Under these assumptions, the KL-divergence terms can be integrated analytically, while the terms $\mathbb{E}_{q_\phi(z|\boldsymbol{x}^{(i)})}\left[\log p_\theta(\boldsymbol{x}^{(i)}|\boldsymbol{z},\boldsymbol{m}^{(i)})\right]$ and $\mathbb{E}_{q_\phi(z|\boldsymbol{x}^{(i)})}\left[\log p_\theta(\boldsymbol{d}^{(i)}|\boldsymbol{z},\boldsymbol{m}^{(i)})\right]$ require estimation by sampling,

$$\begin{aligned}\mathbb{E}_{q_\phi(z|\boldsymbol{x}^{(i)})}\left[\log p_\theta\left(\boldsymbol{x}^{(i)}|\boldsymbol{z},\boldsymbol{m}^{(i)}\right)\right] &\approx \frac{1}{L}\sum_{l=1}^{L}\log p_\theta\left(\boldsymbol{x}^{(i)}|\boldsymbol{z}^{(i,l)},\boldsymbol{m}^{(i)}\right),\\ \mathbb{E}_{q_\phi(z|\boldsymbol{x}^{(i)})}\left[\log p_\theta\left(\boldsymbol{d}^{(i)}|\boldsymbol{z},\boldsymbol{m}^{(i)}\right)\right] &\approx \frac{1}{L}\sum_{l=1}^{L}\log p_\theta\left(\boldsymbol{d}^{(i)}|\boldsymbol{z}^{(i,l)},\boldsymbol{m}^{(i)}\right),\end{aligned}$$

where

$$\boldsymbol{z}^{(i,l)} = g_\phi\left(\boldsymbol{\epsilon}^{(i,l)},\boldsymbol{x}^{(i)}\right), \quad \boldsymbol{\epsilon}^{l} \sim p(\boldsymbol{\epsilon}). \qquad (14)$$

Here, $\boldsymbol{\epsilon}^{l}$ is an auxiliary (noise) variable with independent marginal $p(\boldsymbol{\epsilon})$, and $g_\phi(\cdot)$ is a differentiable transformation of $\boldsymbol{\epsilon}$, parametrized by ϕ (see Ref. 31 for details). We denote $\mathcal{L}_\lambda, \lambda \geq 0$ [Eq. (13)], with the approximation above as $\widehat{\mathcal{L}}_\lambda$, that is,

$$\begin{aligned}\widehat{\mathcal{L}}_\lambda\left(\theta,\phi,\boldsymbol{x}^{(i)},\boldsymbol{m}^{(i)},\boldsymbol{d}^{(i)}\right) &= \frac{1}{L}\sum_{l=1}^{L}\log p_\theta\left(\boldsymbol{x}^{(i)}|\boldsymbol{z}^{(i,l)},\boldsymbol{m}^{(i)}\right)\\ &\quad + \lambda\frac{1}{L}\sum_{l=1}^{L}\log p_\theta\left(\boldsymbol{d}^{(i)}|\boldsymbol{z}^{(i,l)},\boldsymbol{m}^{(i)}\right)\\ &\quad - \beta D_{KL}\left[q_\phi\left(\boldsymbol{z}|\boldsymbol{x}^{(i)}\right)\|p_\theta\left(\boldsymbol{z}|\boldsymbol{m}^{(i)},\boldsymbol{d}^{(i)}\right)\right].\end{aligned} \qquad (15)$$

The objective function $\widehat{\mathcal{L}}_\lambda$ can be maximized by gradient descent. Since the gradient $\nabla_{\theta,\phi}\widehat{\mathcal{L}}_\lambda$ cannot be calculated for large datasets, Stochastic gradient descent methods (see Refs. 66 and 67) are typically used where

$$\begin{aligned}\widehat{\mathcal{L}}_\lambda(\theta,\phi;\boldsymbol{X},\boldsymbol{M},\boldsymbol{D}) &\approx \widehat{\mathcal{L}}^{R}\left(\theta,\phi;\boldsymbol{X}^{R},\boldsymbol{M}^{R},\boldsymbol{D}^{R}\right)\\ &= \frac{K}{R}\sum_{r=1}^{R}\widehat{\mathcal{L}}_\lambda\left(\theta,\phi;\boldsymbol{x}^{(i_r)},\boldsymbol{m}^{(i_r)},\boldsymbol{d}^{(i_r)}\right), \quad \lambda \geq 0. \quad (16)\end{aligned}$$

Here, $\boldsymbol{X}^R = \left\{\boldsymbol{x}^{(i_r)}\right\}_{r=1}^{R}$, $R < K$ is a mini-batch consisting of randomly sampled datapoints, $\boldsymbol{M}^R = \left\{\boldsymbol{m}^{(i_r)}\right\}_{r=1}^{R}$, and $\boldsymbol{D}^R = \left\{\boldsymbol{d}^{(i_r)}\right\}_{r=1}^{R}$. After the network is optimized, a posterior predictive distribution $p_\theta(\boldsymbol{x}|\boldsymbol{m}, \boldsymbol{d})$ can be approximated with a Monte Carlo estimator.

1. Uncertainty quantification

Let $\hat{\theta}$ and $\hat{\phi}$ be an estimation of generative and variational parameters, as described in Sec. III B. Then, the decoder can be used to predict the posterior as

$$p_{\hat{\theta}}\left(\boldsymbol{x}|\boldsymbol{m}^*, \boldsymbol{d}^*\right) \approx \frac{1}{N_{MC}} \sum_{j=1}^{N_{MC}} p_{\hat{\theta}}\left(\boldsymbol{x}|\boldsymbol{z}^{(j)}, \boldsymbol{m}^*, \boldsymbol{d}^*\right)$$

$$\xrightarrow[N_{MC}\to\infty]{\int} p_{\hat{\theta}}\left(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{m}^*, \boldsymbol{d}^*\right) p_{\hat{\theta}}\left(\boldsymbol{z}|\boldsymbol{m}^*, \boldsymbol{d}^*\right) d\boldsymbol{z}. \tag{17}$$

While sampling from the latent space has been viewed typically as an approach for generating new samples with similar properties, here, we use it to estimate the prediction uncertainty of the trained model. From Eq. (17), we are able to estimate the mean prediction $\hat{\boldsymbol{x}}^*$ and empirical covariance matrix $\widehat{\boldsymbol{\Sigma}}$ using a Monte Carlo estimator. We get

$$\widehat{\boldsymbol{x}}^* = \frac{1}{N_{MC}} \sum_{j=1}^{N_{MC}} \boldsymbol{x}^{(j)}$$

and

$$\widehat{\boldsymbol{\Sigma}} = \frac{1}{N_{MC} - 1} \sum_{j=1}^{N_{MC}} \left(\boldsymbol{x}^{(j)} - \widehat{\boldsymbol{x}}^*\right)\left(\boldsymbol{x}^{(j)} - \widehat{\boldsymbol{x}}^*\right)^T, \tag{18}$$

where $\boldsymbol{x}^{(j)} \sim p_{\hat{\theta}}(\boldsymbol{x}|\boldsymbol{m}^*, \boldsymbol{d}^*)$. The empirical standard deviation is then $\widehat{\boldsymbol{\sigma}} = \sqrt{diag(\widehat{\boldsymbol{\Sigma}})}$. To estimate the confidence region, we assume that the predicted $p_{\hat{\theta}}(\boldsymbol{x}|\boldsymbol{m}^*, \boldsymbol{d}^*)$ is well approximated by a normal distribution $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Given that $\widehat{\boldsymbol{x}}^*$ and $\widehat{\boldsymbol{\Sigma}}$ are approximations of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, obtained from N_{MC} samples as above, a confidence region estimate for a prediction $\boldsymbol{x}^*$ can be given as

$$\left\{\boldsymbol{x}^{(i)} \in \mathbb{R}^{2N} : \left(\boldsymbol{x}^{(i)} - \widehat{\boldsymbol{x}}^*\right)^T \hat{\boldsymbol{\Sigma}}^+ (\boldsymbol{x}^{(i)} - \widehat{\boldsymbol{x}}^*) \le \chi_k^2(p)\right\}, \tag{19}$$

where $\chi_k^2(p)$ is the quantile function for probability p of the chi-squared distribution with $k = \min\{N_{MC}, 2N\}$ degrees of freedom and $\widehat{\boldsymbol{\Sigma}}^+$ is the pseudoinverse of $\widehat{\boldsymbol{\Sigma}}$. Using the singular value decomposition, $\widehat{\boldsymbol{\Sigma}} = \boldsymbol{USU}^T$, the corresponding interval for $(\boldsymbol{x}^*)_n$, $n = 1, \ldots, 2N$, is

$$\left[(\widehat{\boldsymbol{x}}^*)_n - \sqrt{\chi_k^2(p)}\, \|\boldsymbol{u}_n^T \boldsymbol{S}^{1/2}\|_2, \quad (\widehat{\boldsymbol{x}}^*)_n + \sqrt{\chi_k^2(p)}\, \|\boldsymbol{u}_n^T \boldsymbol{S}^{1/2}\|_2\right], \tag{20}$$

where $\boldsymbol{u}_n^T$ is nth row of the matrix $\boldsymbol{U}$.

IV. EXPERIMENTS

We will present the SCVAE method on two different datasets. The first one is the 2D flow around a cylinder described in Sec. II, and the second is ocean currents on the seafloor created by the Bergen Ocean Model (BOM).[68] The data $\boldsymbol{X}$ consist of the two dimensional velocities $\boldsymbol{w} = (u, v)$. To illustrate the results, we will plot u and v components of $\boldsymbol{x}^{(i)} \in \boldsymbol{X}$ [see Eq. (1)]. For validation of the models, the data $\boldsymbol{X}$ are split into train, test, and validation subsets, which are subscripted accordingly, if necessary. The datasets, spitting, and preprocessing for each case are described in Secs. IV A and IV B.

We use a schematically simple architecture to explore the SCVAE. The main ingredient of the encoder is the convolutional neural network (CNN),[69,70] and for the decoder, we use transposed CNN-layers.[71] The SCVAE has a slightly different architecture in each case, which we present in Appendix C.

The SCVAE is trained to maximize the objective function in Eq. (16) with the backpropagation algorithm[72] and the Adam algorithm.[73] We use an adaptive approach of weighing the reconstruction term with KL-divergence and/or divergence terms,[74] that is, finding the regularization parameters β and λ. Specifically, we calculate the proportion of contribution of each term to the total value of the objective function and scale the terms accordingly. This approach prevents posterior collapse. Posteriori collapse occurs if the KL-divergence term becomes too close to zero, resulting in a non-probabilistic reconstruction. The approach of weighing the terms proportionally iteratively adjusts the weight of the KL-divergence term, β, such that posterior collapse is mitigated. For the result shown here, we trained the SCVAEs with early stopping criteria of 50 epochs, i.e., the optimization is stopped if we do not see any improvement after 50 epochs and returns the best model. We use a two-dimensional Gaussian distribution for $p_\theta(\boldsymbol{z}|\boldsymbol{m}^{(i)}, \boldsymbol{d}^{(i)})$ in all the experiments.

Let the test data $\boldsymbol{X}_{test}$ consist of n instances $\boldsymbol{x}^{(i)}$, $i = 1, \ldots, n$, and $\widehat{\boldsymbol{x}}^{(i)}$ denote a prediction of the true $\boldsymbol{x}^{(i)}$ given $\boldsymbol{m}^{(i)}$. In the case of the SCVAE, $\widehat{\boldsymbol{x}}^{(i)}$ is the mean prediction obtained as in Eq. (18). For the GPOD method, $\widehat{\boldsymbol{x}}^{(i)}$ is a deterministic output of the optimization problem (see Appendix A). In order to compare the SCVAE results with the results of the GPOD method, we introduce the mean of the relative error for the prediction,

$$\mathcal{E} = \frac{1}{n} \sum_{i=1}^{n} \frac{\|\widehat{\boldsymbol{x}}^{(i)} - \boldsymbol{x}^{(i)}\|_2}{\|\boldsymbol{x}^{(i)}\|_2}, \tag{21}$$

and the mean of the absolute error for the divergence,

$$\mathcal{E}_{div} = \frac{1}{n} \sum_{i=1}^{n} \|L_{div}\, \boldsymbol{x}^{(i)}\|_2. \tag{22}$$

A. 2D flow around a cylinder

Here, we return to the example in Sec. II. In the following, we give some additional details of the data preprocessing and model implementation.

1. Preprocessing

The data are reduced, as described in Sec. II. We assess the SCVAE with a sequential split for train, test, and validation: the last 15% of the data is used for test, the last 30% of the remaining data is used for validation, and the first 70% is used for training. To improve the conditioning of the optimization problem, we scale the data as described in Appendix B. The errors, $\mathcal{E}$ [Eq. (21)] and $\mathcal{E}_{div}$ [Eq. (22)], are calculated after re-scaling the data back. The input to the SVAE $\boldsymbol{x}^{(i)}$ was reshaped as an array with dimension (160 × 50 × 2) in order to apply convolutional layers. Here, we use five, four, three, and two fixed spatial measurements, that is, four different subgrids $\mathcal{Q}$,

$$\begin{aligned} \mathcal{Q}_5 &= \{(12,76),(47,8),(30,40),(153,34),(16,10)\}, \\ \mathcal{Q}_4 &= \{(12,76),(47,8),(30,40),(153,34)\}, \\ \mathcal{Q}_3 &= \{(12,76),(47,8),(30,40)\}, \\ \mathcal{Q}_2 &= \{(12,76),(47,8)\}. \end{aligned} \tag{23}$$

The flow state at these specific locations constitutes $\boldsymbol{M}$.

2. Model

A schematic description of the model is given in Appendix C. The first layer of the encoder is a zero-padding layer that expands the horizontal and vertical dimensions by adding zeros on the boundaries, a stencil of four in the horizontal direction and three in the vertical direction. The subsequent layers consist of two convolutional layers, where the first and second layers have 160 and 200 filters, respectively. We use a kernel size and strides of 2 in both convolutional layers and ReLu activation functions. This design compresses the data into a (42 × 14 × 200) shape. The compressed representation from the convolutional layers is flattened and is further compressed into a 64 dimensional vector through a traditional dense layer. Two outputs layers are defined to represent the mean and log-variance of the latent representation $\boldsymbol{z}$. The reparameterization trick is realized in a third layer, the so-called lambda layer, which takes the mean and log-variance as an input and generates $\boldsymbol{z}$. The output of the encoder are the samples $\boldsymbol{z}^{(i)}$ and the mean and the log-variance of $\boldsymbol{z}^{(i)}$.

The decoder takes the latent representation $\boldsymbol{z}^{(i)}$ and the measurements $\boldsymbol{m}^{(i)}$ as the input. The input $\boldsymbol{m}^{(i)}$ is flattened and then concatenated with $\boldsymbol{z}^{(i)}$. The next layer is a dense layer with shape (42 × 14 × 200). Afterward, there are two transposed convolutional layers with filters of 200 and 160. The strides and the kernel size are the same as those for the encoder. The final layer is a transposed convolutional layer with the same dimension as the input to the encoder, the dimension of $\boldsymbol{x}^{(i)}$. A linear activation function is used for this output layer. The last layer of the model is a lambda layer that removes the zero-padding. In Sec. IV A 3, we show statistics of the probabilistic reconstruction and compare with the GPOD method.

3. Results

In Fig. 5, we have plotted the reconstructed velocity fields and associated statistics. The observations placements are shown as stars (black and white). The SCVAE with the objective function $\widehat{\mathcal{L}}_0$ [see Eq. (7)] was used for this prediction. To generate the posterior predictive distributions [Eq. (17)], we sample 100 realizations from

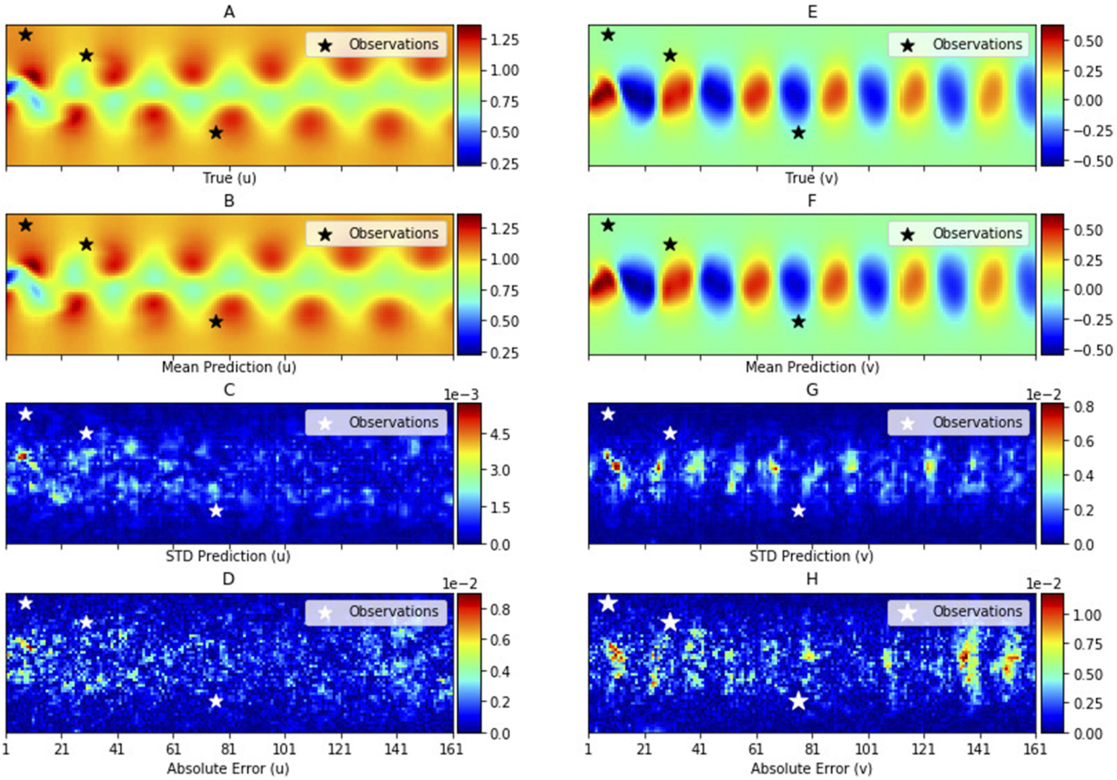

FIG. 5. (a) and (b) represent u-velocities and (e)–(h) represent v-velocities and associated statistical measures, respectively. The results are based on a model trained with $\lambda = 0$ and Q_3 measurement locations. [(a) and (e)] True solutions. [(b) and (f)] Reconstructed solutions. [(c) and (g)] Standard deviations. [(d) and (h)] Absolute errors.

$\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, which allows for calculation mean prediction and uncertainty estimates [see Eq. (18)].

We emphasize again that the SCVAEs with $\widehat{\mathcal{L}}_0$ and $\widehat{\mathcal{L}}_\lambda$, $\lambda > 0$, are two different models. For the sake of notation, we here refer to $\lambda = 0$ when we mean the model with the objective function in Eq. (7) and to $\lambda > 0$ when in Eq. (13). The same holds for the GPOD method (see Appendix B). When $\lambda = 0$, the number of the principle components r is less, $2M$. The number r is chosen such that the prediction on the validation data has the smallest possible error on average. If $\lambda > 0$, no restrictions on r are imposed. In this case, both λ and r are estimated from the validation data.

The results show that the SCVAE reconstructs the data well with the associated low uncertainty. This can be explained by the periodicity in the data. In particular, the training and validation datasets represent the test data well enough.

In Fig. 6, we have plotted four time series of the reconstructed test data at two specific grid points, together with the confidence regions constructed as in Eq. (20) with $p = 0.95$. Figures 6(a) and 6(c) represent the reconstruction at the grid point (6, 31), and Figs. 6(b) and 6(d) represent that at (101, 25) for u and v, respectively. The SCVAE reconstruction is significantly better than the GPOD and close to the true solution for all time steps.

Figure 7 shows the difference between the true values and the model prediction in time for the same two locations. This figure has to be seen in context with Fig. 5. The difference marginals are obtained based on the confidence region in Fig. 11. In Table I, we display the relative errors [Eq. (21)] for the SCVAE and the GPOD method, both with and without divergence regularization, for five, four, three, and two measurement locations given in Eq. (23).

The results of the SCVAE depend on two stochastic inputs, which are (i) randomness in the initialization of the prior weights and (ii) random mini-batch sampling. We have trained the model with each measurement configuration ten times and chose the model that performs the best on the validation dataset. Ideally, we would run test cases where we used all the values as measurements, i.e., $\boldsymbol{M} = \boldsymbol{X}$, and test how well the model would reconstruct in this case. This would then give us the lower bound of the best reconstruction that is possible for this specific architecture and hyper-parameter settings. However, this scenario was not possible to test due to limitations in the memory in the GPU. Therefore, we have used a large enough M, which still allowed us to run the model. In particular, we used every fifth and second pixel in the horizontal and vertical directions, which resulted in a total of (32 × 25) measurement locations, or $M = 800$. We believe that training the model with these settings gave us a good indication of the lower bound of the reconstruction error. The error observed was of the magnitude of 10^{-3}.

This lower bound has been reached for all measurement configurations [see Eq. (23)]. However, a larger computational cost was needed to reach the lower bound for fewer measurement locations. Figure 8 shows the number of epochs as a boxplot diagram. For each measurement configuration and regularization technique, the model is run ten times. The variation of the number of epochs for each measurement locations is due to different priors of the weights and random mini-batch sampling. In comparison with GPOD, the

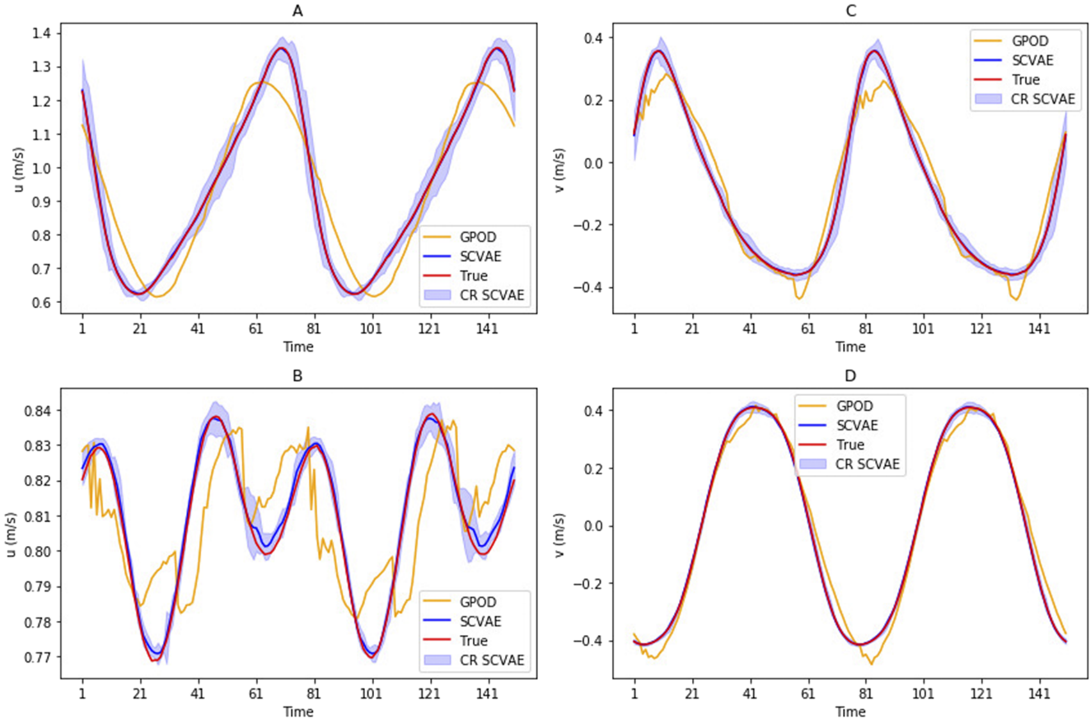

FIG. 6. Velocities u [(a) and (c)] and v [(b) and (d)] at grid points (6, 31) and (101, 25) with associated confidence regions, respectively. The estimates are based on a model trained with $\lambda = 0$ and Q_3 measurement locations.

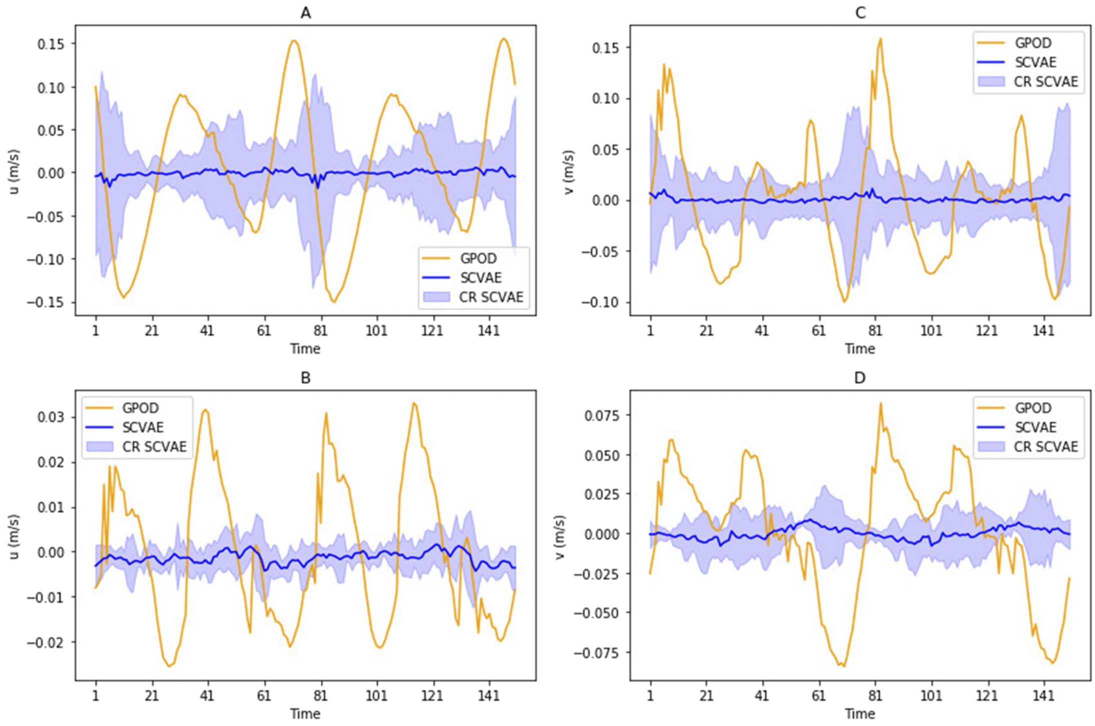

FIG. 7. The difference between the true and predicted u [(a) and (c)] and v [(b) and (d)] at grid points (6, 31) and (101, 25) with associated difference marginals, respectively. The estimates are based on a model trained with $\lambda = 0$ and Q_3 measurement locations.

TABLE I. The mean relative error $\mathcal{E}$ [Eq. (21)] for the SCVAE prediction and the GPOD prediction with or without div-regularization and different number of measurements.

		Measurement locations			
Method	Regularization	5	4	3	2
SCVAE	$\lambda = 0$	0.30×10^{-2}	0.33×10^{-2}	0.26×10^{-2}	0.28×10^{-2}
	$\lambda > 0$	0.31×10^{-2}	0.32×10^{-2}	0.30×10^{-2}	0.28×10^{-2}
GPOD	$\lambda = 0$	2.35×10^{-2}	2.49×10^{-2}	3.38×10^{-2}	17.38×10^{-2}
	$\lambda > 0$	2.12×10^{-2}	2.33×10^{-2}	3.15×10^{-2}	16.38×10^{-2}

SCVAE error is ten times lower than the GPOD-error, and this difference becomes larger with fewer measurements. Note that adding regularization did not have much effect on the relative error. From the motivating example, we observed that regularizing with $\lambda > 0$ is better in terms of a more consistent and low variable error estimation. Here, we selected from the ten trained models the one that performed best on the validation dataset. This model selection approach shows that there are no significant differences between the two regularization techniques. The associated errors in the divergence of the velocity fields are reported in Table II.

B. Current data from Bergen ocean model

We tested the SCVAE on simulations from the Bergen Ocean Model (BOM).[68] The BOM is a three-dimensional terrain-following nonhydrostatic ocean circulation model with capabilities of resolving mesoscale to large-scale processes. Here, we use velocities simulated by Ali *et al.*[43] The simulations were conducted on the entire North Sea with 800 m horizontal and vertical grid resolution and 41 layers for the period from 1 January 2012 to 15 January 2012. Forcing of the model consists of wind, atmospheric pressure, harmonic tides, rivers, and initial fields for salinity and temperature. For details of the setup of the model, forcing, and the simulations, we refer to Ref. 43.

Here, the horizontal velocities in the bottom layer of an excerpt of the model domain, with dimensions 25.6 × 25.6 km^2 in the southern North Sea [center at (58.36°N, 1.91°E)] are used as the dataset for reconstruction. In Fig. 9, we have plotted time series of the mean and extreme values of the two velocity components, u and v, for each time t in $\mathcal{T}$.

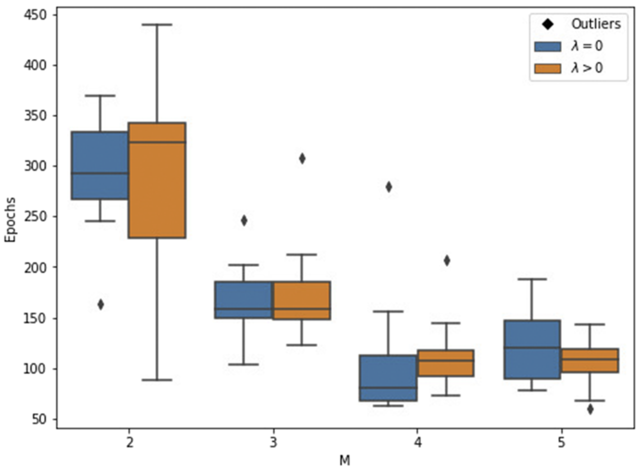

FIG. 8. Number of epochs trained depending on the number of measurements.

TABLE II. Comparison of the divergence error $\mathcal{E}_{div}$ as calculated in Eq. (22) for the different methods and regularization techniques. The true divergence error on the entire test dataset is 0.1058.

		Measurement locations			
Method	Regularization	5	4	3	2
SCVAE	$\lambda = 0$	0.1439	0.1580	0.1383	0.143 2
	$\lambda > 0$	0.1533	0.1408	0.1468	0.141 0
GPOD	$\lambda = 0$	0.1052	0.1047	0.0943	0.088 66
	$\lambda > 0$	0.1039	0.1051	0.0966	0.066 9

1. Preprocessing

We extract the 32 × 32 central grid from the bottom layer velocity data. Hence, $\mathcal{P}$ contains $N = 1024$ points. The temporal resolution is originally 105 000, and the time between each time step is 1 min. We downsample the temporal dimension of the original data uniformly such that the number of time steps in $\mathcal{T}$ is $K = 8500$. We train and validate the SCVAE with two different data splits: randomized and sequential in time. For the sequential split, we have used the last 15% for the test, the last 30% of the remaining data is used for validation, and the fist 70% for training. In Fig. 9, the red and blue vertical lines indicate the data split for this case. For the random split, the instances $\boldsymbol{x}^{(i)}$ are drawn randomly from $\boldsymbol{X}$ with the same percentage. The data were scaled as described in Appendix B. The input $\boldsymbol{x}^{(i)}$ to the SCVAE was shaped as (32 × 32 × 2) in order to apply convolutional layers. We use nine, five, and three fixed spatial measurement locations. In particular, the subgrid $\mathcal{Q}$ is given as

$$\begin{aligned} Q_9 &= \{(6,6),(6,17),(6,27),(17,17),(17,27),\\ &\qquad (17,6),(27,6),(27,17),(27,27)\},\\ Q_5 &= \{(6,6),(17,17),(27,27),(6,27),(27,6)\},\\ Q_3 &= \{(6,27),(17,17),(27,6)\}. \end{aligned} \tag{24}$$

As before, the values of u and v at these specific locations constitute the measurements $\boldsymbol{m}^{(i)} \in \boldsymbol{M}$.

2. Model

A schematic description of the model is given in Appendixes C 3 and C 4. The first two layers of the encoder are convolutional layers with 64 and 128 filters with strides and a kernel size of 2 and ReLu activation functions. This compresses the data into a shape of (8 × 8 × 128). The next layers are flattening and dense layers, where the latter have 16 filters and ReLu activation. The subsequent layers define the mean and log-variance of the latent representation $\boldsymbol{z}$, which is input to a lambda layer for realization of the reparameterization trick. The encoder outputs the samples $\boldsymbol{z}^{(i)}$, the mean, and the log-variance of $\boldsymbol{z}^{(i)}$.

The input to the decoder is the output $\boldsymbol{z}^{(i)}$ of the encoder and the measurement $\boldsymbol{m}^{(i)}$. To concatenate the inputs, $\boldsymbol{m}^{(i)}$ is flattened. After concatenation of $\boldsymbol{z}^{(i)}$ and $\boldsymbol{m}^{(i)}$, the next layer is a dense layer with shape (8 × 8 × 128) and ReLu activation. This allows for the use of transposed convolutional layers to obtain the original shape of the data. Hence, the following layers are two transposed convolutional layers with 64 and 128 filters, strides and kernel size of 2, and ReLu activation. The final layer is a transposed convolutional with linear activation functions and a filter size of shape (32 × 32 × 2), i.e., the same shape as $\boldsymbol{x}^{(i)}$.

3. Results

We illustrate the obtained posterior predictive distribution in terms of the predictive mean and standard deviation for the prediction at a specific time. The SCVAE is compared with the GPOD method, both with $\lambda > 0$ and $\lambda = 0$, for measurement locations given in Eq. (24) for random and sequential split cases. To generate the posterior predictive distributions [Eq. (17)], we sample 200 realizations from $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, which allows for calculation mean prediction and uncertainty estimates [see Eq. (18)]. Figure 10 shows the results of the prediction at time step 1185 for both the u and v components and associated uncertainty estimates for a trained model with $\lambda = 0$ and Q_3 measurement locations [see Eq. (24)].

In Fig. 11, we plot the true solution and the predicted mean velocity [Eq. (18)] with the associated uncertainty [Eq. (20)] for two grid points. We plot only the first 600 time steps for readability. The first grid point is (26, 6) and (4, 1). One location is ~5.1 km from the nearest observation, and another one is about 16.1 km away.

Figure 12 has to be viewed in context with Fig. 11 and shows the difference between the true and the predicted solutions with the associated difference marginal in time for the two locations as in Fig. 11.

Integrating over the latent space generates a posterior distribution of the reconstruction, as described in Sec. III B 1. It is also possible to use the latent space to generate new statistically sound versions of u and v. This is presented in Fig. 13 where it is sampled uniformly over the two dimensional latent space $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, and the result shows how different variations can be created with the SCVAE model, given only the sparse measurements.

These sampled velocities could be used for ensemble simulations when estimating uncertainty in a passive tracer transport (see Ref. 48).

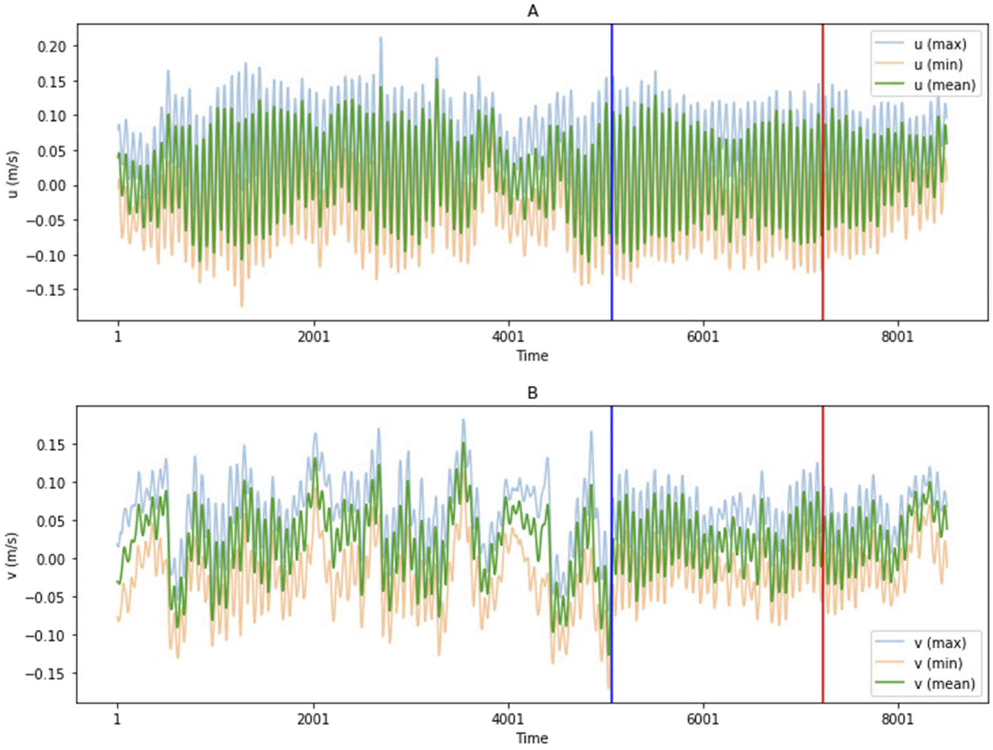

FIG. 9. Mean velocities u (a) and v (b) for each time t in $\mathcal{T}$ and associated extremes for each instance. The horizontal lines indicate the sequential data split. Units on the x-axis are in minutes.

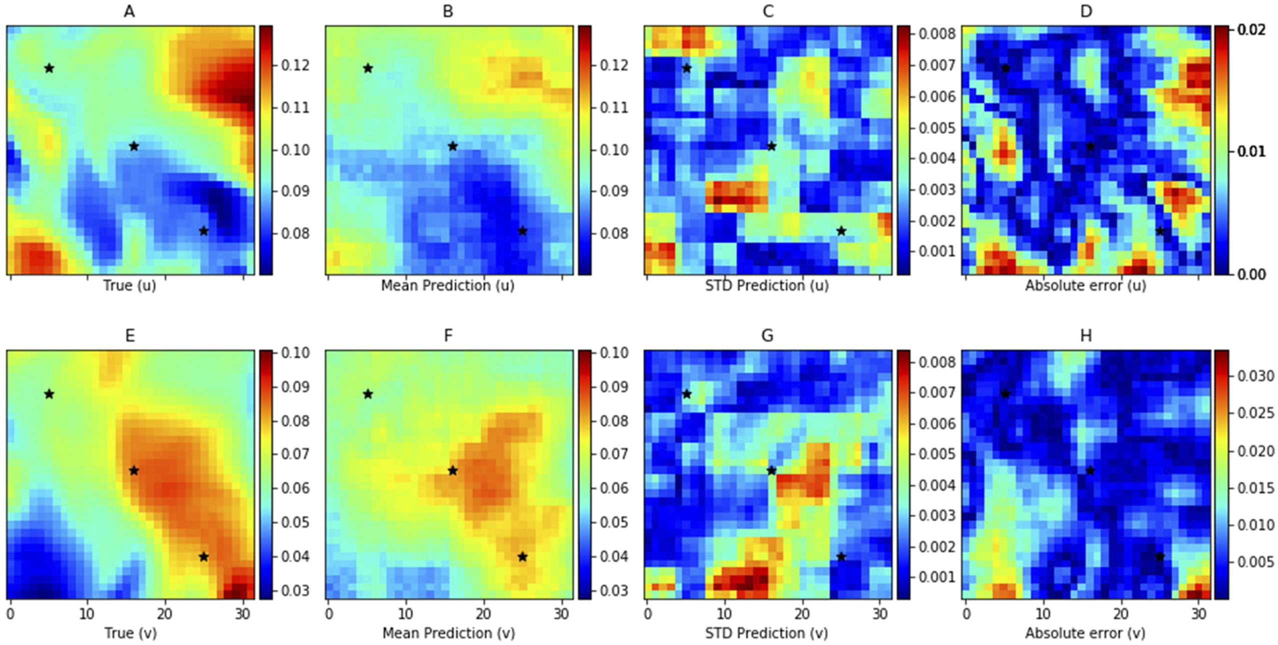

FIG. 10. (a) and (b) represent u-velocities and (e)–(h) represent v-velocities and associated statistical measures, respectively. The results are based on a model trained with $\lambda = 0$ and Q_3 measurement locations. [(a) and (e)] True solutions. [(b) and (f)] Reconstructed solutions. [(c) and (g)] Standard deviations. [(d) and (h)] Absolute errors.

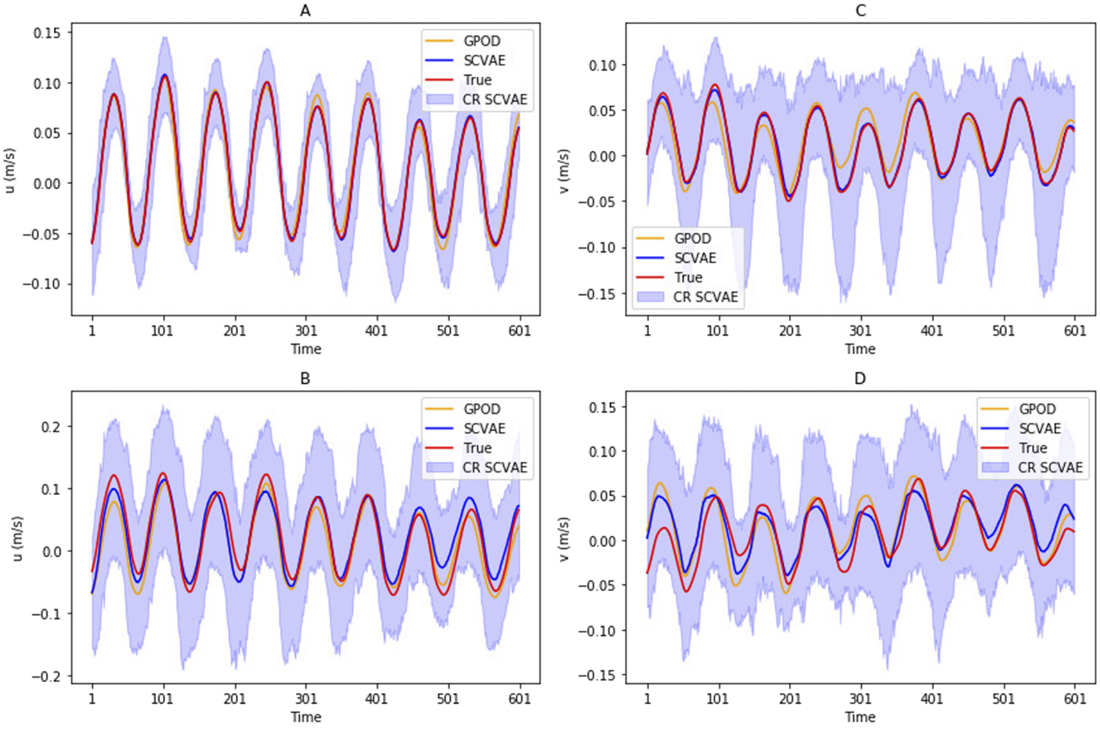

FIG. 11. Velocities u [(a) and (c)] and v [(b) and (d)] at grid points (26, 6) (5.1 km from the nearest observation) and (4, 1) (16.1 km from the nearest observation) with the associated confidence regions, respectively. The estimates are based on a model trained with $\lambda = 0$ and Q_3 measurement locations.

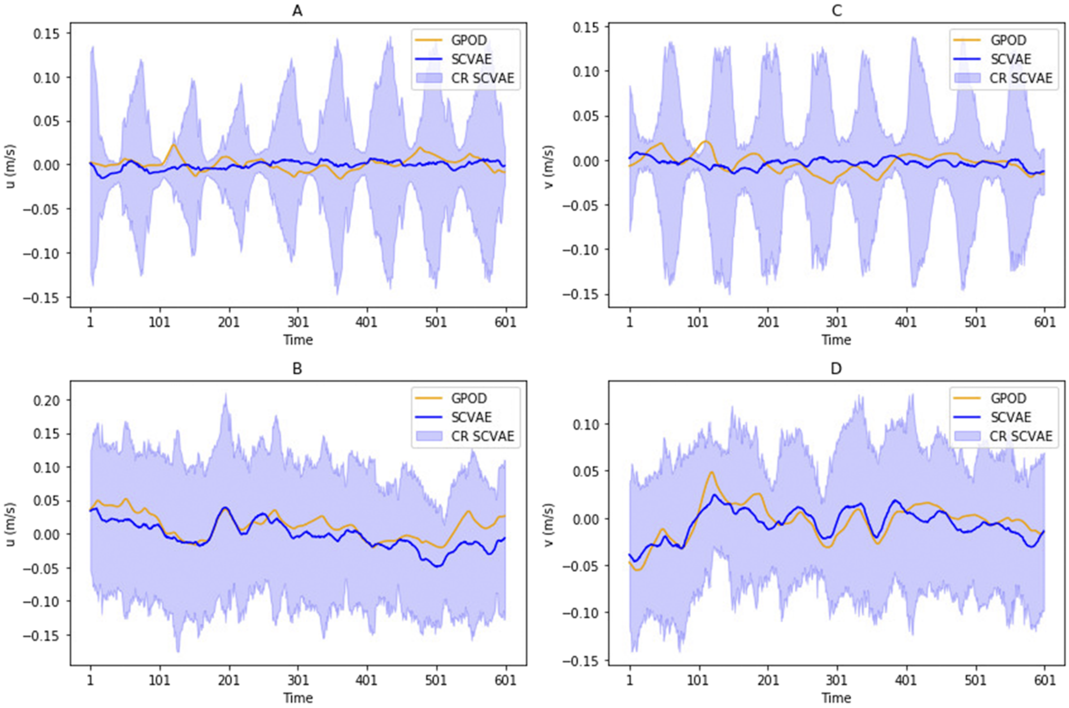

FIG. 12. The difference between the true and predicted u [(a) and (c)] and v [(b) and (d)] at grid points (6, 31) and (101, 25) with the associated difference marginals, respectively. The estimates are based on a model trained with $\lambda = 0$ and Q_3 measurement locations.

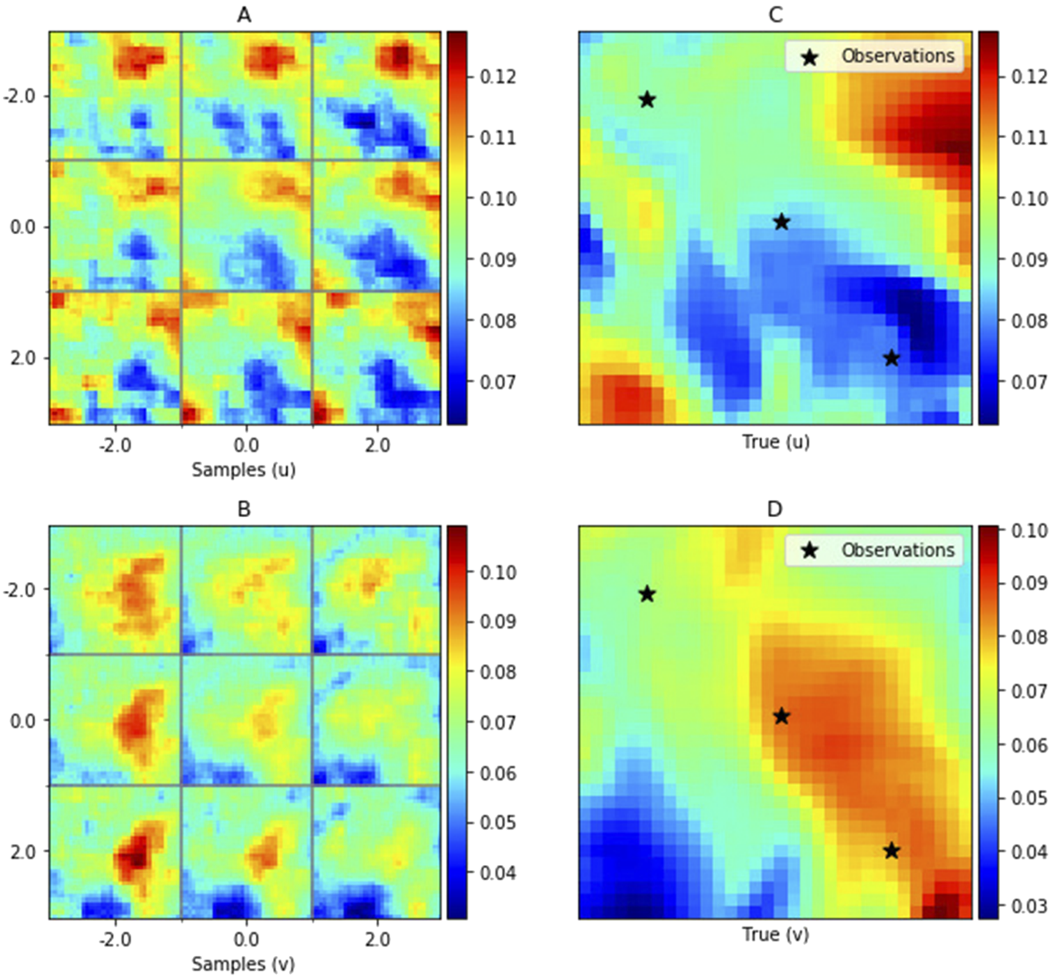

FIG. 13. Predictions with uniformly sampling over the latent representation u (a) and v (b) for a sample number of 1185 with the associated true solutions in (c) and (d), respectively. The predictions are generated from a model with $\lambda = 0$ and Q_3 measurement locations.

TABLE III. Errors as calculated in Eq. (21) for the different methods, regularization techniques ($\lambda = 0$ or $\lambda > 0$), split regimes, and measurements.

			Measurement locations		
Split	Regularization	Method	9	5	3
Random	$\lambda = 0$	SCVAE	0.1379	0.2097	0.2928
		GPOD	0.3300	0.3822	0.4349
	$\lambda > 0$	SCVAE	0.1403	0.2025	0.3016
		GPOD	0.2971	0.3579	0.4039
Time dependent	$\lambda = 0$	SCVAE	0.3493	0.3913	0.4155
		GPOD	0.3767	0.4031	0.4678
	$\lambda > 0$	SCVAE	0.3527	0.3889	0.4141
		GPOD	0.3362	0.3695	0.4462

TABLE IV. Divergence errors as calculated in Eq. (22) for the different methods, regularization techniques ($\lambda = 0$ or $\lambda > 0$), split regimes, and measurements. The true divergence of the test data is of order 10^{-4}.

			Measurement locations		
Split	Regularization	Method	9	5	3
Random	$\lambda = 0$	SCVAE	3.75×10^{-5}	3.62×10^{-5}	3.42×10^{-5}
		GPOD	6.51×10^{-5}	5.88×10^{-5}	5.02×10^{-5}
	$\lambda > 0$	SCVAE	3.60×10^{-5}	3.60×10^{-5}	3.13×10^{-5}
		GPOD	6.23×10^{-5}	4.77×10^{-5}	4.14×10^{-5}
Time dependent	$\lambda = 0$	SCVAE	2.02×10^{-5}	1.80×10^{-5}	1.69×10^{-5}
		GPOD	5.09×10^{-5}	4.03×10^{-5}	4.15×10^{-5}
	$\lambda > 0$	SCVAE	2.05×10^{-5}	1.99×10^{-5}	1.85×10^{-5}
		GPOD	4.39×10^{-5}	3.65×10^{-5}	2.92×10^{-5}

The SCVAE results are compared with results of the GPOD method (see Tables III and IV). The tables show the errors as calculated in Eqs. (21) and (22) of the test dataset for both sequential and random split. For the sequential splitting, the SCVAE is better for three measurement locations, while the GPOD method performs better for nine and five locations. From Fig. 9, we observe that test dataset seems to arise from a different process than the train and validation data (especially for v). Thus, the SCVAE generalize worse than a simpler model such as the GPOD.[75] For the three location case, the number of components in the GPOD is not enough to compete with the SCVAE.

With random split on the train, test, and validation data, we see that the SCVAE is significantly better than the GPOD. The training data and measurements represent the test data and test measurements better with random splitting. This highlights the importance of large datasets that cover as many outcomes as possible. Demanding that $\lambda > 0$ in Eq. (16) does not improve the result. From the SCVAE models with $\lambda = 0$, we learn that the reconstructed representations should have low divergence without explicitly demanding it during optimization. However, as discussed in the 2D flow around cylinder experiment, demanding $\lambda > 0$ seems to improve the conditioning of the optimization problem and give more consistent results. In Fig. 14, we present a boxplot of the number of epochs against the number of measurements. For each measurement configuration and regularization technique, the model is optimized ten times. The variation in the number of epochs for each measurement and regularization technique is due to different priors of the weights and mini-batch sampling.

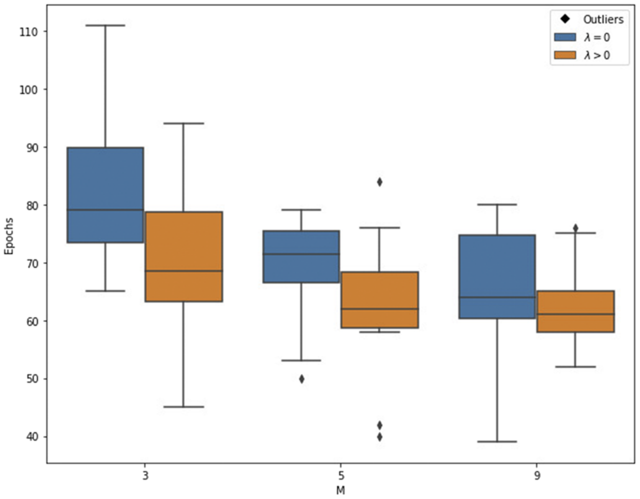

FIG. 14. The figure shows the number of epochs and the number of measurement locations.

V. DISCUSSION

We have presented the SCVAE method for efficient data reconstruction based on sparse observations. The derived objective functions for the network optimization show that the encoding is independent of measurements. This allows for a simpler model structure with fewer model parameters than a CVAE and results in an optimization procedure that requires less computational resources.

We have shown that the SCVAE is suitable for the reconstruction of fluid flow. The method is showcased on two different datasets: velocity data from simulations of 2D flow around a cylinder and bottom currents from the BOM. The fact that the fluids studied in the experiments are incompressible served as a motivation for adding an extra constraint to the objective function with $\lambda > 0$.

Our investigation of additional regularization showed that the mean reconstruction errors over all models were lower with $\lambda > 0$ compared to the model where $\lambda = 0$, but the best reconstruction errors were similar for $\lambda = 0$ and $\lambda > 0$.

The SCVAE is a probabilistic model, which allows us to make predictions, estimate their uncertainty, and draw multiple samples from the predictive distribution. The last two properties make the SCVAE a useful method especially when the predictions are used as a component in a larger application, i.e., ensemble simulations of tracer transport. Motivated by Ref. 17, we compared the SCVAE predictions with the predictions of a modified GPOD method.

Unlike the GPOD method, a benefit with the SCVAE method is that it scales well to larger datasets. Another aspect and as suggested by the experiments in Sec. IV, the GPOD seems more sensitive to the number of measurement locations than the SCVAE. On the other hand, the experiments suggested that GPOD is better than SCVAE with a larger number of measurement locations if the training data and the test data are too different. Essentially, the SCVAE is overfit to the training data and, as a result, performs poorly on the test dataset. This fact shows the importance of training the SCVAE on large datasets, which covers as many potential flow patterns as possible. Furthermore, the results show that the GPOD is more sensitive to the measurement location choice than the SCVAE, and the GPOD method is not expected to perform well on a complex flow with very few fixed measurement locations.

VAEs have been used for generating data in computer vision,[31] and autoencoders are natural choices in reconstruction tasks.[13] Many reconstruction approaches, including the GPOD approach, first create a basis and then use the basis and minimize the error of the observations.[50,76] This makes the GPOD suitable for fast

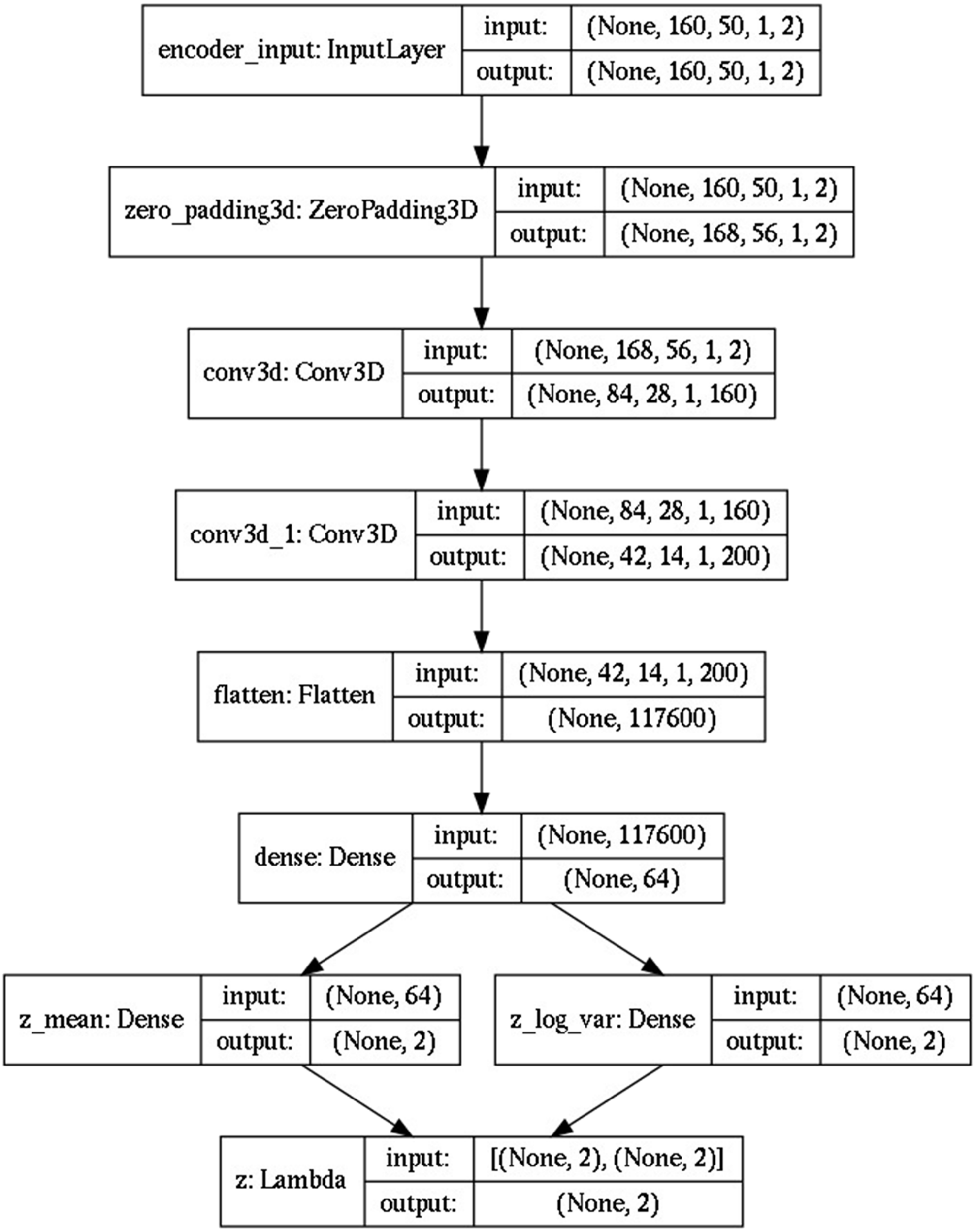
encoder_input: InputLayer
input: (None, 160, 50, 1, 2)
output: (None, 160, 50, 1, 2)
zero_padding3d: ZeroPadding3D
input: (None, 160, 50, 1, 2)
output: (None, 168, 56, 1, 2)
conv3d: Conv3D
input: (None, 168, 56, 1, 2)
output: (None, 84, 28, 1, 160)
conv3d_1: Conv3D
input: (None, 84, 28, 1, 160)
output: (None, 42, 14, 1, 200)
flatten: Flatten
input: (None, 42, 14, 1, 200)
output: (None, 117600)
dense: Dense
input: (None, 117600)
output: (None, 64)
z_mean: Dense
input: (None, 64)
output: (None, 2)
z_log_var: Dense
input: (None, 64)
output: (None, 2)
z: Lambda
input: [(None, 2), (None, 2)]
output: (None, 2)

FIG. 15. Schematic overview and details of the encoder used in the CW-data experiment.

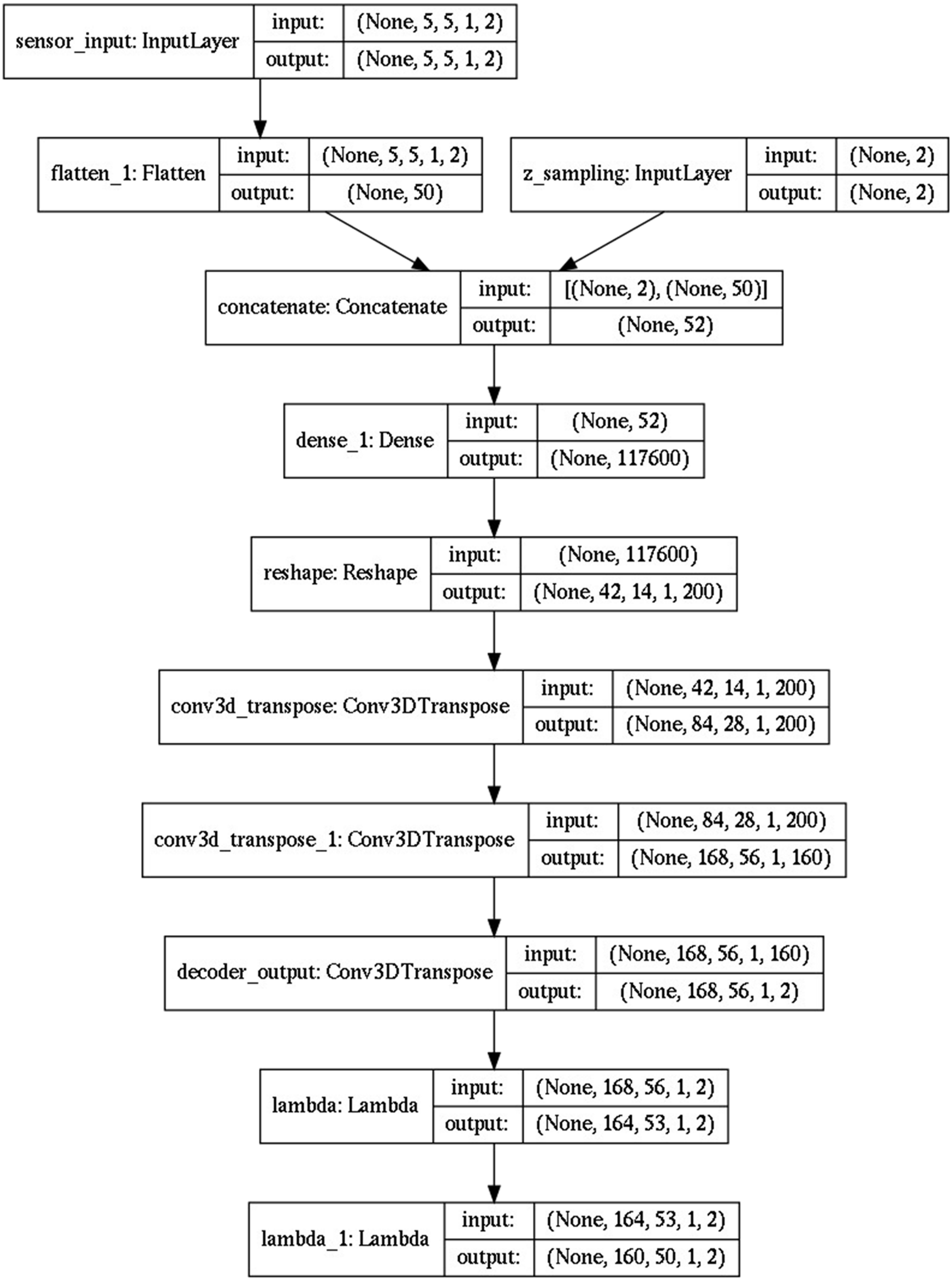
sensor_input: InputLayer
input: (None, 5, 5, 1, 2)
output: (None, 5, 5, 1, 2)
flatten_1: Flatten
input: (None, 5, 5, 1, 2)
output: (None, 50)
z_sampling: InputLayer
input: (None, 2)
output: (None, 2)
concatenate: Concatenate
input: [(None, 2), (None, 50)]
output: (None, 52)
dense_1: Dense
input: (None, 52)
output: (None, 117600)
reshape: Reshape
input: (None, 117600)
output: (None, 42, 14, 1, 200)
conv3d_transpose: Conv3DTranspose
input: (None, 42, 14, 1, 200)
output: (None, 84, 28, 1, 200)
conv3d_transpose_1: Conv3DTranspose
input: (None, 84, 28, 1, 200)
output: (None, 168, 56, 1, 160)
decoder_output: Conv3DTranspose
input: (None, 168, 56, 1, 160)
output: (None, 168, 56, 1, 2)
lambda: Lambda
input: (None, 168, 56, 1, 2)
output: (None, 164, 53, 1, 2)
lambda_1: Lambda
input: (None, 164, 53, 1, 2)
output: (None, 160, 50, 1, 2)

FIG. 16. Schematic overview and details of the decoder used in the CW-data experiment.

optimization of measurement locations that minimize the reconstruction error. On the other hand, the SCVAE optimizes the basis function given the measurements, i.e., they are known and fixed. This makes it challenging to use the framework for optimizing sensor layout. However, if the measurement locations are fixed and large amounts of training data are available, the SCVAE outperforms the GPOD for reconstruction. SCVAE optimizes the latent representation and the neural network model parameters and variational and generative parameters, given the measurements. This ensures that the reconstruction is adapted to the specific configuration of measurements.

A limitation of our experiments is that we used only 100 and 200 samples and constructed the confidence region under further simplifying assumptions. The uncertainty estimate could be improved by increasing the sample size and better model for the confidence region.

Natural applications for the SCVAE are related to environmental data, where we often have sparse measurements. It is possible to optimize the sensor layout to the best possible one for detecting unintentional discharges in the marine environment by using a simple transport model, forced by given flow fields, to predict the area of influence (see the work of Oleynik *et al*).[48] The SCVAE can be used to improve such an approach by efficiently generating realistic flow fields in a Monte Carlo framework. The incompressibility constraint is important in this case, and it assures conservation of mass in the transport model. Such a framework may be important as the input to design, environmental risk assessments, and emergency preparedness plans.

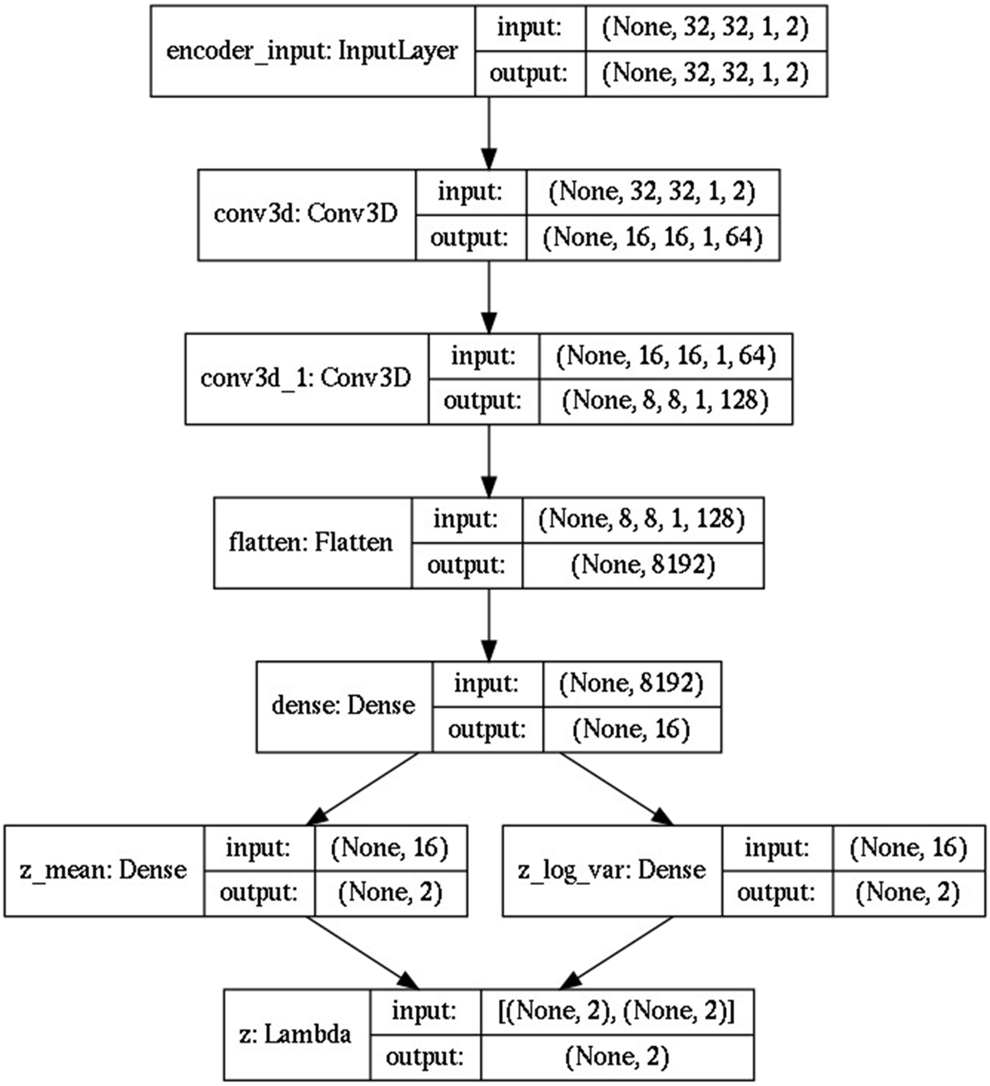
encoder_input: InputLayer
input: (None, 32, 32, 1, 2)
output: (None, 32, 32, 1, 2)
conv3d: Conv3D
input: (None, 32, 32, 1, 2)
output: (None, 16, 16, 1, 64)
conv3d_1: Conv3D
input: (None, 16, 16, 1, 64)
output: (None, 8, 8, 1, 128)
flatten: Flatten
input: (None, 8, 8, 1, 128)
output: (None, 8192)
dense: Dense
input: (None, 8192)
output: (None, 16)
z_mean: Dense
input: (None, 16)
output: (None, 2)
z_log_var: Dense
input: (None, 16)
output: (None, 2)
z: Lambda
input: [(None, 2), (None, 2)]
output: (None, 2)

FIG. 17. Schematic overview and details of the encoder used in the BOM-data experiment.

We have highlighted the SCVAE through the reconstruction of currents and flow field reconstruction; however, the SCVAE method is not limited to fluid flow problems. For instance, the same principles could be used in computer vision to generate a new picture based on sparse pixel representations or in time series reconstruction.

A survey paper on Bayesian networks that accounts for time dependence in the model itself, the so-called dynamical BNNs, was

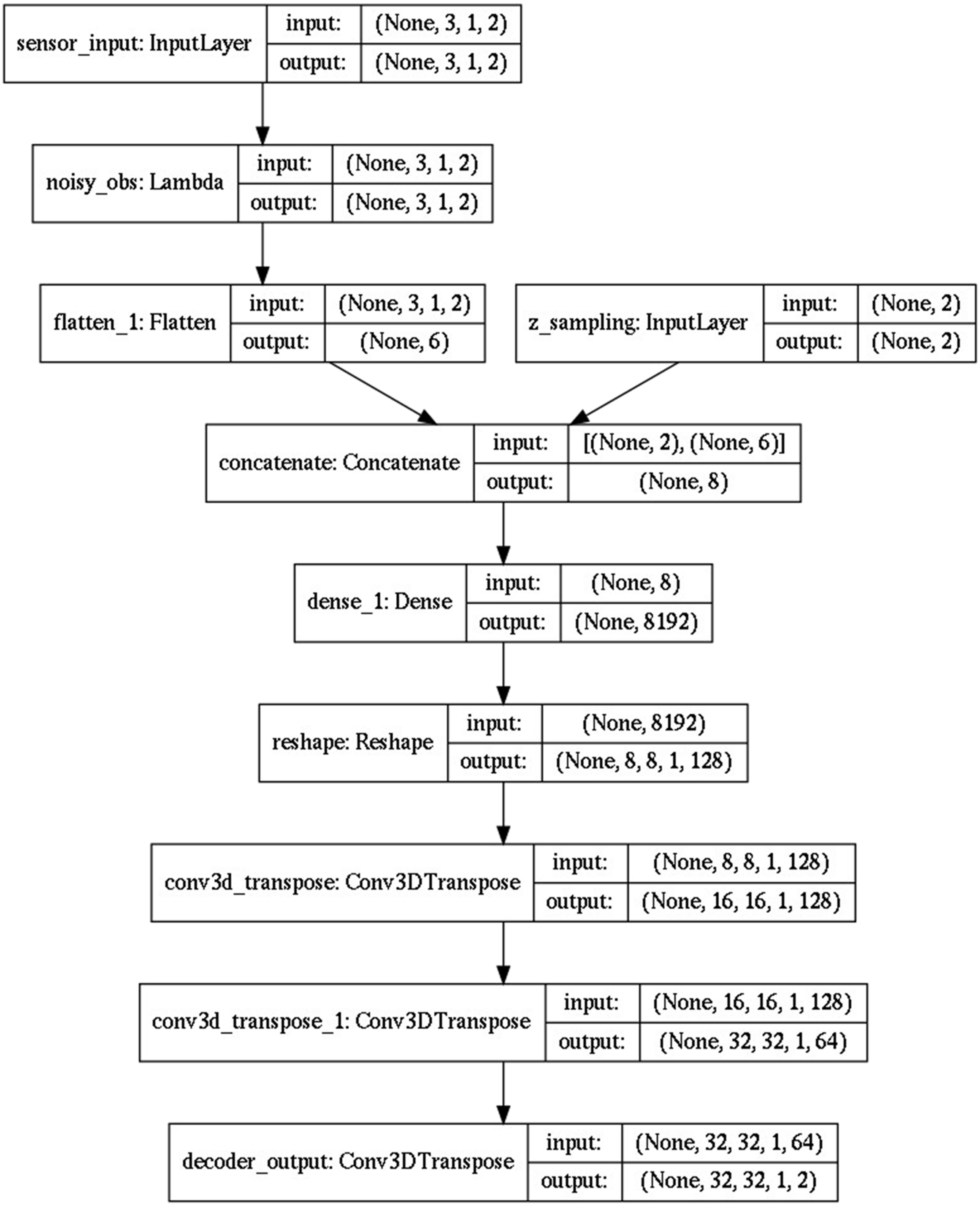

FIG. 18. Schematic overview and details of the decoder used in the BOM-data experiment.

recently published.[77] A variety of recurrent neural networks are presented, applicable for spatial and temporal data without explicitly addressing the flow reconstruction problem. A natural extension of the SCVAE can be to implement it as a dynamical BNN, i.e., to predict the current state $p_\theta(\boldsymbol{x}_t|\boldsymbol{m}_t, \boldsymbol{x}_{t-1})$, given the measurements and the reconstruction from the previous time step. This could potentially improve the reconstruction further.

ACKNOWLEDGMENTS

This work is part of the project ACTOM, funded through the ACT program (Accelerating CCS Technologies, Horizon2020 Project No. 294766). This work received financial contributions from the Research Council of Norway (RCN), Norway; the Netherlands Enterprise Agency (RVO), the Netherlands; the Department for Business, Energy and Industrial Strategy (BEIS) together with extra funding from NERC and EPSRC research councils, United Kingdom; and the U.S. Department of Energy (U.S. DOE), USA. Kristian Gundersen was supported by the Research Council of Norway through the CLIMIT program (Project No. 254711, BayMode) and the European Union Horizon 2020 research and innovation program under Grant Agreement No. 654462, STEMM-CCS. The authors acknowledge NVIDIA Corporation for providing their GPUs in the academic GPU Grant Program.

APPENDIX A: GPOD METHOD WITH DIVERGENCE REGULARIZATION

Let the vectors $\boldsymbol{x}^{(i)}$ in $\boldsymbol{X}$ be organized as a snapshots matrix $\boldsymbol{X}_h \in \mathbb{R}^{2N\times K}$. Here, we consider that the latent space be given by the r principal components of the matrix $\boldsymbol{X}_h$, assuming that $r \ll N$. Thus, $\boldsymbol{X}_h \approx \Phi A_h$, where Φ is the $2N \times r$ matrix of principal components and $A_h = \Phi^{+} X_h$ is the $r \times K$ representation of X_h.

Let $\boldsymbol{x}^* \in \mathbb{R}^{2N}$ be an unknown state that we need to reconstruct from M spatial measurements $\boldsymbol{m}^* = C\boldsymbol{x}^*$. We assume that there is an $\boldsymbol{a}$ such that $\boldsymbol{x}^* = \boldsymbol{\Phi a}$, and we search for a solution of $\boldsymbol{C\Phi a} = \boldsymbol{m}^*$. Even if the system $\boldsymbol{C\Phi a} = \boldsymbol{m}^*$ is overdetermined, the matrix $C\Phi$ could be ill-conditioned or rank-deficient. However, since the number of sensors is usually small, $2M < r$, the system is underdetermined and regularization is required.

Assuming that the flow is incompressible, the natural regularization is to penalize the divergence error of a solution. That is, we solve

$$\boldsymbol{a}^* = argmin_{\boldsymbol{a}} \|\boldsymbol{C\Phi a} - \boldsymbol{m}^*\|_2^2 + \lambda \|L_{div}\boldsymbol{\Phi a}\|_2^2, \tag{A1}$$

where $L_{div} : \mathbb{R}^{2N} \to \mathbb{R}^N$ is a linear operator approximating the divergence and $\lambda > 0$ is a regularization constant. Finally, we decode $\boldsymbol{x}^*$ from the measurements $\boldsymbol{m}^*$ as $\boldsymbol{x}^* = \boldsymbol{\Phi a}^*$.

APPENDIX B: SCALING OF DATA

Let $\mathcal{T}_{train}$ contain the times t_{l_i}, $i = 1, \ldots, n$, corresponding to the training data. We define

$$u_{\max} = \max_{p,t} u(p,t) \quad \text{and} \quad u_{\min} = \min_{p,t} u(p,t)$$

and

$$v_{\max} = \max_{p,t} v(p,t) \quad \text{and} \quad v_{\min} = \min_{p,t} v(p,t)$$

as the largest and smallest values of u and v on $\mathcal{P}$ and $\mathcal{T}_{train}$. Then, the middle points are given as

$$u_c = \frac{u_{\max} + u_{\min}}{2}, \quad v_c = \frac{v_{\max} + v_{\min}}{2},$$

and the half lengths are given as

$$d_u = \frac{u_{\max} - u_{\min}}{2}, \quad d_v = \frac{v_{\max} - v_{\min}}{2}.$$

Then, the whole data are scaled as

$$\tilde{u} = \frac{u - u_c}{d_u}, \quad \tilde{v} = \frac{v - v_c}{d_v},$$

and the divergence operator L_{div} is scaled accordingly.

After the optimization is completed, the data are scaled back, i.e.,

$$u = d_u\tilde{u} + u_c, \quad v = d_v\tilde{v} + v_c.$$

The relative errors in Eq. (21) are calculated on the scaled data. The divergence errors [Eq. (22)] are unaffected by the scaling.

APPENDIX C: DETAILS ON THE EXPERIMENTS

We use Keras[79] in the implementation of the SCVAE for all experiments. Here, we present details on the architecture of the decoders and encoders for the different experiments (Figs. 15–18). We have optimized the SCVAE models with different number of measurements. That is, the shape of the input layer to the decoder will be dependent on the measurements (sensor-input layer). Here, we present details on the architecture of the encoders and decoders with the largest number of measurements for SCVAE models for both experiments. There is one extra dimension in the figures showing the encoders and decoders. Here, this dimension is one, but the framework is implemented to allow for more dimensions in time.

1. Encoder for 2D flow around cylinder data experiment

2. Decoder for 2D flow around cylinder experiment

3. Encoder for BOM data experiment

4. Decoder for BOM data experiment

DATA AVAILABILITY

The 2D flow around a cylinder dataset is simulated by Weinkauf[54] using the Free Software *Gerris Flow Solver*.[55] The data that support the findings of this study are openly available in http://tinoweinkauf.net/notes/cylinder2d.html.[53] The BOM dataset is simulated by Alfatih Ali and contains time series of CO_2 concentration, velocity components, time, and position in longitude and latitude. The data that support the findings of this study are openly available in Zenodo at http://doi.org/10.5281/zenodo.806088.[78]

REFERENCES

[1] S. L. Brunton and B. R. Noack, "Closed-loop turbulence control: Progress and challenges," Appl. Mech. Rev. **67**(5), 050801 (2015).

[2] L. Kong, W. Wei, and Q. Yan, "Application of flow field decomposition and reconstruction in studying and modeling the characteristics of a cartridge valve," Eng. Appl. Comput. Fluid Mech. **12**(1), 385–396 (2018).

[3] T. Bolton and L. Zanna, "Applications of deep learning to ocean data inference and subgrid parameterization," J. Adv. Model. Earth Syst. **11**(1), 376–399 (2019).

[4] D. Venturi and G. E. Karniadakis, "Gappy data and reconstruction procedures for flow past a cylinder," J. Fluid Mech. **519**, 315 (2004).

[5] J. L. Callaham, K. Maeda, and S. L. Brunton, "Robust flow reconstruction from limited measurements via sparse representation," Phys. Rev. Fluids **4**(10), 103907 (2019).

[6] K. Manohar, B. W. Brunton, J. N. Kutz, and S. L. Brunton, "Data-driven sparse sensor placement for reconstruction: Demonstrating the benefits of exploiting known patterns," IEEE Control Syst. Mag. **38**(3), 63–86 (2018).

[7] K. Yeo, "Data-driven reconstruction of nonlinear dynamics from sparse observation," J. Comput. Phys. **395**, 671–689 (2019).

[8] P. D. Oikonomou, A. H. Alzraiee, C. A. Karavitis, and R. M. Waskom, "A novel framework for filling data gaps in groundwater level observations," Adv. Water Resour. **119**, 111–124 (2018).

[9] L. Sirovich, "Turbulence and the dynamics of coherent structures. I. Coherent structures," Q. Appl. Math. **45**(3), 561–571 (1987).

[10] R. Everson and L. Sirovich, "Karhunen–Loève procedure for gappy data," J. Opt. Soc. Am. A **12**(8), 1657–1664 (1995).

[11] D. L. Donoho, "Compressed sensing," IEEE Trans. Inf. Theory **52**(4), 1289–1306 (2006).

[12] P. J. Schmid, "Dynamic mode decomposition of numerical and experimental data," J. Fluid Mech. **656**, 5–28 (2010).

[13] S. M. A. Al Mamun, C. Lu, and B. Jayaraman, "Extreme learning machines as encoders for sparse reconstruction," Fluids **3**(4), 88 (2018).

[14] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," J. Comput. Phys. **378**, 686–707 (2019).

[15] I. Bright, G. Lin, and J. N. Kutz, "Compressive sensing based machine learning strategy for characterizing the flow around a cylinder with limited pressure measurements," Phys. Fluids **25**(12), 127102 (2013).

[16] K. Cohen, S. Siegel, and T. McLaughlin, "Sensor placement based on proper orthogonal decomposition modeling of a cylinder wake," in *33rd AIAA Fluid Dynamics Conference and Exhibit* (Aerospace Research Central, 2003), p. 4259.

[17] B. Yildirim, C. Chryssostomidis, and G. E. Karniadakis, "Efficient sensor placement for ocean measurements using low-dimensional concepts," Ocean Modell. **27**(3-4), 160–173 (2009).

[18] S. Pawar and O. San, "Data assimilation empowered neural network parameterizations for subgrid processes in geophysical flows," arXiv:2006.08901 (2020).

[19] N. B. Erichson, L. Mathelin, Z. Yao, S. L. Brunton, M. W. Mahoney, and J. N. Kutz, "Shallow neural networks for fluid flow reconstruction with limited sensors," Proc. R. Soc. A **476**(2238), 20200097 (2020).

[20] J. M. Pérez, S. Le Clainche, and J. M. Vega, "Reconstruction of three-dimensional flow fields from two-dimensional data," J. Comput. Phys. **407**, 109239 (2020).

[21] S. Le Clainche and J. M. Vega, "Higher order dynamic mode decomposition to identify and extrapolate flow patterns," Phys. Fluids **29**(8), 084102 (2017).

[22] Y. Gao, L. Liu, C. Zhang, X. Wang, and H. Ma, "SI-AGAN: Spatial interpolation with attentional generative adversarial networks for environment monitoring," in *24th European Conference on Artificial Intelligence–ECAI* (IOS Press BV, 2020), pp. 1786–1794.

[23] R. Maulik, K. Fukami, N. Ramachandra, K. Fukagata, and K. Taira, "Probabilistic neural networks for fluid flow surrogate modeling and data recovery," Phys. Rev. Fluids **5**(10), 104401 (2020).

[24] O. Makansi, E. Ilg, O. Cicek, and T. Brox, "Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2019), pp. 7144–7153.

[25] M. Raissi, A. Yazdani, and G. E. Karniadakis, "Hidden fluid mechanics: A Navier-Stokes informed deep learning framework for assimilating flow visualization data," arXiv:1808.04327 (2018).

[26] A. Grover and S. Ermon, "Uncertainty autoencoders: Learning compressed representations via variational information maximization," in *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics* (PMLR, 2019), pp. 2514–2524.

[27] K. Fukami, T. Nakamura, and K. Fukagata, "Convolutional neural network based hierarchical autoencoder for nonlinear mode decomposition of fluid field data," Phys. Fluids **32**(9), 095110 (2020).

[28] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning Internal Representations by Error Propagation* (MIT Press, Cambridge, MA, USA, 1986), pp. 318–362.

[29] F. R. S. Karl Pearson, "LIII. On lines and planes of closest fit to systems of points in space," London, Edinburgh Dublin Philos. Mag. J. Sci. **2**(11), 559–572 (1901).

[30] H. Bourlard and Y. Kamp, "Auto-association by multilayer perceptrons and singular value decomposition," Biol. Cybern. **59**(4-5), 291–294 (1988).

[31] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," arXiv:1312.6114 (2013).

[32] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," in *Advances in Neural Information Processing Systems* (NIPS, 2015), pp. 3483–3491.

[33] D. J. C. MacKay, "A practical Bayesian framework for backpropagation networks," Neural Comput. **4**(3), 448–472 (1992).

[34] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," J. Mach. Learn. Res. **14**(1), 1303–1347 (2013).

[35] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," J. Am. Stat. Assoc. **112**(518), 859–877 (2017).

[36] B. S. Halpern, C. Longo, D. Hardy, K. L. McLeod, J. F. Samhouri, S. K. Katona, K. Kleisner, S. E. Lester, J. O'Leary, M. Ranelletti, A. A. Rosenberg, C. Scarborough, E. R. Selig, B. D. Best, D. R. Brumbaugh, F. S. Chapin, L. B. Crowder, K. L. Daly, S. C. Doney, C. Elfes, M. J. Fogarty, S. D. Gaines, K. I. Jacobsen, L. B. Karrer, H. M. Leslie, E. Neeley, D. Pauly, S. Polasky, B. Ris, K. St Martin, G. S. Stone, U. R. Sumaila, and D. Zeller, "An index to assess the health and benefits of the global ocean," Nature **488**(7413), 615–620 (2012).

[37] E. Domínguez-Tejo, G. Metternicht, E. Johnston, and L. Hedge, "Marine spatial planning advancing the ecosystem-based approach to coastal zone management: A review," Mar. Policy **72**, 115–130 (2016).

[38] H. Drange, G. Alendal, and O. M. Johannessen, "Ocean release of fossil fuel CO_2: A case study," Geophys. Res. Lett. **28**(13), 2637–2640, https://doi.org/10.1029/2000gl012609 (2001).

[39] S. F. Barstow, "The ecology of Langmuir circulation: A review," Mar. Environ. Res. **9**(4), 211–236 (1983).

[40] A. Ali, Ø. Thiem, and J. Berntsen, "Numerical modelling of organic waste dispersion from fjord located fish farms," Ocean Dyn. **61**(7), 977–989 (2011).

[41] K. L. Law, "Plastics in the marine environment," Annu. Rev. Mar. Sci. **9**(1), 205–229 (2017).

[42] K. Hylland, T. Burgeot, C. Martínez-Gómez, T. Lang, C. D. Robinson, J. Svavarsson, J. E. Thain, A. D. Vethaak, and M. J. Gubbins, "How can we quantify impacts of contaminants in marine ecosystems? The ICON project," Mar. Environ. Res. **124**, 2 (2015).

[43] A. Ali, H. G. Frøysa, H. Avlesen, and G. Alendal, "Simulating spatial and temporal varying CO_2 signals from sources at the seafloor to help designing risk-based monitoring programs," J. Geophys. Res.: Oceans **121**(1), 745–757, https://doi.org/10.1002/2015jc011198 (2016).

[44] J. Blackford, G. Alendal, H. Avlesen, A. Brereton, P. W. Cazenave, B. Chen, M. Dewar, J. Holt, and J. Phelps, "Impact and detectability of hypothetical CCS offshore seep scenarios as an aid to storage assurance and risk assessment," Int. J. Greenhouse Gas Control **95**, 102949 (2020).

[45] H. K. Hvidevold, G. Alendal, T. Johannessen, A. Ali, T. Mannseth, and H. Avlesen, "Layout of CCS monitoring infrastructure with highest probability of detecting a footprint of a CO_2 leak in a varying marine environment," Int. J. Greenhouse Gas Control **37**, 274–279 (2015).

[46] H. K. Hvidevold, G. Alendal, T. Johannessen, and A. Ali, "Survey strategies to quantify and optimize detecting probability of a CO_2 seep in a varying marine environment," Environ. Modell. Software **83**, 303–309 (2016).

[47] G. Alendal, "Cost efficient environmental survey paths for detecting continuous tracer discharges," J. Geophys. Res.: Oceans **122**(7), 5458–5467, https://doi.org/10.1002/2016jc012655 (2017).

[48] A. Oleynik, M. I. García-Ibáñez, N. Blaser, A. Omar, and G. Alendal, "Optimal sensors placement for detecting CO_2 discharges from unknown locations on the seafloor," Int. J. Greenhouse Gas Control **95**, 102951 (2020).

[49] K. Gundersen, G. Alendal, A. Oleynik, and N. Blaser, "Binary time series classification with Bayesian convolutional neural networks when monitoring for marine gas discharges," Algorithms **13**(6), 145 (2020).

[50] K. Willcox, "Unsteady flow sensing and estimation via the gappy proper orthogonal decomposition," Comput. Fluids **35**(2), 208–226 (2006).

[51] T. Jo, B. Koo, H. Kim, D. Lee, and J. Y. Yoon, "Effective sensor placement in a steam reformer using gappy proper orthogonal decomposition," Appl. Therm. Eng. **154**, 419–432 (2019).

[52] M. Mifsud, A. Vendl, L.-U. Hansen, and S. Görtz, "Fusing wind-tunnel measurements and CFD data using constrained gappy proper orthogonal decomposition," Aerosp. Sci. Technol. **86**, 312–326 (2019).

[53] T. Weinkauf, 2D flow around a cylinder dataset, www.csc.kth.se/~weinkauf/datasets/Cylinder2D.7z, 2010.

[54] T. Weinkauf and H. Theisel, "Streak lines as tangent curves of a derived vector field," IEEE Trans. Visualization Comput. Graphics **16**(6), 1225–1234 (2010).

[55] S. Popinet, "Free computational fluid dynamics," ClusterWorld **2**(6), 7 (2004).

[56] J. L. Proctor, S. L. Brunton, B. W. Brunton, and J. N. Kutz, "Exploiting sparsity and equation-free architectures in complex systems," Eur. Phys. J. Spec. Top. **223**, 2665–2684 (2014).

[57] D. Kingma and M. Welling, "An introduction to variational autoencoders," Found. Trends Mach. Learn. **12**, 307–392 (2019).

[58] K. Gregor, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra, "DRAW: A recurrent neural network for image generation," in *Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 7-9 July 2015*, Proceedings of Machine Learning Research Vol. 37, edited by F. Bach and D. Blei (PMLR, 2015), pp. 1462–1471.

[59] S. R. Bowman, L. Vilnis, O. Vinyals, A. Dai, R. Jozefowicz, and S. Bengio, "Generating sentences from a continuous space," in *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, Berlin, Germany, August 2016* (Association for Computational Linguistics, 2016), pp. 10–21.

[60] S. Kullback and R. A. Leibler, "On information and sufficiency," Ann. Math. Stat. **22**(1), 79–86 (1951).

[61] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. M. Botvinick, S. Mohamed, and A. Lerchner, "beta-VAE: Learning basic visual concepts with a constrained variational framework," in *International Conference on Learning Representations ICLR* (OpenReview.net/ICLR, 2017).

[62] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," in *Traces and Emergence of Nonlinear Programming* (Springer, 2014), pp. 247–258.

[63] W. Karush, "Minima of functions of several variables with inequalities as side constraints," M.Sc. dissertation (Department of Mathematics, University of Chicago, 1939).

[64] R. Caruana, "Multitask learning," Mach. Learn. **28**(1), 41–75 (1997).

[65] J. Baxter, "A Bayesian/information theoretic model of learning to learn via multiple task sampling," Mach. Learn. **28**(1), 7–39 (1997).

[66] J. Kiefer, J. Wolfowitz *et al.*, "Stochastic estimation of the maximum of a regression function," Ann. Math. Stat. **23**(3), 462–466 (1952).

[67] H. Robbins and S. Monro, "A stochastic approximation method," Ann. Math. Stat. **22**, 400–407 (1951).

[68] J. Berntsen, "Users guide for a modesplit σ-coordinate numerical ocean model," Department of Applied Mathematics, University of Bergen, Technical Report No. 135, 2000, p. 48.

[69] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proc. IEEE **86**(11), 2278–2324 (1998).

[70] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems* (OpenReviwe.net/ICLR, 2012), pp. 1097–1105.

[71] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proceedings of the IEEE International Conference on Computer Vision* (IEEE, 2015), pp. 1520–1528.

[72] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," Neural Comput. **1**(4), 541–551 (1989).

[73] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv:1412.6980 (2014).

[74] A. Ali Heydari, C. A. Thompson, and A. Mehmood, "SoftAdapt: Techniques for adaptive loss weighting of neural networks with multi-part loss functions," arXiv:1912.12355 (2019).

[75] K. Burnham and D. Anderson, *Model Selection and Multimodel Inference - A Practical Information-theoretic Approach* (Springer-Verlag, 2004).

[76] T. Bui-Thanh, M. Damodaran, and K. Willcox, "Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition," AIAA J. **42**(8), 1505–1516 (2004).

[77] L. Girin, S. Leglaive, X. Bie, J. Diard, T. Hueber, and X. Alameda-Pineda, "Dynamical variational autoencoders: A comprehensive review," arXiv:2008.12595 (2020).

[78] A. Ali, G. Alendal, and H. Avlesen (2017). "Modelled time series of CO_2 in the vicinity of a seep in the North Sea," Zenodo. https://doi.org/10.5281/zenodo.806088.

[79] F. Chollet *et al.*, Keras, https://keras.io, 2015.

[80] See https://www.oceandecade.org/ for more information about the project "United Nations Decade of Ocean Science for Sustainable Development (2021-2030)."

[81] The simulations are run from $t = 0$ to $t = 23$, but velocities are only extracted from $t = 15$ to $t = 23$.