IISE Transactions

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/uiie21

DETONATE: Nonlinear Dynamic Evolution Modeling of Time-dependent 3-dimensional Point Cloud Profiles

Michael Biehler, Daniel Lin & Jianjun Shi

To cite this article: Michael Biehler, Daniel Lin & Jianjun Shi (2024) DETONATE: Nonlinear Dynamic Evolution Modeling of Time-dependent 3-dimensional Point Cloud Profiles, IISE Transactions, 56:5, 541-558, DOI: 10.1080/24725854.2023.2207615

To link to this article: https://doi.org/10.1080/24725854.2023.2207615

IISE TRANSACTIONS
2024, VOL. 56, NO. 5, 541–558
https://doi.org/10.1080/24725854.2023.2207615

Taylor & Francis
Taylor & Francis Group

DETONATE: Nonlinear Dynamic Evolution Modeling of Time-dependent 3-dimensional Point Cloud Profiles

Michael Biehler[a], Daniel Lin[b], and Jianjun Shi[a]

[a]H. Milton School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, USA; [b]Walton High School, Marietta, USA

ABSTRACT
Modeling the evolution of a 3D profile over time as a function of heterogeneous input data and the previous time steps' 3D shape is a challenging, yet fundamental problem in many applications. We introduce a novel methodology for the nonlinear modeling of dynamically evolving 3D shape profiles. Our model integrates heterogeneous, multimodal inputs that may affect the evolvement of the 3D shape profiles. We leverage the forward and backward temporal dynamics to preserve the underlying temporal physical structures. Our approach is based on the Koopman operator theory for high-dimensional nonlinear dynamical systems. We leverage the theoretical Koopman framework to develop a deep learning-based framework for nonlinear, dynamic 3D modeling with consistent temporal dynamics. We evaluate our method on multiple high-dimensional and short-term dependent problems, and it achieves accurate estimates, while also being robust to noise.

ARTICLE HISTORY
Received 13 January 2023
Accepted 23 April 2023

KEYWORDS
Nonlinear dynamic evolution modeling; 3D profiles; Koopman operator theory; heterogenous data fusion; unstructured 3D data

1. Introduction

Spatiotemporal 3D shape profiles provide a flexible and rich geometric representation of objects in a physical world. In contrast with images, 3D point clouds can provide accurate shape and displacement measurements. However, most of the existing works focus on static 3D shape analysis, e.g., classification (Qi *et al.* 2017), regression (Biehler *et al.* 2022), segmentation (Landrieu and Simonovsky, 2018), or object detection (Zhou and Tuzel, 2018). Few works study the nonlinear evolution of 3D shape profiles. Additionally, the evolution of 3D shapes is impacted by a wide range of heterogeneous (i.e., multimodal) inputs. To the best of our knowledge, no existing work has modeled the evolution of 3D shapes as a function of multiple heterogeneous inputs and their previous 3D shape. There are many important applications of those data characteristics: In multistage manufacturing, the geometry of a part is affected by both the shape of the previous stage and the process conditions and settings at the current stage. Blast explosions and their evolution over time are affected by environmental conditions such as wind speed and initial energy. The evolution of landslides on mountain ranges over time is affected by their shape topology at a given time and the environmental conditions at a given time point. We have visualized those motivating applications in Figure 1, to allow readers to identify new application scenarios for the proposed methodology. The data characteristics of our framework can be summarized as follows:

- Dynamically, nonlinear, temporally evolving 3D shape profiles.
- 3D shape profiles are spatially affected by heterogeneous input data as well as temporally affected by their previous time steps' shape.
- 3D shapes are represented by unstructured 3D point clouds.

More formally, given point cloud frames $\mathcal{X}_S^1, \mathcal{X}_S^2, \ldots, \mathcal{X}_S^{t-1}$ and K heterogenous process inputs $\mathcal{X}_{h,\ j}^{t-1}, j = (1, \ldots, K)$, we are interested in predicting $\hat{\mathcal{X}}_S^t$, with no prior knowledge of the ground truth $\mathcal{X}_S^t$.

To tackle this challenging problem, we take inspiration from the existing work in physics and dynamic control on Koopman-based models (Durbin and Kooperman, 2012; Otto and Rowley, 2019; Azencot *et al.*, 2020; Han *et al.*, 2020; Surana, 2020; Bevanda *et al.*, 2021; Brunton *et al.* 2021; Lange *et al.*, 2021; Wang *et al.*, 2022). On a high level, Koopman theory is based on the insight that a nonlinear dynamic system can be fully described using an operator that describes how scalar functions propagate over time. The Koopman operator is linear, and thus, preferable in practice. However, the Koopman operator maps between function spaces, and it is infinite-dimensional. When the Koopman theory was discovered a century ago, this property limited its applications (Koopman 1931). However, advancements in computation and operator theory has led to a revived interest. Nowadays, machine learning can be utilized to learn a data transformation under which an approximate finite-dimensional Koopman operator is available. This data mapping can be represented by an autoencoder network, which embeds the high-dimensional input onto a low-dimensional latent space.

CONTACT Jianjun Shi Jianjun.shi@isye.gatech.edu
Supplemental data for this article can be accessed online at https://doi.org/10.1080/24725854.2023.2207615.

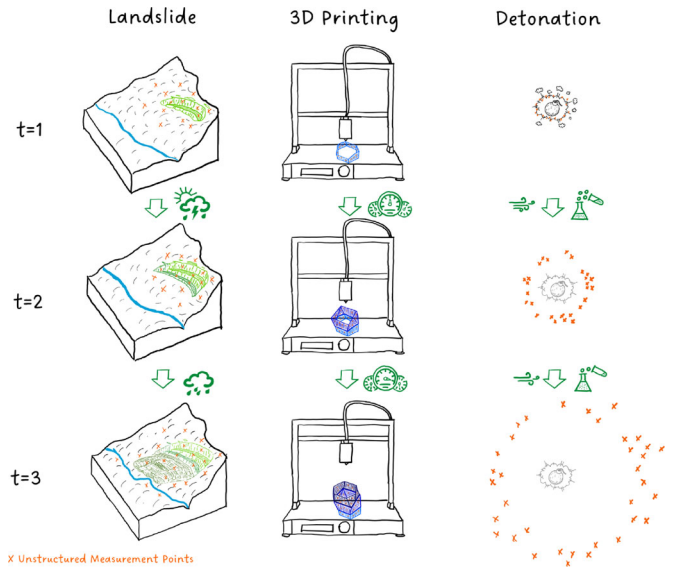


Figure 1. Overview of possible application scenarios for the DETONATE Framework.

In this latent space, the Koopman operator is approximated using a linear layer that encodes the dynamics (Takeishi *et al.*, 2017).

Based on Koopman theory, we proposed a novel framework for the modeling of nonlinear dynamically evolving 3D shape profiles. Our model simultaneously leverages the forward and backward dynamics of the 3D shape profiles to obtain consistent prediction results. Although not for all systems (e.g., diffuse systems) the backward map exists, this assumption holds in many practical cases. Related ideas have been applied to structured time series (Otto and Rowley, 2019; Azencot *et al.*, 2020; Nayak *et al.* 2021, Girgis *et al.*, 2022) and generative adversarial networks (Zhu *et al.*, 2017; Hoffman *et al.*, 2018).

The main contributions of this article are as follows:

1. We develop a modeling framework for nonlinear dynamically evolving 3D shape profiles that also considers multiple, heterogeneous, multi-modal data inputs.
2. We provide a principled way to integrate heterogeneous, multimodal data (i.e., tabular data, images, time series) into deep-learning frameworks. Our model addresses the need for a generalizable methodology to handle heterogeneous data formats exhibiting complex, nonlinear relationships. It can leverage user-defined pre-trained feature extraction models as part of a unified processing and feature aggregation stage.
3. We show that Koopman operator theory for dynamic, time-dependent shape modeling is especially suitable for high-dimensional (unstructured) 3D data that exhibits strong short-term dependencies.
4. We evaluate the performance of our proposed framework on both clean and noisy systems including 3D blast explosions and additive manufacturing and achieve superior results with our model.
5. We conduct extensive case study experiments and show how to leverage 3D shape profiles along with heterogeneous process data in a unified way. We see this as a guiding example for the design and modeling of novel machine learning systems for dynamic 3D shape modeling.

The remainder of the article is organized as follows. Section 2 gives a brief literature review. Then the proposed DETONATE framework for consistent temporal forecasting and backcasting with heterogeneous input and unstructured 3D point cloud output is introduced in Section 3. Section 4 validates the proposed methodology by using simulated data. Furthermore, the performance of the proposed method is compared with existing benchmark methods in terms of estimation accuracy and computational time. In Section 5, we conduct a real-world case study for predicting the 3D shape evolvement in additive manufacturing. Finally, we conclude the article with a brief discussion and an outline of future work in Section 6.

2. Literature review

In this section, we will review four major categories of methodology related to the DETONATE framework: tensor-based methods for 3D time series modeling, deep-learning-based methods for 3D point cloud times series modeling, methods for modeling the forward and backward dynamics, and data-driven Koopman frameworks.

2.1 Tensor-based methods for 3D time series modeling

The analysis of tensor (multidimensional array) time series is one of the most active areas of modern statistical methodology. The classic Vector AutoRegressive (VAR) model is fundamental to multivariate time series modeling and has recently been applied to the high-dimensional case (Zheng and Cheng, 2021). To alleviate the curse of dimensionality when estimating a large number of parameters, Wang *et al.* (2021) proposed a high-dimensional VAR time series model that utilizes tensor decomposition for dimensionality reduction. Other tensor time series models based on factor models (Chen *et al.*, 2022) and block Hankel tensor ARIMA (Shi *et al.*, 2020) have been proposed. However, one major drawback of tensor-based methods is the restrictive assumptions of (linear) tensor decompositions. Another drawback is the fact that they cannot be applied to unstructured data, which cannot be represented efficiently as a tensor.

2.2 Deep-learning-based methods for 3D point cloud time series modeling

Sequential data are commonly modeled using Recurrent Neural Networks (RNNs) (Elman, 1990). The main difference compared with standard neural networks is the fact that RNNs maintain a hidden state that uses the current input and previous inner states to make predictions. Due to the unstructured data format of 3D point clouds, most of the existing spatiotemporal RNN models cannot be utilized. Point clouds are unordered, so point features and states from two different time steps cannot be directly processed. To this end, PointRNN (Fan and Yang, 2019) utilizes point-based states

based on point coordinates. Gomes *et al.* (2021) extended spatiotemporally local correlation to aggregate point features and those ideas to a spatiotemporal Graph-RNN, which utilizes a layer that learns topological information of point clouds to form a representative spatiotemporal neighborhood. The main drawback of the existing deeplearning-based methods is that they do not consider any additional heterogeneous data inputs, which affect the shape evolvement.

In contrast with deep-learning-based methods, Koopman-based frameworks offer several advantages, such as interpretability and theoretical backing for modeling spatiotemporal dynamics. Due to those properties, it enables a wide variety of future research and downstream tasks, such as Koopman-based control. On the other hand, deep learning approaches for temporal modeling have disadvantages such as the need to train separate models for backward and forward predictions, temporal consistency issues, overparameterization and overfitting, and optimization issues such as exploding and vanishing gradients.

2.3 Forward and backward temporal modeling

Modeling the forward and backward dynamics of a system is essential in applications, where the past, as well as the future system states, are not available. Furthermore, considering both the forward and backward dynamics can improve the performance of the time series models by considering the underlying physical structure of the sequence. The majority of time series models do not consider the forward and backward dynamics concurrently. However, models for missing time series value imputation have leveraged the forward and backward dynamics (Moahmed *et al.,* 2014). Another example is the integration of forward and backward dynamics for wind power forecasting (Zhao *et al.*, 2016). Other models that consider both forward and backward dynamics can be found in economics (Gardini *et al.*, 2009), physics-based modeling (Bot and Csethek, 2016), or resource allocation (He and Zhang, 2013). The main drawback of the existing methods for forecasting and backcasting is the fact that they cannot be applied to unstructured, high-dimensional point cloud time series data. The conversion to structured data is not straightforward and will lead to information loss, especially in cases when no reference object (e.g., CAD (Computer Aided Design) model) is available.

2.4 Data-driven Koopman frameworks

The modeling of dynamical systems via Koopman operators has received increased attention in recent years. Most prominently, Schmid (2010) proposed to approximate the Koopman operator via the Dynamic Mode Decomposition (DMD) algorithm. Many extensions to the original DMD algorithm have been proposed in the last decade. Most prominently, Williams *et al.* (2015) proposed the extended DMD to allow for a better global approximation of Koopman modes, eigenvectors, and eigenfunctions. Most related to our approach are the variants of DMD, which address the dynamics of the symmetric, forward, and backward system (Arbabi and Mezig, 2017; Le Clainche and Vega, 2017; Azencot *et al.*, 2019; Haseli and Cortes, 2019; Salova *et al.*, 2019; Azencot *et al.*, 2020; Nayak *et al.*, 2021; Haseli and Cortes, 2022). The control of dynamical systems via Koopman theory is also an important research area. Pan *et al.* (2021) proposed a control method for large-scale problems utilizing the direct identification of Koopman eigenfunctions via a sparsity-promoting algorithm. In robotics, Abraham and Murphy (2019) use Koopman Operators to derive a linearizable data-driven model and utilize it for model-based control. Lusch *et al.* (2018) were the first to incorporate the Koopman theory into a deep learning-based framework utilizing an Autoencoder structure. Since then, an extension to graph convolution networks has been proposed. However, to the best of our knowledge, no existing methodology addresses the time series modeling for unstructured 3D profile point clouds, while also considering several heterogeneous input data sources.

3. DETONATE methodology

In this section, we introduce the DETONATE framework as an approach to model an unstructured, 3D point cloud profile sequence as a function of heterogeneous input data and the previous time step 3D shape. We assume a set of N 3D point cloud sequences with T time steps is available, denoted by $\mathfrak{X}_{S,i} = \{\mathcal{X}^1_{S,i}, \mathcal{X}^2_{S,i}, ..., \mathcal{X}^T_{S,i}\}$, where i denotes the sample index, $i = 1, ...N$; subscript S denotes 3D shape data; and T is the length of the sequence. Additionally, a sequence of K heterogeneous inputs $\mathfrak{X}_{h,i} = \{\mathcal{X}^1_{h,j,i}..., \mathcal{X}^T_{h,j,i}\}$, $j = 1, ...K$ is available, where j is the index denoting the data inputs and K is the total number of heterogeneous inputs.

3.1 Problem setup using Koopman operator theory

In our work, we focus on the modeling of 3D time series sequences, which can be described by a nonlinear 3D profile propagation model (DETONATE) as follows

$$\mathcal{X}^{t+1}_S = f(\mathcal{X}^t_S, \mathfrak{X}^t_h), \quad \mathcal{X}^t_S \in \mathcal{M} \subset \mathbb{R}^{N_p \times 3}, \ \mathfrak{X}^t_h \subset \mathbb{R}^{m_h}, \tag{1}$$

where $\mathcal{X}^t_S$ denotes the state (i.e., 3D profile) of the system at time $t \in T$; N_p denotes the number of Cartesian coordinate points representing the 3D shape; $\mathfrak{X}^t_h$ are the heterogeneous inputs at time t, and m_h is the dimension of the heterogenous input data sources. Then the map $f: \mathcal{M}' \to \mathcal{M}'$ is a (potentially nonlinear) update on a finite-dimensional manifold $\mathcal{S}$. Here, $\mathcal{M}'$ is the space spanned by $\mathcal{X}^t_S$ and $\mathfrak{X}^t_h$. The associated probability space is denoted by $(\mathcal{M}, \Sigma, \mu)$. The model assumes that future states $\mathcal{X}^{t+1}_S$ only depend on the current state $\mathcal{X}^t_S$, the heterogeneous inputs $\mathfrak{X}^t_h$, and not on information from a sequence of previous states. In the setting of 3D shape evolvement, this assumption is not extremely restrictive, given that shape evolvement exhibits strong short-term dependencies, and we additionally consider other heterogeneous inputs. However, for example, when modeling materials with shape memory effects, this framework could be extended to an autoregressive structure.

To predict future shapes, one could directly train an end-to-end neural network that learns an approximation of the map f. However, the resulting model ignores the prior knowledge about the sequential problem structure and suffers from the curse of dimensionality. Solving directly for f is challenging, since the m, the number of features $m \gg n$, and the number of samples $n = N \cdot T$, thus leading to an underdetermined system. In our setting, $m = N_p \times 3 + m_h$ is extremely large, here m_h is the feature dimension of heterogeneous inputs. An alternative approach is to learn a data transformation (i.e., latent embedding) for the states $\mathcal{X}_S^t$ and $\mathfrak{X}_h^t$ so that the corresponding latent variables x_s^t and x_h^t evolve on a linear path, as illustrated in Figure 2.

Using latent encoding, the dynamics can be approximated using a linear model, which alleviates the curse of dimensionality and allows the integration of prior physics knowledge about the data sequence into the training process. This is a direct implication from Koopman theory, which states that there exists a data transformation for any nonlinear dynamical system so that the future states can be predicted via a linear mapping from t to $t+1$. Formally, the Koopman operator is defined as

$$\mathcal{K}_\varphi g(\mathcal{X}_S, \mathfrak{X}_h) = g \circ \varphi(\mathcal{X}_S, \mathfrak{X}_h), \tag{2}$$

where the dynamics φ induce the Koopman operator $\mathcal{K}_\varphi$ that acts on scalar functions g: $\mathcal{M}' \to \mathbb{R} \in \mathcal{S}$, where $\mathcal{S}$ is a finite-dimensional manifold on $\mathcal{M}'$ (Koopman 1931). Throughout this article, we use the following notations: The function composition $\circ$ takes two functions f and g and produces a function $h = g \circ f$ such that $h(x) = g(f(x))$. A composition operator C_ϕ, with symbol ϕ is a linear operator defined by the rule $C_\phi(f) = f \circ \phi$, where $f \circ \phi$ denotes the function composition. The Koopman operator is a composition operator, as the function g is composed with the map φ. In other words, $g \circ \varphi$ is defined as $g \circ \varphi(\mathcal{X}_S, \mathfrak{X}_h) = g(\varphi(\mathcal{X}_S, \mathfrak{X}_h))$. The operator models the evolution of the scalar function by utilizing future values: $\mathcal{K}_\varphi g$ at $\mathcal{X}_S, \mathfrak{X}_h$ is the value of g evaluated at the future state $\mathcal{X}_S^{t+1}$. We note that our data is equally spaced in time, i.e., Δt is the sampling interval, which is the same for each time t, although extensions to this setting can be accommodated (Tu *et al.*, 2014). Intriguingly, it is easy to show that $\mathcal{K}_\varphi$ is linear for any $\alpha, \beta \in \mathbb{R}$

$$\begin{aligned}\mathcal{K}_\varphi(\alpha g + \beta h) &= (\alpha g + \beta h) \circ \varphi = \alpha g \circ \varphi + \beta h \circ \varphi \\ &= \mathcal{K}_\varphi \alpha g + \mathcal{K}_\varphi \beta h\end{aligned} \tag{3}$$

In the remainder, we assume the backward dynamics τ exists, and we denote the associated Koopman operator by $\mathcal{T}_\tau$. One key obstacle is that the theoretical Koopman operator $\mathcal{K}_\varphi$ is infinite-dimensional. Therefore, we work with the key assumption, that in most practical applications, there exists a transformation ψ, whose conjugation with $\mathcal{K}_\varphi$ leads to a finite-dimensional approximation of the dynamics. Assume the data are elements of $L^2(\mathcal{M}', \mu)$. The Koopman invariant subspace is defined as $\mathcal{G}' \subset L^2(\mathcal{M}', \mu)$ s.t. $\forall g' \in \mathcal{G}'$, $\mathcal{K}_\varphi g' \in \mathcal{G}'$. If $\mathcal{G}'$ is spanned by a finite number of functions, then the restriction of $\mathcal{K}_\varphi$ to $\mathcal{G}'$ becomes a finite-dimensional linear operator. The field of Koopman analysis is concerned with learning the right nonlinear dimensionality reduction function to form an invariant subspace (Brunton *et al.*, 2016). We utilize the approximate forward Koopman matrix $\mathcal{F}$

$$\mathcal{F} = \psi \circ \mathcal{K}_\varphi \circ \psi^{-1}, \ \mathcal{F} \in \mathbb{R}^{\mathcal{K} \times \mathcal{K}}, \tag{4}$$

where ψ and its inverse ψ^{-1} extract and reconstruct the crucial information from $\mathcal{K}_\varphi$. Analogously, we denote the approximate backward Koopman matrix by

$$\mathcal{B} = \psi \circ \mathcal{T}_\tau \circ \psi^{-1}, \ \mathcal{B} \in \mathbb{R}^{\mathcal{K} \times \mathcal{K}} \tag{5}$$

The main focus of the remainder of this methodology section is how to find matrices $\mathcal{F}$ and $\mathcal{B}$, and a nonlinear transformation ψ such that the underlying system dynamics are recovered well.

Additionally, we develop a principled way to incorporate heterogeneous data inputs into the Koopman framework. The methodology overview of the DETONATE framework is visualized in Figure 3. The horizontal direction describes the Koopman-based temporal modeling, whereas the vertical direction shows the treatment of heterogeneous inputs used to enhance the modeling of the nonlinear system dynamics.

3.2 Latent encoding via 3D autoencoders

Many dimensionality reduction techniques have been developed in recent years, which can be applied to estimate a

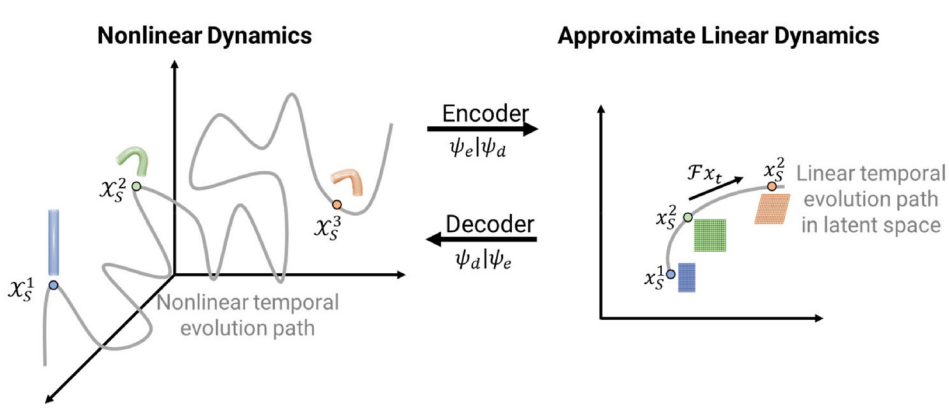


Figure 2. Linearization of nonlinear dynamics via a nonlinear data transformation (Encoder and Decoder) using a latent space.

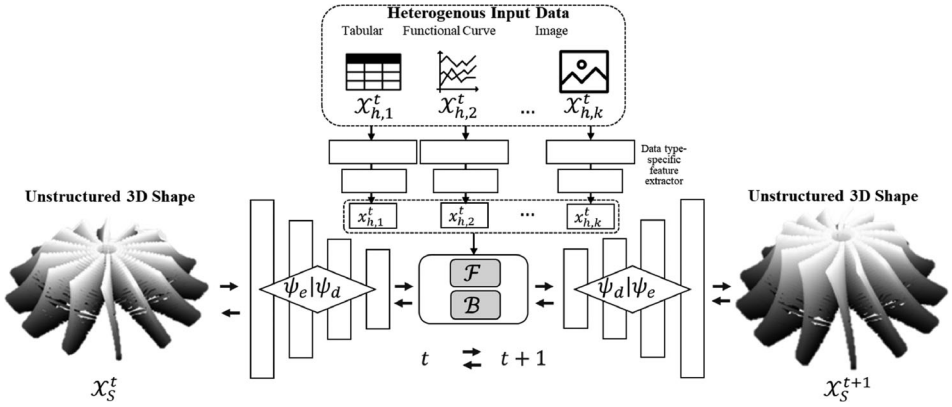


Figure 3. Illustration of the DETONATE methodology.

data transformation ψ, which maps the high-dimensional inputs to a low-dimensional embedding. However, unstructured 3D point clouds cannot be modeled by methods such as fully connected AutoEncoders (AEs). 3D point cloud processing is more challenging than 2D images since point cloud samples live on an irregular structure, whereas 2D images rely on 2D grids (pixels) with regular spacing. Additionally, point cloud geometries are represented by a set of sparse 3D points. These data characteristics make it difficult to apply traditional machine-learning frameworks.

Therefore, we propose to utilize recent advances in unsupervised learning for point clouds to model the transformation ψ. In particular, we utilize a deep point cloud AE. The input to the encoder is a $N_p \times 3$ matrix, where N_p is the number of measurement points in each sample. Each row of the matrix is composed of 3D positional coordinates (x, y, z). The output is a $m \times 3$ matrix, representing the reconstructed point positions. The number of reconstructed points m is not necessarily the same as N_p. Therefore, the loss between the input point set S_1 and the reconstructed point set S_2 can be computed via the extended Chamfer distance

$$d_{eCH}(S_1, S_2) = \max\left\{\frac{1}{|S_1|}\sum_{x\in S_1}\min_{y\in S_2}\|x-y\|_2, \frac{1}{|S_2|}\sum_{y\in S_2}\min_{x\in S_1}\|y-x\|_2,\right\} \tag{6}$$

The term $\min_{y\in S_2}\|x-y\|_2$ enforces that any 3D point x in the original point cloud has a matching 3D point y in the reconstructed point cloud and the term $\min_{x\in S_1}\|y-x\|_2$ enforces the matching vice versa cases. The max operation enforces that the distance from S_1 to S_2 and the distance vice versa cases have to be small simultaneously. If there are multiple matching 3D points with the same distance, any matching 3D point can be chosen randomly, since it does not change the computation of the distance.

We note that the number of measurement points may vary across samples, due to different data acquisition techniques. Therefore, we apply sampling strategies to up- or down-sample all the 3D point clouds to a fixed number of measurement points. For up-sampling, we use furthest point sampling and for down-sampling, we use random sampling without replacement, which are standard techniques in point cloud processing (Krim and Yezzi, 2006). We note that if the number of measurement points varies significantly or very high-resolution 3D shapes are needed, some existing works explicitly handle the problem of varying-sized, unstructured 3D point clouds (Biehler *et al.*, 2022).

In a nutshell, the deep AE computes a latent representation for each input point cloud and the decoder reconstructs the point cloud using the latent representation. We have evaluated several deep AE architectures for 3D point clouds and have confirmed the state-of-the-art performance of the FoldingNet (Yang *et al.*, 2018). For the encoder, this method utilizes graph-based enhancement to promote the learning of local structures. For the decoder, a folding-based operation deforms a canonical 2D grid onto the underlying 3D object surface of a point cloud, leading to high-accuracy reconstructions of delicate 3D structures. We initialize our network with a pretrained model on ShapeNet and train on our dataset. We can achieve reasonable reconstruction performance with a sample size of about 100 samples, which is in line with the findings of previous studies on 3D point cloud AEs. For instance, Yan *et al.* (2022) demonstrated that a FoldingNet-based architecture, pre-trained on a larger dataset, can achieve good performance for a sample size of 100. For architecture and implementation details, interested readers are referred to Yang *et al.* (2018).

More formally, given a set of N observations $\mathcal{D} = \{\mathcal{X}_{S,i}\}_{i=1}^N$ where each $\mathcal{X}_{S,i}$ is a sequence of T time steps, we embed our inputs into a low-dimensional latent space using the nonlinear map ψ_e, represented by the point cloud encoder. The decoder map ψ_d allows the reconstruction of the latent embeddings into the spatial domain. To train the AE, we define

$$\hat{\mathcal{X}}_S = \psi_d \circ \psi_e(\mathcal{X}_S), \tag{7}$$

where $\hat{\mathcal{X}}_S$ denotes the reconstructed version of $\mathcal{X}_S$ utilizing the AE. Consequently, the reconstruction loss for our DETONATE framework becomes

$$\mathcal{L}_{rec} = \sum_{i=1}^{N}\sum_{t=2}^{T-1} d_{eCH}(\mathcal{X}_S, \hat{\mathcal{X}}_S) \tag{8}$$

3.3 Backward dynamics

Most time series models for dynamic systems φ are concerned with the forward prediction to future time points.

However, in several applications, modeling the backward dynamics $\omega: \mathcal{X}_S^t \to \mathcal{X}_S^{t-1}$ is an important problem. For example, in warhead blast experiments, backward dynamics are utilized to design warheads that minimize civilian casualties (Mulekar *et al.*, 2021). Most existing time series modeling techniques do not consider the backward dynamics in their modeling or training.

Additionally, in our framework, the time sequence of high-dimensional data is assumed to be measured by sensor devices. Naturally, in these settings, the observations are assumed to be corrupted by various types of noise. The existence of process or sensor noise (e.g., 3D surface reflections) may result in a bias in traditional time series models that only consider the forward modeling. By considering both forward and backward dynamics, much of the bias due to noise can be eliminated. In the remainder of this work, we will assume additive white Gaussian noise as the noise distribution.

To account for the forward as well as the backward dynamics, we incorporate two linear layers with no biases into our network to represent the approximate forward and backward Koopman operators (Takeishi *et al.*, 2017; Dogra and Redman, 2020). In particular, the Koopman operators $\mathcal{F}$ and $\mathcal{B}$ allow obtaining forward forecasts $\hat{\mathcal{X}}_S^{t+1}$ and backward backcasts $\breve{\mathcal{X}}_S^{t-1}$. This leads to better generalizability and more accurate forecasting and backcasting of the proposed DETONATE framework. We define the following loss terms to model the forward and backward dynamics:

$$\mathcal{L}_{fwd} = \sum_{i=1}^{N}\sum_{t=1}^{T-1} d_{eCH}(\mathcal{X}_S^{t+1}, \hat{\mathcal{X}}_S^{t+1}) \tag{9}$$

$$\mathcal{L}_{bwd} = \sum_{i=1}^{N}\sum_{t=2}^{T} d_{eCH}(\mathcal{X}_S^{t-1}, \breve{\mathcal{X}}_S^{t-1}) \tag{10}$$

A schematic illustration of this structure is given in Figure 3 on the horizontal axis. The design includes an encoder and decoder architecture (i.e., ψ_e and ψ_d), as well as the Koopman matrices $\mathcal{F}$ and $\mathcal{B}$. Note all the connections on the horizontal axis are bi-directional, representing that data can flow from left to right and right to left.

3.4 Consistent dynamics

Because the backward mapping is highly unconstrained (i.e., paired forward and backward data is not available), the accuracy of backcasts typically falls short of the forward mapping (Zhu *et al.*, 2017; Hoffman *et al.*, 2018). Therefore, we would like to exploit the forward Koopman matrix $\mathcal{F}$ to obtain more accurate backcasts. A straightforward approach to this is to model the backward evolution as the inverse of $\mathcal{F}^{-1}$ as follows:

$$\breve{\mathcal{X}}_S^{t-1} = \psi_d \circ \mathcal{F}^{-1}(\mathcal{X}_S^t, \mathcal{X}_h^t) \tag{11}$$

However, the backward loss $\mathcal{L}_{bwd}$ does not consider this inverse relationship. This means that the backward and forward prediction errors will not affect the optimization of the complementary Koopman matrices $\mathcal{F}$ and $\mathcal{B}$. Therefore, we will introduce a loss function, which promotes consistent dynamics in both directions. An initial version of this loss function would take the form

$$\min_{\mathcal{F}} \sum_{i=1}^{N}\sum_{t=2}^{T-1} \left\{ \frac{1}{2}\|\mathcal{F}(\mathcal{X}_S^t, \mathcal{X}_h^t) - \mathcal{X}_S^{t+1}\|_F + \frac{1}{2}\|\mathcal{X}_S^t - \mathcal{F}^{-1}(\mathcal{X}_S^{t+1}, \mathcal{X}_h^{t+1})\|_F \right\} \tag{12}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. If $\mathcal{X}_h^t = \mathcal{X}_h^{t+1}$ and $\mathcal{F}$ is orthogonal (i.e., $\mathcal{F}^{-1} = \mathcal{F}^T$), then the terms $\|\mathcal{F}(\mathcal{X}_S^t, \mathcal{X}_h^t) - \mathcal{X}_S^{t+1}\|_F$ and $\|\mathcal{X}_S^t - \mathcal{F}^{-1}(\mathcal{X}_S^{t+1}, \mathcal{X}_h^{t+1})\|_F$ in (12) are equal. However, in a general case, even after dropping the heterogeneous inputs for the case $\mathcal{X}_h^t = \mathcal{X}_h^{t+1}$, we have

$$\begin{aligned} \|\mathcal{F}(\mathcal{X}_S^t) - \mathcal{X}_S^{t+1}\|_F &= Tr((X_S^t)^T\mathcal{F}^T\mathcal{F}\mathcal{X}_S^t \\ &-2(\mathcal{X}_S^t)^T\mathcal{F}^T\mathcal{X}_S^{t+1} + (\mathcal{X}_S^{t+1})^T\mathcal{X}_S^{t+1}) \neq Tr((X_S^t)^T\mathcal{X}_S^t \\ &-2(\mathcal{X}_S^t)^T\mathcal{F}^{-1}\mathcal{X}_S^{t+1} + (\mathcal{X}_S^{t+1})^T\mathcal{F}^T\mathcal{F}^{-1}\mathcal{X}_S^{t+1}) \\ &= \|\mathcal{X}_S^t - \mathcal{F}^{-1}(\mathcal{X}_S^{t+1})\|_F \end{aligned} \tag{13}$$

Due to the inverse forward dynamics $\mathcal{F}^{-1}$, this optimization problem is highly nonconvex. Therefore, we reformulate (12) by utilizing the backward dynamics $\mathcal{B} = \mathcal{F}^{-1}$ as an auxiliary variable;

$$\min_{\mathcal{F},\mathcal{B}} \sum_{i=1}^{N}\sum_{t=2}^{T-1} \left\{ \frac{1}{2}\|\mathcal{F}(\mathcal{X}_S^t, \mathcal{X}_h^t) - \mathcal{X}_S^{t+1}\|_F + \frac{1}{2}\|\mathcal{X}_S^t - \mathcal{B}(\mathcal{X}_S^{t+1}, \mathcal{X}_h^{t+1})\|_F \right\} \; s.t. \; \mathcal{F}\mathcal{B} = I, \; \mathcal{B}\mathcal{F} = I, \tag{14}$$

where the constitutive constraints guarantee that the minimizers of (14) are the inverse of each other. From an optimization perspective, those two constraints are equivalent. However, in practice, we utilize them to make the approximate invertible relations of $\mathcal{F}$ and $\mathcal{B}$ symmetric. For designing the DENOTATE loss function, we rely on the observation that if the approximate Koopman operators span a Koopman-invariant subspace, then $\mathcal{F}\mathcal{B} = I$. In particular, the consistency error between the forward and backward mapping can be quantified by $M_C = \|\mathcal{F}\mathcal{B} - I\|_F$. The consistency index $\mathcal{I}_C = sprad(M_C) \in [0, 1]$, where $sprad(A) := \max\{|\lambda| \mid \lambda \in spec(A)\}$ is the spectral radius of A. The spectral radius is the largest Koopman eigenvalue associated with the evolution of the observable. From a methodology perspective - beyond the ability to obtain both forecasting and backcasting - enforcing consistency avoids overfitting, and hence, leads to better generalization. This modeling choice is directly motivated by the theoretical results on extended dynamical mode decomposition, which is the most popular dimensionality reduction algorithm for learning approximate Koopman operators. In particular, Haseli and Cortes (2022) have shown that the temporal forward-backward consistency measures the prediction accuracy of approximate Koopman operators.

Proposition 1: *Under mild assumptions, the relative root mean squared error of the approximate Koopman operator is tightly upper bounded by the consistency index* $\sqrt{\mathcal{I}_C}$.

We refer interested readers to Haseli and Cortes (2022) for a detailed proof of this proposition.

This theoretical finding motivates us to directly integrate the consistency of the invertible system dynamics as an additional loss function into the DENOTATE framework.

To understand the properties of inversion, we resort to the spectral representation of the Koopman operator in the function space $\mathcal{S}$. We choose an orthogonal basis $\{e_q\}_{q=0}^{\infty}$ for $\mathcal{S}$, where for any r, s we have

$$\langle e_r, e_s \rangle_{\mathcal{M}'} = \int_{\mathcal{M}'}^{0} e_r(x) e_s(x) dx = \delta_{rs}, \tag{15}$$

where δ_{rs} denotes the Kronecker delta function. Under this choice of basis, any function $f_s \in \mathcal{S}$ can be represented by $f_s = \sum_q \langle f_s, e_q \rangle_{\mathcal{M}'} e_q$ (Dunford and Schwartz, 1971). From the linearity of the Koopman operator $\mathcal{K}_\varphi$, follows the following relationship: $\mathcal{K}_{rs} = \langle e_r, \mathcal{K}_\varphi e_s \rangle_{\mathcal{M}'}$. Now we can utilize the results from Ergodic theory to state the conditions for invertibility (Singh and Manhas, 1993; Eisner *et al.*, 2015).

Proposition 2: *Given a manifold $\mathcal{M}'$, a dynamical system φ is invertible if and only if for every r and s the Koopman operators $\mathcal{F}_\varphi$ and $\mathcal{B}_\omega$ satisfy $\langle e_r, \mathcal{B}_\omega \mathcal{F}_\varphi e_s \rangle_{\mathcal{M}'} = \delta_{rs}$.*

Proof: ($\Rightarrow$): If φ is invertible, then the composition $\omega \circ \varphi = I$ for every $x \in \mathcal{S}$. Hence, $\langle e_r, \mathcal{B}_\omega \mathcal{F}_\varphi e_s \rangle_{\mathcal{M}'} = \int_{\mathcal{M}'}^{0} e_r(x) e_s (\omega \circ \varphi(x))\ dx = \langle e_r, e_s \rangle_{\mathcal{M}'}$.

($\Leftarrow$): Per assumption, we have $\langle e_r, \mathcal{B}_\omega \mathcal{F}_\varphi e_s \rangle_{\mathcal{M}'} = \delta_{rs}$ for all r, s. Thus, for every q, we have $\mathcal{B}_\omega \mathcal{F}_\varphi e_q = e_q$ since e_q is orthogonal to every $e_o, o \neq q$. Then for a scalar function f, we have $f(\omega \circ \ \varphi(x)) = \mathcal{B}_\omega \mathcal{F}_\varphi f(x) = f(x)$ and it follows that $\omega \circ \ \varphi = I$. ■

The results from the proposition along with the results from consistent dynamic mode decomposition (Le Clainche and Vega, 2017; Azencot *et al.*, 2019; Haseli *et al.*, 2019) motivate the consistency loss $\mathcal{L}_{con}$.

Formally, two mappings are considered consistent if $\mathcal{F} \circ \mathcal{B}(\mathcal{X}_S) = \mathcal{X}_S$ for any $\mathcal{X}_S \in \mathcal{M}'$. In the Koopman matrix setting, this is equivalent to requiring $\mathcal{F}\mathcal{B} = I_\mathcal{K}$, where $I_\mathcal{K}$ is the identity matrix of size $\mathcal{K}$. For the remainder of this work, we assume that the underlying system dynamics to be modeled are invertible.

Using our previous notation, where the approximate forward and backward Koopman operators $\mathcal{F}$ and $\mathcal{B}$ are $\mathcal{K} \times \mathcal{K}$, the above condition reduces to

$$\frac{1}{2} \|\mathcal{B}\mathcal{F} - I_\mathcal{K}\| = 0, \tag{16}$$

However, in the discrete setting, a slightly modified version of this condition is necessary. Based on recent results from dynamic mode decomposition, we derive a consistent loss for our Koopman-based framework (Azencot *et al.*, 2019; Salova *et al.*, 2019; Haseli and Cortes, 2022). We also note the connection to the consistency of functional maps (Huang *et al.*, 2014). In line with existing theory, we assume that the manifold $\mathcal{M}$ is represented by the domain $M \subset \mathbb{R}^{N_p \times 3}$, which is sampled using m vertices. Under this assumption, the scalar functions $f: M \to \mathbb{R}$ are vectors $f \in \mathbb{R}^m$ storing values of vertices with intermediate values obtained via interpolation. The encoding map $\varphi: \mathcal{M}' \to \mathcal{M}'$ can be defined using a matrix $P_\varphi \in \mathbb{R}^{m \times m}$

$$P_\varphi \delta_x = g_{\varphi(x)}, \tag{17}$$

where g_x is a function that stores the vertex coefficients such that $g_x^T C = x^T$, with $C \in \mathbb{R}^{m \times N_p \times 3}$ being the spatial coordinates of M. Analogously, we denote P_ω as the matrix associated with ω by $P_\omega \delta_x = g_{\omega(x)}$. Now we denote $B \in \mathbb{R}^{m \times m}$ as the matrix containing the orthogonal basis elements in its columns (i.e., $\langle b_u, b_v \rangle_M = b_u^T b_v = \delta_{uv}$ for every u, v). Then we can define the forward and backward matrices as follows

$$\begin{aligned} \mathcal{F} &= B^T P_\varphi B \\ \mathcal{B} &= B^T P_\omega B \end{aligned} \tag{18}$$

Formally, a discrete map φ is consistent if, for every x, we have $\varphi \circ \varphi(x) = x$. Utilizing the results from functional mappings and consistent dynamical mode decomposition (Le Clainche and Vega, 2017; Azencot *et al.*, 2019; Haseli and Cortes, 2019), we can state and prove the following theorem.

Theorem 1: *Given a domain M, the map φ is consistent if and only if for every u and v the forward and backward matrices $\mathcal{F}$ and $\mathcal{B}$ satisfy*

$$\sum\nolimits_{l=1}^{\mathcal{K}} \frac{1}{2l} \|\mathcal{B}_l \mathcal{F}_l - I_l\|_F^2 = 0,$$

where $\mathcal{B}_l$ and $\mathcal{F}_l$ are the upper l rows of $\mathcal{B}$ and the leftmost l columns of the Koopman matrix $\mathcal{F}$.

Proof: ($\Rightarrow$): If φ is a consistent map, then $P_\omega P_\varphi \delta_x = \delta_x$ for every x and thus $P_\omega P_\varphi = I$. In addition, for every l we have that $\mathcal{B}_l \mathcal{F}_l = B_l^T P_\omega B B^T P_\varphi B_l = B_l^T P_\omega P_\varphi B_l = B_l^T B_l = I$, where B_l are the first l basis elements of B and $BB^T = I$.

($\Leftarrow$): Furthermore, we assume that the forward and backward maps are constructed using (18) and the condition $\sum_{l=1}^{\mathcal{K}} \frac{1}{2l} \|\mathcal{B}_l \mathcal{F}_l - I_l\|_F^2 = 0$ holds. Then $B_l^T P_\omega P_\varphi B_l = I_l$ for every l. By induction on l it follows that $P_\omega P_\varphi b_l = b_l$, for every l, where b_l are the l th columns of B and thus $P_\omega P_\varphi = I$ as B spans the space of scalar functions on M. ■

Therefore, we incorporate the following loss into our DETONATE framework to promote consistency

$$\mathcal{L}_{con} = \sum_{i=1}^{N} \sum_{t=2}^{T-1} \sum\nolimits_{l=1}^{\mathcal{K}} \left\{ \frac{1}{2l} \|\mathcal{B}_l \mathcal{F}_l - I_l\|_F^2 + \frac{1}{2l} \|\mathcal{F}_l \mathcal{B}_l - I_l\|_F^2 \right\}. \tag{19}$$

This loss is directly motivated by the consistent dynamics of the time series sequences and derived based on recent advances in the consistency of functional mappings and dynamic mode decompositions.

The consistency loss in DETONATE serves as a regularization technique that addresses the curse of dimensionality, which is a common problem in many engineering applications, where the number of training samples is significantly smaller than the number of model parameters. Although

typical regularization methods promote sparsity by penalizing model weights, different neural architectures, data modalities, and learning problems may require alternative regularization techniques. In DETONATE the consistency loss can be interpreted as a relaxed version of strict reversibility and stability constraints in physics-informed learning, while still regularizing the parameter space.

3.5 Heterogeneous input data sources

The accurate 3D sequence modeling of complex systems requires knowledge from multiple data sources and various, heterogeneous input modalities. Multimodal deep learning architectures are a natural extension to existing single modality frameworks: they emulate the approach that domain experts or engineers currently use to perform predictions for complex problems. Typical engineering practice uses a diverse set of data formats such as tabular data (e.g., process settings, material properties), image data (e.g., photographs, infrared images, X-ray, and computerized-tomograph scans), and functional curves (e.g., temperature data, acoustic emissions). Although recent efforts have utilized linear tensor-decompositions to fuse multiple sources of heterogeneous data (Gahrooei *et al.*, 2021), those techniques are not applicable during the model deployment in real-time and the modeling of complex, nonlinear, and extremely high-dimensional heterogeneous data sources. This motivates the need for a generalizable and modular deep learning approach to extract information from heterogeneous data sources. It can leverage user-defined pretrained feature-extraction models as part of a processing and feature aggregation stage that allows the downstream modeling of a variety of predictive tasks. This process is visualized in the vertical direction in Figure 3, which shows how multiple, heterogeneous data sources $\mathcal{X}^t_{h,1}, \mathcal{X}^t_{h,2}, \ldots, \mathcal{X}^t_{h,k}$ are fed into data type-specific feature extractors to obtain their respective latent embeddings $x^t_{h,1}, x^t_{h,2}, \ldots, x^t_{h,k}$.

$$\mathcal{L}_h = \min_{\theta_h} \sum_{i=1}^{N} \sum_{t=1}^{T-2} \sum_{j=1}^{k} \|f_{h,k}\left(x^t_{h,j}, \theta_h\right) - \mathcal{X}^t_{h,j}\|_F^2, \tag{20}$$

where $f_{h,k}$ denotes the data-type-specific feature extractor with parameters θ_h. To integrate the heterogeneous data into the DETONATE framework, we directly integrate the learning of latent embeddings $x^t_{h,j}$ into the overall DETONATE loss. By jointly optimizing heterogeneous data loss with the temporal model, we learn those latent embeddings of the heterogeneous data that are most correlated with the temporal evolution model. In other words, we want to extract the part of the variation in the heterogeneous input that contributes the most to the temporal evolution of the 3D profiles.

The data-type-specific feature extractor $f_{h,k}$ can be chosen as follows: (i) based on engineering domain knowledge or (ii) based on well-established knowledge about feature extractors for certain data modalities. Common feature extractors for image, functional curve, and tabular data are listed in Table 1 and serve as guiding examples for the selection of $f_{h,k}$.

Table 1. Guiding examples for the selection of the data-type-specific feature extractor $f_{h,k}$.

Data Type	*Data-type-specific feature extractor $f_{h,k}$*
Image	Convolution Neural Network, Speeded up robust features (SURF), Histogram of Oriented Gradients (HOG)
Tabular	Feature Selection via Regularized Regression (e.g., LASSO), Fully connected Neural Network
Functional Curves	Wavelet Basis, Time-Frequency (Fourier) transformations

Our approach is motivated by insights from transfer learning related to layer (un-)freezing (Guo *et al.*, 2019). To avoid convergence issues, the AE is first trained without heterogeneous data only on the shape reconstruction tasks. Then, we freeze the early layers of the AE architecture to enable learning generic 3D shape features. Those layers are marked gray in Figure 4. Then, we fine-tune the last p layers by utilizing ideas from multi-head attention to jointly model information from different data subspaces. We extract the hidden activations $a_p^{h,j,t}$ from each heterogeneous data source and concatenate them with the corresponding p hidden activation layer in the AE architecture. The same process is applied to the last layer containing the latent embeddings of the heterogeneous data sources $x^t_{h,j}$, which are added to the penultimate activation layer of the AE. This process is illustrated in Figure 4. If no intermediate hidden layers are available for the heterogeneous data feature extractors, only the latent embeddings $x^t_{h,j}$ are concatenated. If the heterogeneous data are added at all layers (i.e., p is equal to the depth of the AE), this approach reduces to a multi-input neural network architecture, which can be easily implemented with deep learning packages.

3.6 Unified DETONATE framework

In this section, we propose an efficient framework to simultaneously optimize the components of the DETONATE framework. The DETONATE framework allows us to model the time series sequence of unstructured, 3D point clouds $\mathcal{X}_S^{t+1}$ as a function of their previous time step shape $\mathcal{X}_S^t$ as well as multiple heterogeneous inputs $\mathcal{X}^t_{h,j}$ by utilizing the following loss function:

$$\begin{aligned}
\mathcal{L}_{DETONATE} &= \mathcal{L}_{rec} + \lambda_1 \cdot \mathcal{L}_{fwd} + \lambda_2 \cdot \mathcal{L}_{bwd} + \lambda_3 \cdot \mathcal{L}_{con} + \lambda_4 \cdot \mathcal{L}_h \\
&= \sum_{i=1}^{N} \sum_{t=2}^{T-1} d_{eCH}(\mathcal{X}_S, \hat{\mathcal{X}}_S) + \lambda_1 \cdot \sum_{i=1}^{N} \sum_{t=2}^{T-1} d_{eCH}(\mathcal{X}_S^{t+1}, \hat{\mathcal{X}}_S^{t+1}) \\
&\quad + \lambda_2 \cdot \sum_{i=1}^{N} \sum_{t=2}^{T-1} d_{eCH}(\mathcal{X}_S^{t-1}, \breve{\mathcal{X}}_S^{t-1}) \\
&\quad + \lambda_3 \cdot \sum_{i=1}^{N} \sum_{t=2}^{T-1} \sum_{l=1}^{\mathcal{K}} \left\{ \frac{1}{2l} \|\mathcal{B}_l \mathcal{F}_l - I_l\|_F^2 + \frac{1}{2l} \|\mathcal{F}_l \mathcal{B}_l - I_l\|_F^2 \right\}. \\
&\quad + \lambda_4 \cdot \sum_{i=1}^{N} \sum_{t=1}^{T-2} \sum_{j=1}^{k} \|f_{h,k}\left(x^t_{h,j}, \theta_h\right) - \mathcal{X}^t_{h,j}\|_F^2,
\end{aligned} \tag{21}$$

where $\lambda_1, \lambda_2, \lambda_3, \lambda_4 \in \mathbb{R}^+$ are turning parameters. The minimization of the DETONATE loss guarantees to achieve good AE reconstruction performance while keeping forecasts

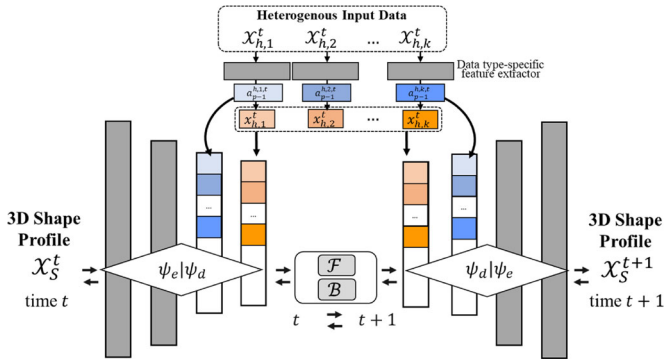


Figure 4. Holistic framework to integrate heterogeneous data source into deep learning models.

and backcasts in time accurate, and ensuring the consistency of forward and backward dynamics.

One of the key features of the model is that it allows direct backcasting. Given an observation $\mathcal{X}_S^t$ and heterogeneous inputs $\mathcal{X}_{h,j}^t$, the DETONATE model yields the forward prediction $\hat{\mathcal{X}}_S^{t+1} = \psi_d \circ \mathcal{F} \circ \psi_e(\mathcal{X}_S^t, \mathcal{X}_h^t)$, as well as the backcasts $\check{\mathcal{X}}_S^{t-1} = \psi_d \circ \mathcal{B} \circ \psi_e(\mathcal{X}_S^t, \mathcal{X}_h^t)$. Typically, neural networks require the training of a separate model in the reverse direction to be able to perform backcasting. We note that in general, multi-step predictions are possible by iteratively substituting one-step predictions. However, this extension is non-trivial and may require further assumptions on future inputs and noise distributions. Therefore, we mention it at this stage as a topic worthy of further investigation.

An important property of the proposed DETONATE loss is the fact that it allows joint optimization with respect to all parameters using stochastic gradient descent algorithms. This allows for automatic differentiation and the integration of the proposed framework into efficient computing environments.

3.7 Tuning parameter selection

The DETONATE framework relies on four main tuning parameters $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ in the loss function. The use of machine learning algorithms commonly involves careful tuning of learning parameters requiring expert experience, rules of thumb, or bruteforce search. On the contrary, we view this issue as the global derivative-free optimization of an unknown (nonconvex) black-box function and utilize the Bayesian optimization procedure proposed by Snoek *et al.* (2012) with its accompanying Python package "Spearmint" to automatically optimize the performance of the DETONATE framework for a given problem. Bayesian optimization has been shown to outperform other global optimization algorithms for tuning parameter selection on several multimodal black-box functions (Jones, 2001). The objective of the hyper-parameter tuning is to maximize the predictive performance of forecasts and backcasts (i.e., $\mathcal{L}_{fwd} + \mathcal{L}_{bwd}$). We weight the importance of each task equally. The parameters tuned by the Bayesian optimization are $\lambda_1, \lambda_2, \lambda_3$ and λ_4. Spearmint uses a Gaussian process prior and the Expected Improvement (EI) criterion as an acquisition function. For further implementation details, readers are referred to Snoek *et al.* (2012) and the corresponding code repository.

4. Simulation studies – 4D warhead detonations

In this section, we evaluate the DETONATE framework using a simulated 4D warhead fragmentation in-flight behavior. Warhead detonations eject fragments over large distances, causing collateral damage to structures, vehicles, and persons. We represent the fragment locations as unstructured 3D point clouds evolving over time as a function of heterogeneous inputs modeled by a characteristic (time-varying) velocity curve. To avoid civilian casualties, it is imperative to understand the in-flight detonation behavior so that collateral damage and lethality estimates can be used to make decisions regarding effective target application. However, the current state of the art is expensive field testing. The installation of vision systems enables the real-time tracking of fragments and enables new statistical analysis of the 3D evolvement of the blast. It allows the comparison and contrasting of detonations under a wide range of process conditions. Furthermore, the backward modeling of warhead detonations is vital for forensic tasks. Therefore, we use our DETONATE model as a guiding example for another important application with the same data characteristic: an unstructured 3D shape nonlinearly evolves as a function of time and (multiple) heterogeneous inputs.

We start by initializing the 3D shape starting from the coordinate origin:

$$\mathcal{X}_S^0 = \overrightarrow{0} \in \mathbb{R}^{N_p \times 3} \tag{22}$$

Then the evolvement of the 3D shape is modeled using the previous time steps shape and multiple heterogeneous inputs $\mathcal{X}_{h,j}$. According to Gurney's equation (Gurney, 1943), the initial velocity of a fragment from a cylindrical casing is expressed by

$$v_0(\mathcal{X}_{h,1}, \mathcal{X}_{h,2}, \mathcal{X}_{h,3}) = \sqrt{2(\beta_{h,1}^0 + \beta_{h,1}^1 \mathcal{X}_{h,1})} \sqrt{\frac{\mathcal{X}_{h,2}/\mathcal{X}_{h,3}}{1 + 0.5\mathcal{X}_{h,2}/\mathcal{X}_{h,3}}}, \tag{23}$$

where $\beta_{h,1}^0, \beta_{h,1}^1$ are explosive specific constants, $\mathcal{X}_{h,1}$ is the detonation velocity of the explosive, $\mathcal{X}_{h,2}$ is the mass of the explosive, and $\mathcal{X}_{h,3}$ is the mass of the warhead casing, respectively. We consider four different material settings utilized to compute the initial velocity, which are summarized in Table 2.

We consider four different material settings, in each simulation study, we either consider the following settings:

m.i) homogenous material: We only consider material setting A.
m.ii) heterogeneous material: We generate an equal portion of samples (i.e., 25%) for each material setting (A, B, C, and D).

Furthermore, the warhead fragment velocity is impacted by the characteristic flight curve, which depends on the exact design and material properties of the warhead. We consider this

Table 2. Material settings utilized for simulation study.

Material setting	*Heterogeneous Input Data*
Setting A: Explosive: Thermobaric Explosive (TBE) Shell: Carbon Steel (0.45%)	$\beta^0_{h,1} = 520;\ \beta^1_{h,1} = 0.28;\ \mathcal{X}_{h,1} = 8480\frac{\text{m}}{\text{s}};\ \mathcal{X}_{h,2} = 0.5\ \text{kg};\ \mathcal{X}_{h,3} = 1\ \text{kg}$
Setting B: Explosive: Thermobaric Explosive (TBE) Shell: Copper	$\beta^0_{h,1} = 520;\ \beta^1_{h,1} = 0.28;\ \mathcal{X}_{h,1} = 8480\frac{\text{m}}{\text{s}};\ \mathcal{X}_{h,2} = 1\ \text{kg};\ \mathcal{X}_{h,3} = 2.5\ \text{kg}$
Setting C: Explosive: Trinitrotoluene (TNT) Shell: Carbon Steel (0.45%)	$\beta^0_{h,1} = 400;\ \beta^1_{h,1} = 0.2;\ \mathcal{X}_{h,1} = 6940\frac{\text{m}}{\text{s}};\ \mathcal{X}_{h,2} = 0.5\ \text{kg};\ \mathcal{X}_{h,3} = 1\ \text{kg}$
Setting D: Explosive: Trinitrotoluene (TNT) Shell: Copper	$\beta^0_{h,1} = 400;\ \beta^1_{h,1} = 0.2;\ \mathcal{X}_{h,1} = 6940\frac{\text{m}}{\text{s}};\ \mathcal{X}_{h,2} = 1\ \text{kg};\ \mathcal{X}_{h,3} = 2.5\ \text{kg}$

flight curve as a heterogeneous input $\mathcal{X}^t_{h,4}$. We choose the following settings to account for a wide range of external impacts (e.g., warhead design and material, the impact of wind):

v.i) Increasing: $\mathcal{X}_{h,4}(t) = 100 \cdot t\ \left[\frac{\text{m}}{\text{s}}\right]$
v.ii) Varying: $\mathcal{X}_{h,4}(t) = 100 \cdot \sin(t)\ \left[\frac{\text{m}}{\text{s}}\right]$

Using the previous time step's shape and the heterogeneous inputs, we can describe the temporal evolution of the 3D fragments with the following equation

$$\mathcal{X}^t_S = \mathcal{X}^{t-1}_S + v_0(\mathcal{X}_{h,1}, \mathcal{X}_{h,2}, \mathcal{X}_{h,3}) \cdot \Delta t + \mathcal{X}_{h,4}(t) + \frac{1}{2} G \cdot \Delta t^2 + \varepsilon, \tag{24}$$

where $G = (0, 0, 9.8)\ \frac{\text{m}}{\text{s}^2}$ is a constant gravity vector, and $\varepsilon = N(\mu, \Sigma)$ is a 3D multivariate Gaussian, where the mean matrix $\mu \in \mathbb{R}^{N_p \times 3}$ is centered at the new coordinate points of the 3D point cloud and $\Sigma \in \mathbb{R}^{(N_p \times 3) \times (N_p \times 3)}$ is the corresponding covariance tensor with its diagonal entries as σ. These settings are consistent with real experiments of blast experiments (Wang *et al.*, 2019).

We evaluate the robustness of DETONATE with respect to different noise levels and correlation levels between heterogeneous inputs and the 3D shapes, which are modeled using the different characteristic velocity curves.

An example of a blast simulation at different time steps is presented in Figure 5.

Using this procedure, we create $N = 200$ samples, $T = 10$ time steps for each sample, where each step is 1 μs, and $N_p = 5000$ unstructured fragment measurement points for each of the time steps, which is intended to simulate the low sample size in many engineering applications.

To model the heterogeneous input data, we utilize the following data-type-specific feature extraction procedures: For the tabular data (i.e., $\beta^0_{h,1}$, $\beta^1_{h,1}$, $\mathcal{X}_{h,1}, \mathcal{X}_{h,2}, \mathcal{X}_{h,3}$), we directly use the variables without feature extractions. For the functional curve $\mathcal{X}_{h,4}(t)$, we use a fully connected neural network AE structure. The encoder and decoder have the inverse structure of each other. We use one hidden layer with 10 nodes and a bottleneck size of three. The activation functions are chosen as rectified linear units (ReLU).

4.1 Benchmark methods and evaluation metrics

As pointed out in the literature review, to the best of our knowledge, there is no existing method that can model the evolution of an unstructured 3D point cloud shape profile as a function of heterogeneous input data and the previous time steps' 3D shape. However, to thoroughly benchmark the performance of our DETONATE framework, we adopted an existing method based on Graph-RNN (Gomes *et al.*, 2021) to enable the prediction of shape evolvement. Although this method cannot consider heterogeneous inputs, it can predict future frames in a point cloud sequence. Additionally, this method has been shown to outperform the previous state-of-the-art method PointRNN (Fan and Yang, 2019) as well as naïve methods such as "Copy Last Input", which simply uses the shape at the previous (one) time step as a predictor.

To evaluate the performance of the proposed method, we must compare the predicted 3D shape (in the form of a 3D point cloud) against the ground-truth 3D shape, which is also represented by a point cloud. Two permutation-invariant metrics for comparing unordered point sets have been proposed in the literature (Fan *et al.*, 2017). On the one hand, the Earth Mover's Distance (EMD) (Rubner *et al.*, 2000) is the solution to a transportation problem that attempts to transform one set into the other. For two equally sized subsets $S_1 \subseteq \mathbb{R}^3$, $S_2 \subseteq \mathbb{R}^3$, their EMD is defined by

$$d_{EMD}(S_1, S_2) = \min_{\phi: S_1 \to S_2} \sum_{x \in S_1} \|x - \phi(x)\|_2, \tag{25}$$

where ϕ is a bijection. As a loss, EMD is differentiable almost everywhere. On the other hand, the Chamfer (pseudo)-Distance (CD) measures the squared distance between each point in one set to its nearest neighbor in the other set:

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2 \tag{26}$$

The CD is differentiable and compared with EMD more efficient to compute. We will use both metrics to evaluate the performance of the proposed DETONATE framework.

4.2 Ablation studies: Impact of DETONATE loss terms

We conduct an ablation study to investigate the performance of the DETONATE model by removing certain loss terms, which allows us to understand the contribution of the component to the overall prediction performance. The overall goal of the DETONATE framework is accurate forward and backward predictions. Therefore, we cannot remove the forward and backward loss ($\mathcal{L}_{fwd}, \mathcal{L}_{bwd}$) from our model. Additionally,

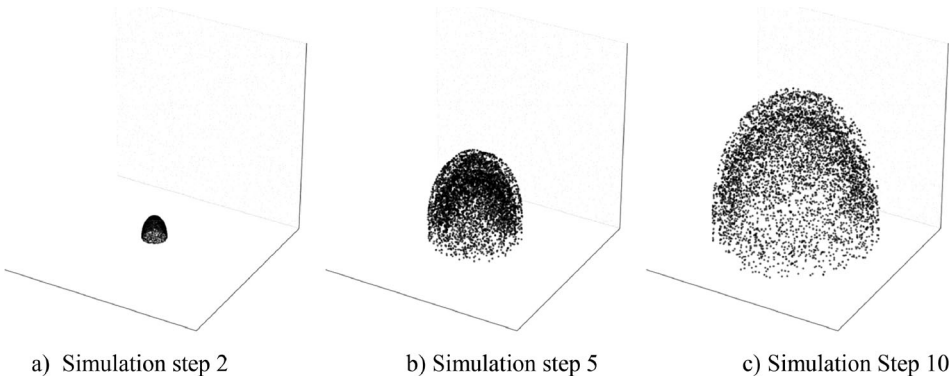


Figure 5. Example of a blast explosion over time.
(a) Simulation step 2; (b) Simulation step 5; (c) Simulation Step 10.

the reconstruction loss $\mathcal{L}_{rec}$ is required to obtain a low-dimensional embedding that is utilized for the forward and backward loss. Therefore, we investigate the impact of removing the consistency loss $\mathcal{L}_{con}$ and the heterogenous input data loss $\mathcal{L}_h$. In particular, we obtain three partial DETONATE models:

1. Partial DETONATE Model 1: Without consistency loss $\mathcal{L}_{con}$ and heterogenous input data loss $\mathcal{L}_h$
$$\mathcal{L}_{Partial_1} = \mathcal{L}_{rec} + \lambda_1 \cdot \mathcal{L}_{fwd} + \lambda_2 \cdot \mathcal{L}_{bwd} \tag{27}$$
2. Partial DETONATE Model 2: Without consistency loss $\mathcal{L}_{con}$
$$\mathcal{L}_{Partial_2} = \mathcal{L}_{rec} + \lambda_1 \cdot \mathcal{L}_{fwd} + \lambda_2 \cdot \mathcal{L}_{bwd} + \lambda_4 \cdot \mathcal{L}_h \tag{28}$$
3. Partial DETONATE Model 3: Without heterogenous input data loss $\mathcal{L}_h$
$$\mathcal{L}_{Partial_3} = \mathcal{L}_{rec} + \lambda_1 \cdot \mathcal{L}_{fwd} + \lambda_2 \cdot \mathcal{L}_{bwd} + \lambda_3 \cdot \mathcal{L}_{con} \tag{29}$$

Note, that the first partial DETONATE Model can be viewed as an AE benchmark since AEs are commonly used for one-step-ahead predictions, and the partial model 1, combines the AE-based reconstruction $\mathcal{L}_{rec}$ with the forward and backward predictions.

The results of the ablation study are discussed in Sections 4.3 and 5.2 for the simulation and case study, respectively.

4.3 Simulation study prediction results

In this section, we will compare the proposed DETONATE framework against the existing GraphRNN method. In each case, we compare the proposed method based on the CD and the EMD, which is also known as the Wasserstein distance. Table 3 reports the average and standard deviation of the evaluation metrics average over both one-step forecasts and backcasts via 10-fold cross-validation. The performance in 10-fold cross-validation is evaluated on an unseen test set (i.e., 4D blast explosions that were not used during training). Since the scale of the point clouds changes significantly across time steps, we normalize the point clouds to be able to compare prediction results across time. Furthermore, exemplary prediction results are visualized in Figure 6.

In Figure 6, the green points show the ground truth warhead fragments. The blue points show the predictions of the proposed DETONATE framework. The red points show the predictions of the GraphRNN benchmark. We can see that the DETONATE predictions match the shape of the ground truth. The GraphRNN on the other hand exhibits a much higher variance. To better evaluate the prediction performance, we also visualize horizontal and vertical slices (gray planes in figures) of the predictions in Figure 6. Then, we project the ground truth and the predictions along the corresponding axis into a 2D image plane. We can see that DETONATE predictions are tightly clustered around the ground truth fragments. The GraphRNN prediction exhibits a large spatial dispersion. We note that scale changes along the temporal dimension since the warhead blast is evolving and getting bigger in radius. To better show the difference in prediction, we added a magnification of the horizontal slice for time step 10. We can still see the same qualitative prediction behavior as in previous time steps: the DETONATE predictions closely match the ground truth, while GraphRNN exhibits a large prediction variance.

From Table 3, we can see that generally, the backward predictions fall short of the forward predictions, which is consistent with the literature. One reason for this phenomenon is the inherent nonlinearities of neural networks, which can pose challenges in constraining both the forward and backward models.

In our simulation studies, for the GraphRNN the backward error is on average 9.3% (CD) higher than the forward error. However, for (full) DETONATE this gap between forward and backward predictions is much smaller (4.4%), due to the consideration of the consistency error. The same pattern can be observed for the standard deviation: the backcast standard deviation of GraphRNN is 10.7% high than the forecast standard deviation, and for DETONATE the backcasts only have a 6.8% higher standard deviation. Therefore, the consistency loss can effectively close the performance gap between forecasts and backcasts commonly observed when training temporal deep learning models.

In all cases, the proposed DETONATE framework outperforms the benchmark method GraphRNN, reflecting the advantage of DETONATE on both clean and noisy systems, with different levels of correlation between the 3D time series and the heterogeneous input data. The performance improvement is due to the following three reasons: (i) consistent forecasting and backcasting, (ii) incorporation of

Table 3. Simulation study results (standard deviation in (brackets), best-performing model in **bold**).

	Simulation Setting	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
Method	*Metric*	$\sigma = 1$ *v.i)* *m.i)*	$\sigma = 1$ *v.ii)* *m.i)*	$\sigma = 1$ *v.i)* *m.ii)*	$\sigma = 1$ *v.ii)* *m.ii)*	$\sigma = 5$ *v.i)* *m.i)*	$\sigma = 5$ *v.ii)* *m.i)*	$\sigma = 5$ *v.i)* *m.ii)*	$\sigma = 5$ *v.ii)* *m.ii)*	$\sigma = 15$ *v.i)* *m.i)*	$\sigma = 15$ *v.ii)* *m.i)*	$\sigma = 15$ *v.i)* *m.ii)*	$\sigma = 15$ *v.ii)* *m.ii)*
Partial DETONATE 1 (No $\mathcal{L}_{con}$, No $\mathcal{L}_h$)	Forward CD	1630.83 (230.44)	1741.17 (244.63)	1559.73 (229.12)	1873.94 (263.55)	1747.69 (271.07)	2591.01 (381.66)	2087.30 (298.90)	2692.31 (408.69)	2432.34 (423.71)	3101.08 (527.18)	2880.78 (490.31)	3330.05 (535.14)
	Forward EMD	0.105 (0.01)	0.143 (0.02)	0.122 (0.02)	0.155 (0.02)	0.152 (0.02)	0.149 (0.02)	0.160 (0.02)	0.174 (0.03)	0.357 (0.06)	0.381 (0.07)	0.489 (0.09)	0.501 (0.09)
	Backward CD	1668.16 (238.71)	1779.66 (237.23)	1592.25 (231.51)	1907.08 (268.14)	1812.82 (277.72)	2494.50 (390.64)	2499.72 (392.21)	2730.11 (394.71)	2434.34 (393.31)	3021.23 (510.59)	2870.66 (479.11)	3469.56 (594.68)
	Backward EMD	0.112 (0.01)	0.144 (0.02)	0.134 (0.02)	0.159 (0.02)	0.145 (0.02)	0.152 (0.02)	0.156 (0.02)	0.176 (0.02)	0.361 (0.06)	0.391 (0.07)	0.489 (0.09)	0.525 (0.09)
Partial DETONATE 2 (No $\mathcal{L}_{con}$, Includes $\mathcal{L}_h$)	Forward CD	1608.31 (227.09)	1612.13 (226.83)	1531.68 (204.79)	1755.18 (243.62)	1737.31 (252.95)	2237.23 (302.25)	2048.88 (305.69)	2311.82 (365.50)	2381.99 (411.61)	2708.72 (461.57)	2812.22 (473.58)	2918.52 (509.57)
	Forward EMD	0.094 (0.01)	0.114 (0.02)	0.113 (0.02)	0.130 (0.02)	0.154 (0.02)	0.130 (0.02)	0.160 (0.02)	0.149 (0.02)	0.330 (0.06)	0.332 (0.06)	0.474 (0.08)	0.501 (0.09)
	Backward CD	1654.54 (221.05)	1743.22 (229.93)	1526.74 (205.65)	1798.53 (244.24)	1784.14 (255.85)	2236.95 (331.74)	2311.82 (341.69)	2398.15 (363.32)	2405.01 (407.89)	2820.13 (479.70)	2820.13 (475.76)	2952.09 (507.17)
	Backward EMD	0.115 (0.02)	0.122 (0.02)	0.117 (0.02)	0.137 (0.02)	0.152 (0.02)	0.139 (0.02)	0.159 (0.02)	0.146 (0.02)	0.338 (0.06)	0.339 (0.06)	0.478 (0.08)	0.511 (0.09)
Partial DETONATE 3 (No $\mathcal{L}_h$, Includes $\mathcal{L}_{con}$)	Forward CD	1638.47 (218.08)	1678.75 (225.29)	1526.35 (199.49)	1755.18 (210.80)	1775.28 (242.50)	2252.45 (304.98)	2138.24 (319.03)	2293.12 (325.85)	2313.36 (347.70)	2898.63 (460.30)	2918.02 (472.14)	2927.93 (490.14)
	Forward EMD	0.099 (0.01)	0.136 (0.02)	0.117 (0.02)	0.133 (0.02)	0.153 (0.02)	0.136 (0.02)	0.155 (0.02)	0.164 (0.02)	0.338 (0.05)	0.332 (0.05)	0.478 (0.08)	0.513 (0.09)
	Backward CD	1640.24 (202.24)	1683.82 (221.76)	1527.12 (205.70)	1798.52 (216.36)	1779.84 (245.62)	2271.82 (316.92)	2185.60 (323.03)	2290.05 (324.27)	2376.37 (358.59)	2988.94 (476.74)	2921.80 (482.39)	3080.52 (496.89)
	Backward EMD	0.098 (0.01)	0.137 (0.02)	0.120 (0.02)	0.134 (0.02)	0.154 (0.02)	0.138 (0.02)	0.159 (0.02)	0.162 (0.02)	0.338 (0.05)	0.333 (0.05)	0.487 (0.08)	0.518 (0.08)
DETONATE (ours)	Forward CD	**1229.26 (139.03)**	**1298.79 (144.13)**	**1300.04 (161.23)**	**1398.79 (185.77)**	**1410.92 (190.37)**	**1726.29 (232.61)**	**1407.52 (205.82)**	**1862.90 (287.61)**	**1843.49 (287.03)**	**1935.95 (315.03)**	**1951.75 (328.24)**	**2131.21 (338.32)**
	Forward EMD	**0.083 (0.01)**	**0.103 (0.01)**	**0.092 (0.01)**	**0.106 (0.02)**	**0.118 (0.02)**	**0.097 (0.02)**	**0.113 (0.02)**	**0.119 (0.03)**	**0.219 (0.04)**	**0.286 (0.05)**	**0.283 (0.05)**	**0.348 (0.06)**
	Backward CD	**1291.10 (163.35)**	**1310.13 (156.10)**	**1265.66 (161.26)**	**1357.47 (186.96)**	**1478.26 (201.44)**	**1868.83 (320.99)**	**1660.78 (241.23)**	**1929.60 (297.81)**	**1855.59 (292.51)**	**2097.29 (341.94)**	**2016.69 (330.92)**	**2330.51 (408.76)**
	Backward EMD	**0.101 (0.001)**	**0.113 (0.01)**	**0.098 (0.01)**	**0.128 (0.02)**	**0.106 (0.02)**	**0.139 (0.02)**	**0.115 (0.02)**	**0.123 (0.02)**	**0.293 (0.05)**	**0.290 (0.05)**	**0.287 (0.05)**	**0.339 (0.06)**
GraphRNN (Benchmark)	Forward CD	1612.91 (268.19)	1784.20 (290.37)	1617.77 (259.37)	1799.52 (307.97)	1720.96 (306.90)	2497.31 (447.89)	1943.24 (356.35)	2522.24 (464.49)	2165.88 (396.31)	3002.74 (563.52)	2766.96 (545.15)	3215.28 (597.98)
	Forward EMD	0.108 (0.02)	0.138 (0.02)	0.115 (0.02)	0.143 (0.02)	0.133 (0.02)	0.147 (0.03)	0.155 (0.03)	0.170 (0.03)	0.346 (0.07)	0.411 (0.08)	0.427 (0.08)	0.514 (0.10)
	Backward CD	1642.66 (280.32)	1735.81 (287.98)	1766.99 (283.90)	1942.32 (330.66)	1830.21 (318.62)	2530.50 (474.90)	2006.11 (368.62)	2738.46 (513.90)	2898.744 (530.49)	3043.36 (568.65)	3192.09 (620.67)	3805.68 (744.47)
	Backward EMD	0.109 (0.02)	0.138 (0.02)	0.115 (0.02)	0.143 (0.03)	0.133 (0.02)	0.147 (0.03)	0.155 (0.03)	0.170 (0.03)	0.346 (0.07)	0.411 (0.08)	0.427 (0.08)	0.514 (0.10)

heterogeneous input data, and (ii) efficient dimensionality reduction and linearization via the Koopman framework.

An ablation study is a technique used to measure the contribution of each component or feature of a model to its overall predictive performance. For the DETONATE model, this consists of removing individual components of the overall loss function such as the consistency and heterogeneous input data loss. By conducting an ablation study on partial DETONATE models, we can determine the importance of each component in achieving high predictive accuracy. This knowledge can then be used to optimize the model by focusing on the most critical components and improving their performance. From the results of the ablation study (partial DETONATE models), we can see the effect of each of the loss terms on the predictive performance. In particular, the performance gains due to the consistency loss and heterogenous input data loss can be summarized in Table 4. Since both metrics CD and EMD have different scales, we first calculate the average percentage gain for each metric and then average across metrics.

When adding the heterogenous input data loss, we see a strong dependence on the simulation setting: for velocity setting *vi)*, which consisted of a linear increase in velocity the addition of the heterogenous data loss leads to a small performance improvement (1.86% for both forward and backward predictions), since this behavior can easily be learned from data. However, for velocity setting vii), which consists of a varying temporal velocity according to a sine wave, we can see a substantial performance gain (10.1% for both forward and backward predictions). This result is highly intuitive: if the 3D shape evolution is strongly influenced by heterogeneous input sources, then integrating this information into the model can yield significant performance improvements. Thus, when utilizing our DETONATE framework for novel applications, we recommend analyzing the 3D shape evolution from a systems perspective. This entails incorporating all relevant parameters that have a substantial impact on the shape evolution into the DETONATE model.

When adding the consistency loss (Partial DETONATE model 3), we observe a modest performance improvement. However, most importantly, the consistency loss is effective in reducing the performance gap between forward and backward predictions and the variance of the predictions. As mentioned earlier, the consistency loss has a regularization effect which can be interpreted as a relaxed version of strict reversibility and stability constraints in physics-informed learning. However, the regularization initially leads to a reduced representation power of the neural network. This explains the modest improvement in average prediction performance.

When all loss terms are combined into a unified framework that is jointly optimized, we fully realize the benefits of both consistency and heterogenous input data loss. In the full DETONATE model, the heterogenous inputs can overcome the reduced representation power of the consistency regularization, and the consistency loss can close the performance gap between forward and backward predictions effectively.

We note that our partial DETONATE model 1 (AE benchmark) demonstrates a performance that is comparable to the GraphRNN benchmark. This similarity arises from the fact that both models do not integrate any heterogeneous input data nor enforce forward-backward prediction consistency. Moreover, the GraphRNN's increased expressivity and incorporation of temporal dependence do not offer a significant advantage over the AE model, due to the small sample size and strong short-term dependence of the 3D shape evolution problem.

Overall, the results of our ablation study provide valuable insights into how the DETONATE model works and which

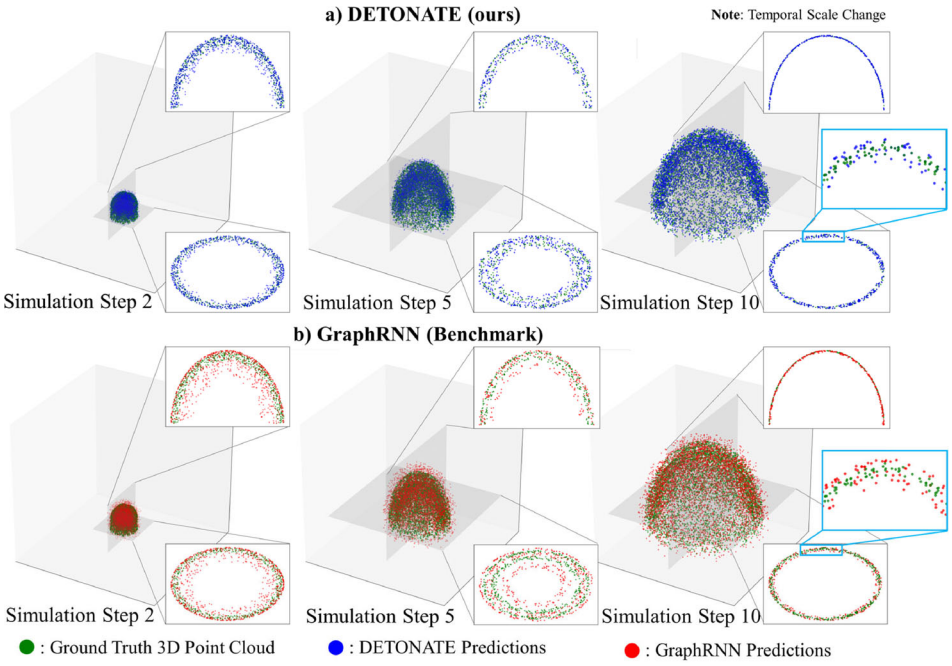
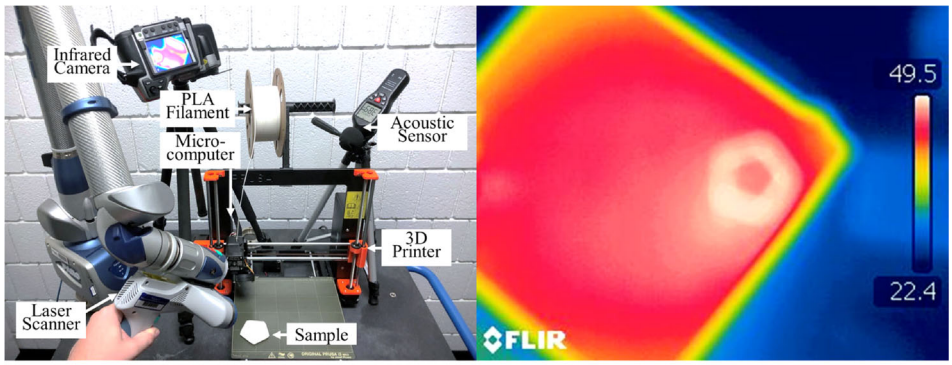


Figure 6. Visualization of prediction results in low noise and constant velocity setting.

Table 4. Ablation study with loss terms of DETONATE model: over Partial DETONATE model 1 (No consistency loss, no heterogenous input data loss).

Addition of Loss term	*Performance Gain (%) Forward Prediction*	*Performance Gain (%) Backward Prediction*
Addition of Heterogenous Input Data Loss (Partial DETONATE model 2)	7.17	6.61
Addition of Consistency Loss (Partial DETONATE model 3)	5.5	5.8
Both Consistency and Heterogenous Input Data Loss (Full DETONATE model)	31.01	27.57



a) Experimental Setup b) Infrared Image

Figure 7. Experimental setup and example of an infrared image.

components are most essential for accurate predictions. This knowledge can help researchers create more effective DETONATE models for a wide range of applications.

5. Case study – additive manufacturing

To study the DETONATE framework described above, we conducted a real-world case study using fused filament fabrication (FFF) of the 3D printing process for the fabrication of Polylactic acid (PLA) specimens. The in-situ shape modeling in additive manufacturing is an essential building block of in-situ functional qualification. Therefore, the modeling of the 3D shape evolvement as a function of heterogeneous process inputs solves a key problem hindering the application of additive manufacturing to safety-critical applications. Existing work in the field of 3D shape modeling focuses on the static case (Huang *et al.*, 2020, Jin *et al.*, 2020; Wang *et al.*, 2022): the 3D shape is modeled *after* it is finished printing to assess the printing quality and informs batch-wise compensation strategies. However, these approaches cannot account for spatial-temporal shape evolvement *during* the printing process. This information is critical for downstream tasks such as real-time feedback control and process optimization.

5.1 Experimental setup

The specimens for the experiments have been printed using a Prusa MK3S printer developed by Prusa Research. The measurement setup is complemented by a FLIR T360 Thermal Imaging Infrared Camera with 1.3 MegaPixel resolution and a FARO Quantum ScanArm with Laser Line Probe. A microcomputer is used to log the nozzle and print bed temperature. An acoustic sensor is installed to collect acoustic emission signals of the process. The experimental setup is visualized in Figure 7(a).

Table 5. Parameter settings for design of experiments.

Process Setting	*Range*
Printing speed	35-100 mm/s
Fan speed	0-100%
Nozzle Temperature	190-240 °C
Print Bed Temperature	40-75 °C
Extrusion width	0.35-0.55 mm

Table 6. Heterogeneous input data description.

Heterogenous Data Source	*Data Type*	*Data Dimension*
Process settings (not monitored):	Tabular	$\mathbb{R}^3$
• Fan Speed: $\mathcal{X}_{h,1}$		
• Extrusion width: $\mathcal{X}_{h,2}$		
• Printing Speed: $\mathcal{X}_{h,3}$		
Nozzle Temperature: $\mathcal{X}_{h,4}^{(t)}$	Functional Curve	$\mathbb{R}^{1000\times L}$
Print Bed Temperature: $\mathcal{X}_{h,5}^{(t)}$	Functional Curve	$\mathbb{R}^{1000\times L}$
Infrared Image: $\mathcal{X}_{h,6}^{(t)}$	Image	$320 \times 240 \times L$

To capture the influence of process parameters on the FDM printing process, a space-filling Latin Hypercube design with $N = 100$ is utilized. The corresponding process parameter ranges are reported in Table 5.

When conducting the experiments, eight experiments failed due to improper process parameter combinations, resulting in 92 samples in total. For each of those experiments, 3D point cloud data is available for each of the $L = 20$ printing layers per part, modeling the layer-wise evolution of the printed parts. Additionally, we recorded in-situ sensing data from six heterogeneous data sources ranging from the three data types tabular, functional curve, and image data. An example of an infrared image obtained at each layer of the printing process is visualized in Figure 7 (b). The heterogeneous data sources along with their typical dimensions are listed in Table 6.

In terms of data preprocessing, the functional curves of the nozzle and bed temperature are fixed to a length of 1000 using dynamic time warping. The point clouds $\mathcal{X}_S^{(t)}$ are up- or down-sampled to a fixed-point number of $N_p = 60,000$ resulting in a

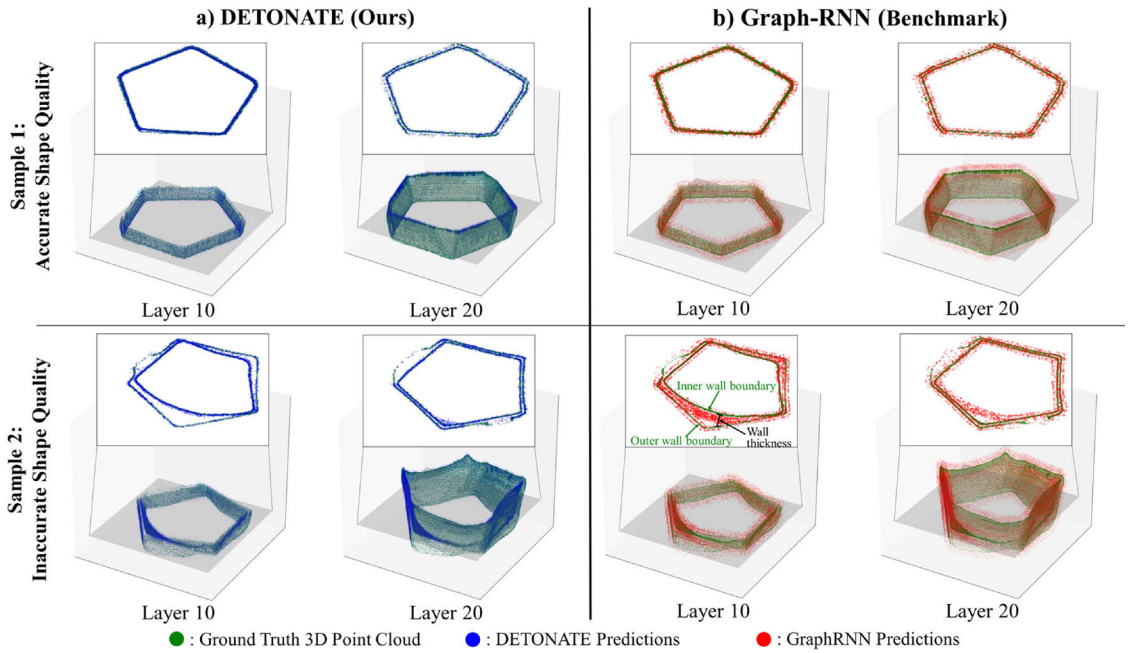


Figure 8. Visualization of prediction results compared to the ground-truth 3D shape.

data dimension $\mathbb{R}^{60,000\times 3\times L}$ for each sample. We determine the number of points based on the following procedure:

1. Determine the minimal point number $N_p^{\min}$
   We utilize a point cloud version of the random sample consensus algorithm (Schnabel et al. 2007), which aims to recover a given shape by randomly drawing a minimal set $N_p^{\min}$ of points. A minimal set is the smallest number of points required to uniquely define a given type of geometry. The resulting candidate shapes are tested against all points in the sample to determine how many points are well approximated.
2. Obtain a conservative number of points N_p
   Then we obtain a conservative number as the fixed-point number $N_p = \lceil 2 \cdot n_p^{\min} \rceil$, where $\lceil \cdot \rceil$ denotes the rounding operation to the next 10,000 increments.
3. Afterward, all the point cloud data are normalized to improve computational efficiency and make it invariant to scaling.

The multistage process (layer-by-layer) of additive manufacturing is dependent on process parameters and in-situ measurements, which provide insight into the physical process of 3D printing. The process parameters, such as nozzle temperature and feed rate, determine the melting, solidification, and cooling of each layer. Therefore, in-situ measurements of these parameters are crucial for understanding the evolution of the 3D shape. The G-code only reflects the desired shape and not the actual process conditions. Previous research has shown that *actual* process measurements can differ significantly from the G-code settings (Anderegg *et al.*, 2019). As such, relying solely on G-code cannot accurately predict the final shape of the printed part. Moreover, the G-code is static for each part and does not consider manufacturing variability. Therefore, it cannot be used for in-situ compensation and control of the 3D printing process.

To model the heterogenous input data, we use the following data-type-specific feature extractors: for the tabular data, we do not employ feature extractors and directly utilize $\mathcal{X}_{h,1}$, $\mathcal{X}_{h,2}$ and $\mathcal{X}_{h,3}$. For the two functional curves (i.e., $\mathcal{X}_{h,4}^{(t)}$, $\mathcal{X}_{h,5}^{(t)}$), we utilize the deep convolutional neural network architecture proposed by Yang *et al.* (2015) in an AE setting. As the feature extractor for the infrared images $\mathcal{X}_{h,6}^{(t)}$, we utilize a convolutional AE structure proposed by Valdarrama (2021).

The DOE (design of experiments) produced samples with varying quality, even though the process limits were chosen within reasonable engineering limits.

5.2 Case study prediction results

We perform one-step-ahead predictions and utilize 10-fold Cross-validation to calculate evaluation metrics. The qualitative prediction results for our method compared to both benchmarks for different layer heights and two different samples with varying quality are visualized in Figure 8. In Figure 8, the green points show the ground-truth printed shape acquired via the FARO laser scanner. The blue points show the predictions of the proposed DETONATE framework. The red points show the predictions of the GraphRNN benchmark. In the first row, we visualize Sample 1, which very closely matched the optimal CAD design of the printed part. We can see that the DETONATE predictions match the shape of the ground truth for different layer heights (i.e., 10 and 20). The GraphRNN on the other hand exhibits a much higher variance. Also, note that the ground-truth consists of an inner and outer wall boundary and the space in between them consists of solid material forming the wall thickness. To better show this structure of the 3D point clouds, we have added additional illustrations in the 2D projection plot of Sample 2, Layer 10 (bottom right quadrant).

However, the predictions of DETONATE are especially useful for in-process quality improvement methodologies

Table 7. Case study prediction results (Standard deviation in brackets (), best method in **bold**).

Method/Metric	*Forward CD*	*Forward EMD*	*Backward CD*	*Backward CD*
Partial DETONATE 1 (no $\mathcal{L}_{con}$, no $\mathcal{L}_h$)	3867.41 (851.62)	2.817 (0.57)	3923.39 (788.99)	2.979 (0.652)
Partial DETONATE 2 (no $\mathcal{L}_{con}$, with $\mathcal{L}_h$)	2654.15 (498.98)	1.811 (0.36)	2842.89 (566.02)	1.913 (0.375)
Partial DETONATE 3 (no $\mathcal{L}_h$, with $\mathcal{L}_{con}$)	3285.52 (496.77)	2.219 (0.35)	3346.65 (506.68)	2.281 (0.386)
DETONATE (ours)	**2394.25 (483.40)**	**1.468 (0.18)**	**2539.04 (498.16)**	**1.590 (0.19)**
Graph-RNN (Benchmark)	3798.25 (694.32)	2.235 (0.32)	3945.02 (744.03)	2.813 (0.41)

Table 8. Ablation study with loss terms of DETONATE model: over Partial DETONATE model 1 (No consistency loss, no heterogenous input data loss).

Addition of Loss term	*Performance Gain (%) Forward Prediction*	*Performance Gain (%) Backward Prediction*
Addition of Heterogenous Input Data Loss (Partial DETONATE model 2)	29.4	35.7
Addition of Consistency Loss (Partial DETONATE model 3)	14.8	22.2
Both Consistency and Heterogenous Input Data Loss (Full DETONATE model)	36.7	47.2

(Shi, 2022) and related downstream tasks (e.g., compensation and control), if we can predict accurate shapes for a wide range of manufacturing quality levels. Therefore, in the second row, we visualize Sample 2, which exhibited a large shape distortion (i.e., bad printing quality). We can see that even for large distortions, the DETONATE predictions closely match the ground-truth shape profile. The GraphRNN on the other hand exhibits a large variance. Even though subsequent layer shapes are determined by previous layers' shapes, the benchmark method fails to capture the accurate 3D shape profiles of the printed samples for both small (Sample 1) and large (Sample 2) shape distortion.

To measure prediction quality, we utilize the CD and the EMD, to measure the point cloud-to-point cloud distance between ground-truth 3D shapes and predicted 3D shapes. The predicted results calculated using 10-fold cross-validation are summarized in Table 7. The total sample size was 92 parts. Therefore, the results from cross-validation are the average of the test set (nine parts) for 10 non-overlapping folds (an average of 90 parts in total). The point clouds are normalized making them invariant of the changing scaling across time.

For all evaluation metrics, the proposed DETONATE framework outperforms the Graph-RNN baseline. It is worth noting that the modeling of the consistent forward and backward dynamic significantly improves the prediction accuracy compared to the Graph-RNN. When modeling only the forward dynamics in the presence of small sample sizes, the 3D shape predictions of the Graph-RNN behave more like a global sample average prediction rather than modeling the exact shape of each sample. DETONATE demonstrates a better preservation of the spatial structure over time. This is a direct effect of the consistent forward and backward dynamics as well as its ability to consider heterogeneous data inputs.

Through an ablation study on partial DETONATE models, we can gain insights into the impact of each model component on the predictive performance. In particular, we can summarize the performance gains resulting from the consistency loss and heterogeneous input data loss as show in Table 8.

By comparing the performance of different partial DETONATE models, we can observe that the inclusion of heterogeneous input data in partial DETONATE model 2 leads to a significant performance gain. The 3D shape evolution of the next layer is directly influenced by the current process conditions and bonding with the previous layers, and the heterogeneous input data is strongly correlated with this evolution. Although the addition of the consistency loss is also beneficial, its impact is less pronounced compared to the heterogeneous input data.

However, the combination of both loss terms in the full DETONATE model results in a significant advantage over partial DETONATE model 1, which only includes the forward, backward, and reconstruction losses. This highlights the importance of incorporating both heterogeneous input data and consistency loss for achieving optimal performance.

In particular, the full potential of the consistency loss is more apparent in the full DETONATE model, as it enables bi-directional learning and leads to remarkable performance gains. This underscores the importance of considering both the heterogenous input data and temporal consistency in the DETONATE model design for achieving the best predictive accuracy.

6. Conclusion

In this article, we proposed a novel framework for the modeling of nonlinear dynamically evolving 3D shape profiles. Our framework is based on Koopman operator theory as we approximate the nonlinear dynamical system via linear evolvement matrices. It allows the end-to-end learning of dynamic, unstructured 3D point clouds and accurate predictions for future as well as past temporal observations. Key to our approach is the integration of heterogeneous input data in a holistic framework and the consideration of forward and backward dynamics while requiring temporal consistency. We evaluate our method on challenging datasets and compare our approach with a state-of-the-art point cloud prediction network.

A limitation of the current framework is the limited ability to capture complex geometric and boundary conditions. Our model does not have explicit physics priors; it learns approximate physics knowledge directly from data. This generally prevents it from learning exact physical laws. Furthermore, if the 3D profiles are extremely discontinuous, the model may fail to capture those dynamics. In this case, multiple models for different temporal segments may be necessary. Existing works on physics-informed Koopman models could serve as

a starting point to extend DETONATE to incorporate boundary and geometric constraints.

We believe our work can be extended in many ways, from a methodology and application perspective. Specifically, our work has the potential to be extended to several areas: (i) process monitoring for 3D shape quality, (ii) root-cause analysis, (iii) Uncertainty quantification of the DETONATE model, (iv) in-process control and compensation of 3D profiles, (v) manufacturing process design for variation reduction, and (vi) multi-step ahead prediction that consider both short- and long-term dependencies.

Acknowledgments

We would like to express our gratitude to the three anonymous referees whose insightful comments and suggestions have significantly enhanced the quality of our manuscript.

Funding

We acknowledge the generous support provided by the National Science Foundation (NSF) under Award Number 2019378.

Notes on contributors

Michael Biehler received his BS and MS degrees in industrial engineering with a major in production engineering from the Karlsruhe Institute of Technology (KIT) in 2017 and 2020, respectively. He is currently pursuing a PhD degree with the H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology. His research rests at the interface between 3D machine learning and cyber-physical security, where he aims to develop methods for monitoring, prognostics, and control.

Daniel Lin is a junior at Walton High School in Marietta, GA. He is interested in applying machine learning to a wide range of applications such as advanced manufacturing and biology and plans to pursue a career in science or engineering in the future.

Dr. Jianjun Shi received the BS and MS degrees in automation from the Beijing Institute of Technology in 1984 and 1987, respectively, and the PhD degree in mechanical engineering from the University of Michigan in 1992. Currently, he is the Carolyn J. Stewart Chair and a Professor at the H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology. His research interests include the fusion of advanced statistical and domain knowledge to develop methodologies for modeling, monitoring, diagnosis, and control of complex manufacturing systems. He is a fellow of four professional societies, including ASME, IISE, INFORMS, and SME, an Elected Member of the International Statistics Institute (ISI), a Life Member of ASA, an Academician of the International Academy for Quality (IAQ), and a member of the National Academy of Engineers (NAE).

References

Abraham, I. and Murphey, T.D. (2019) Active learning of dynamics for data-driven control using Koopman operators. *IEEE Transactions on Robotics*, **35**(5), 1071–1083.

Anderegg, D.A., Bryant, H.A., Ruffin, D.C., Skrip Jr, S.M. Fallon, J.J., Gilmer, E.L. and Bortner, M.J. (2019) In-situ monitoring of polymer flow temperature and pressure in extrusion based additive manufacturing. *Additive Manufacturing*, **26**, 76–83.

Arbabi, H. and Mezic, I. (2017) Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the Koopman operator. *SIAM Journal on Applied Dynamical Systems*, **16**(4), 2096–2126.

Azencot, O., Erichson, N.B., Lin, V. and Mahoney, M. (2020) Forecasting sequential data using consistent Koopman autoencoders, in *International Conference on Machine Learning*, pp. 475–485.

Azencot, O., Yin, W. and Bertozzi, A. (2019) Consistent dynamic mode decomposition. *SIAM Journal on Applied Dynamical Systems*, **18**(3), 1565–1585.

Bevanda, P., Sosnowski, S. and Hirche, S. (2021) Koopman operator dynamical models: Learning, analysis and control. *Annual Reviews in Control*, 52, 197–212.

Biehler, M., Yan, H. and Shi, J. (2022) ANTLER: Bayesian nonlinear tensor learning and modeler for unstructured, varying-size point cloud data. *IEEE Transactions on Automation Science and Engineering*, 1–14

Bot, R.I. and Csetnek, E.R. (2016) Second order forward-backward dynamical systems for monotone inclusion problems. *SIAM Journal on Control and Optimization*, **54**(3), 1423–1443.

Brunton, S.L., Brunton, B.W., Proctor, J.L. and Kutz, J.N. (2016) Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. *PloS One*, **11**(2), 1–19.

Brunton, S.L., Budišić, M., Kaiser, E. and Kutz, J.N. (2021) Modern Koopman theory for dynamical systems. *arXiv preprint arXiv: 2102.12086*, pp.1–110.

Chen, R., Yang, D. and Zhang, C.-H. (2022) Factor models for high-dimensional tensor time series. *Journal of the American Statistical Association*, **117**(537), 94–116.

Dogra, A.S. and Redman, W. (2020) Optimizing neural networks via Koopman operator theory. *Advances in Neural Information Processing Systems*, **33**, 2087–2097.

Dunford, N. and Schwartz, J.T. (1971) *Linear Operators Part I, Volume VII of Pure and Applied Mathematics*, Interscience Publishers, Geneva, Switzerland.

Durbin, J. and Koopman, S.J. (2012) *Time Series Analysis by State Space Methods*, Oxford University Press (OUP), Oxford, UK.

Eisner, T., Farkas, B., Haase, M. and Nagel, R. (2015) *Operator Theoretic Aspects of Ergodic Theory*, Springer, Cham.

Elman, J.L. (1990) Finding structure in time. *Cognitive Science*, **14**(2), 179–211.

Fan, H., Su H. and Guibas, L.J. (2017) A point set generation network for 3d object reconstruction from a single image, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE Press, Piscataway, NJ, pp. 605–613.

Fan, H. and Yang, Y. (2019) PointRNN: Point recurrent neural network for moving point cloud processing. *arXiv preprint arXiv: 1910.08287*.

Gahrooei, M.R., Yan, H., Paynabar, K. and Shi, J. (2021) Multiple tensor-on-tensor regression: An approach for modeling processes with heterogeneous sources of data. *Technometrics*, **63**(2), 147–159.

Gardini, L., Hommes, C., Tramontana, F. and De Vilder, R. (2009) Forward and backward dynamics in implicitly defined overlapping generations models. *Journal of Economic Behavior & Organization*, **71**(2), 110–129.

Girgis, A.M., Seo, H., Park, J., Bennis, M. and Choi, J. (2022) Predictive closed-loop remote control over wireless two-way split Koopman autoencoder. *IEEE Internet of Things Journal*, **9**(23), 23285–23301.

Gomes, P., Rossi, S. and Toni, L. (2021) Spatio-temporal graph-RNN for point cloud prediction, in *2021 IEEE International Conference on Image Processing (ICIP)*, IEEE Press, Piscataway, NJ, pp. 3428–3432.

Guo, Y., Shi, H., Kumar, A., Grauman, K., Rosing, T. and Feris, R. (2019) Spottune: transfer learning through adaptive fine-tuning, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE Press, Piscataway, NJ, pp. 4805–4814.

Gurney, R.W. (1943) The initial velocities of fragments from bombs, shell and grenades, Army Ballistic Research Lab (BRL), Aberdeen Proving Ground, Maryland, pp. 1–10

Han, Y., Hao, W. and Vaidya, U. (2020) Deep learning of Koopman representation for control, in *2020 59th IEEE Conference on Decision and Control (CDC)*, IEEE Press, Piscataway, NJ, pp. 1890–1895.

Haseli, M. and Cortés, J. (2019) Approximating the Koopman operator using noisy data: Noise-resilient extended dynamic mode

decomposition, in *2019 American Control Conference (ACC)*, IEEE Press, Piscataway, NJ, pp. 5499–5504.

Haseli, M. and Cortés, J. (2022) Temporal forward–backward consistency, not residual error, measures the prediction accuracy of extended dynamic mode decomposition. *IEEE Control Systems Letters*, **7**, 649–654.

He, L. and Zhang, L. (2013) Dynamic priority rule-based forward-backward heuristic algorithm for resource levelling problem in construction project. *Journal of the Operational Research Society*, **64**(8), 1106–1117.

Hoffman, J., Tzeng, E., Park,T., Zhu, J.Y., Isola, P., Saenko, K., Efros, A. and Darrell, T. (2018) Cycada: Cycle-consistent adversarial domain adaptation, in *International Conference on Machine Learning*, Proceedings of Machine Learning Research (PMLR), Stockholm, Sweden, pp. 1989–1998.

Huang, Q., Wang, F. and Guibas, L. (2014) Functional map networks for analyzing and exploring large shape collections. *ACM Transactions on Graphics (ToG)*, **33**(4), 1–11.

Huang, Q., Wang, Y., Lyu, M. and Lin, W. (2020) Shape deviation generator—a convolution framework for learning and predicting 3-D printing shape accuracy. *IEEE Transactions on Automation Science and Engineering*, **17**(3), 1486–1500.

Jin, Y., Qin, S.J. and Huang, Q. (2020) Modeling inter-layer interactions for out-of-plane shape deviation reduction in additive manufacturing. *IISE Transactions*, **52**(7), 721–731.

Jones, D.R. (2001) A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, **21**(4), 345–383.

Koopman, B.O. (1931) Hamiltonian systems and transformation in Hilbert space. *Proceedings of the National Academy of Sciences*, **17**(5), 315–318.

Krim, H. and Yezzi, A.J. (2006) *Statistics and Analysis of Shapes*, Birkhäuser, Boston, USA.

Landrieu, L. and Simonovsky, M. (2018) Large-scale point cloud semantic segmentation with superpoint graphs, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE Press, Piscataway, NJ, pp. 4558–4567.

Lange, H., Brunton, S.L. and Kutz, J.N. (2021) From Fourier to Koopman: Spectral methods for long-term time series prediction. *Journal of Machine Learning Research*, **22**(41), 1–38.

Le Clainche, S. and Vega, J.M. (2017) Higher order dynamic mode decomposition. *SIAM Journal on Applied Dynamical Systems*, **16**(2), 882–925.

Lusch, B., Kutz, J.N. and Brunton, S.L. (2018) Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, **9**(1), 1–10.

Moahmed, T.A., Gayar, N.E. and Atiya, A.F. (2014) Forward and backward forecasting ensembles for the estimation of time series missing data, in *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*, Springer, Montreal, Canada, pp. 93–104.

Mulekar, O.S., Bevilacqua, R., Jerome, E.L. and Hatch-Aguilar, T.J. (2021) Transfer function to predict warhead fragmentation in-flight behavior from static data. *AIAA Journal*, **59**(11), 4777–4793.

Nayak, I., Teixeira, F.L. and Kumar, M. (2021) Koopman autoencoder architecture for current density modeling in kinetic plasma simulations, in *2021 International Applied Computational Electromagnetics Society Symposium (ACES)*, IEEE Press, Piscataway, NJ, pp. 1–3.

Otto, S.E. and Rowley, C.W. (2019) Linearly recurrent autoencoder networks for learning dynamics. *SIAM Journal on Applied Dynamical Systems*, **18**(1), 558–593.

Pan, S., Arnold-Medabalimi, N. and Duraisamy, K. (2021) Sparsity-promoting algorithms for the discovery of informative Koopman-invariant subspaces. *Journal of Fluid Mechanics*, **917**, 1–49.

Qi, C.R., Su, H., Mo, K. and Guibas, L.J. (2017) Pointnet: Deep learning on point sets for 3d classification and segmentation, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE Press, Piscataway, NJ, pp. 652–660.

Rubner, Y., Tomasi, C. and Guibas, L.J. (2000) The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, **40**(2), 99–121.

Salova, A., Emenheiser, J., Rupe, A., Crutchfield, J.P. and D'Souza, R.M. (2019) Koopman operator and its approximations for systems with symmetries. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, **29**(9), 1–16.

Schmid, P.J. (2010) Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, **656**, 5–28.

Schnabel, R., Wahl, R. and Klein, R. (2007) Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum*, **26**(2), 214–226.

Shi, J. (2022) In-process quality improvement: Concepts, methodologies, and applications. *IISE Transactions*, **55**(1), 2–21

Shi, Q., Yin, J., Cai, J., Cichocki, A., Yokota, T., Chen, L., Yuan, M. and Zeng, J. (2020) Block Hankel tensor ARIMA for multiple short time series forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(4), pp. 5758–5766.

Singh, R.K. and Manhas, J.S.(1993) *Composition Operators on Function Spaces*, Elsevier, Amsterdam, The Netherlands.

Snoek, J., Larochelle, H. and Adams, R.P. (2012) Practical Bayesian optimization of machine learning algorithms. *Advances in Neural Information Processing Systems*, **25**, 1–9.

Surana, A. (2020) Koopman operator framework for time series modeling and analysis. *Journal of Nonlinear Science*, **30**(5), 1973–2006.

Takeishi, N., Kawahara, Y. and Yairi, T. (2017) Learning Koopman invariant subspaces for dynamic mode decomposition. *Advances in Neural Information Processing Systems*, **30**, 1–11.

Tu, J.H., Rowley, C.W., Kutz, J.N. and Shang, J.K. (2014) Spectral analysis of fluid flows using sub-Nyquist-rate PIV data. *Experiments in Fluids*, **55**(9), 1–13.

Valdarrama, S. (2021) Convolutional autoencoder for image denoising from https://keras.io/examples/vision/autoencoder/, (accessed 1 January 2023).

Wang, D., Zheng, Y., Lian, H. and Li, G. (2021) High-dimensional vector autoregressive time series modeling via tensor decomposition. *Journal of the American Statistical Association*, **117**(539), 1338–1356.

Wang, H., Bai, C., Feng, C., Xue, K. and Zhu, X. (2019) An efficient CDEM-based method to calculate full-scale fragment field of warhead. *International Journal of Impact Engineering*, **133**, 1–16.

Wang, Y., Ruiz, C. and Huang, Q. (2022) Learning and predicting shape deviations of smooth and non-smooth 3D geometries through mathematical decomposition of additive manufacturing. *IEEE Transactions on Automation Science and Engineering*, Early Access, 1–12.

Williams, M.O., Kevrekidis, I.G. and Rowley, C.W. (2015) A data–driven approximation of the Koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, **25**(6), 1307–1346.

Yan, S., Yang, Z., Li, H., Guan, L., Kang, H., Hua, G. and Huang, Q. (2022) Implicit autoencoder for point cloud self-supervised representation learning. *arXiv preprint arXiv:2201.00785*.

Yang, J., Nguyen, M.N., San, P.P., Li, X.L. and Krishnaswamy, S. (2015) Deep convolutional neural networks on multichannel time series for human activity recognition, in *Twenty-fourth International Joint Conference on Artificial Intelligence*, AAAI Press, Buenos Aires, Argentina, pp. 3995–4001.

Yang, Y., Feng, C., Shen, Y. and Tian, D. (2018) Foldingnet: Point cloud auto-encoder via deep grid deformation, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE Press, Piscataway, NJ, pp. 206–215.

Zhao, Y., Ye, L., Li, Z., Song, X., Lang, Y. Su, J. (2016) A novel bidirectional mechanism based on time series model for wind power forecasting. *Applied Energy*, **177**, 793–803.

Zheng, Y. and Cheng, G. (2021) Finite-time analysis of vector autoregressive models under linear restrictions. *Biometrika*, **108**(2), 469–489.

Zhou, Y. and Tuzel, O. (2018) Voxelnet: End-to-end learning for point cloud based 3d object detection, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE Press, Piscataway, NJ, pp.4490–4499.

Zhu, J.-Y., Park, T., Isola, P. and Efros, A.A. (2017) Unpaired image-to-image translation using cycle-consistent adversarial networks, in *Proceedings of the IEEE International Conference on Computer Vision*, IEEE Press, Piscataway, NJ, pp. 2223–2232.