Physics-Integrated Variational Autoencoders for Robust and Interpretable Generative Modeling

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Integrating physics models within machine learning models holds considerable promise toward learning robust models with improved interpretability and abilities to extrapolate. In this work, we focus on the integration of incomplete physics models into deep generative models. In particular, we introduce an architecture of variational autoencoders (VAEs) in which a part of the latent space is grounded by physics. A key technical challenge is to strike a balance between the incomplete physics and trainable components such as neural networks for ensuring that the physics part is used in a meaningful manner. To this end, we propose a regularized learning method that controls the effect of the trainable components and preserves the semantics of the physics-based latent variables as intended. We not only demonstrate generative performance improvements over a set of synthetic and real-world datasets, but we also show that we learn robust models that can consistently extrapolate beyond the training distribution in a meaningful manner. Moreover, we show that we can control the generative process in an interpretable manner.

1 Introduction

Data-driven modeling is often opposed to theory-driven modeling, yet their integration has also been recognized as an important approach known as *gray-box* or *hybrid* modeling. In statistical machine learning, incorporation of physics (in a broad sense; including knowledge of biology, economics, etc.) has also been attracting attention. Gray-box / hybrid modeling in machine learning holds considerable promise toward learning robust models with improved abilities to extrapolate beyond the distributions that they have been exposed to during training. Moreover, it can bring significant benefits in terms of model interpretability since parts of a model get semantically grounded to concrete inductive bias.

A technical challenge in gray-box deep modeling is to ensure an appropriate use of physics models. A careless design of models and learning can lead to an erratic behavior of the components meant to represent physics (e.g., with erroneous estimation of physics parameters), and eventually, the overall gray-box model just learns to ignore them. This is particularly the case when we bring together simplified or imperfect physics models with very expressive data-driven machine learning models such as deep neural networks. Such cases call for principled methods for striking an appropriate balance between physics and data-driven models to prevent the detrimental effects during learning.

Integration of physics models into machine learning has been considered in various contexts (see, e.g., [41, 38] and our Section 4), but most existing studies focus on prediction or forecasting tasks and are not directly applicable to other tasks. More importantly, hardly any have addressed the careful orchestration of physics-based and data-driven components to avoid the detrimental effects. A notable exception is Yin et al. [44], in which they proposed a method to harness the action of trainable components of a hybrid model of differential equations. Their method has been developed for dynamics forecasting and is limited to additive combinations of physics and trainable models.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

In this work, we aim at the integration of (incomplete) physics models into deep generative models, variational autoencoders (VAEs, [15, 28]) in particular, while the basic idea is applicable to other models. In our VAE, the decoder comprises physics-based models and trainable neural networks, and some of the latent variables are semantically grounded to the parameters of the physics models. Such a VAE, if appropriately trained, is by construction partly interpretable. Moreover, since it can by construction capture the underlying physics, it will be robust in out-of-distribution regime and exhibit meaningful extrapolation properties. We propose a regularized learning framework for ensuring the meaningful use of the physics models and the preservation of the semantics of the latent variables in the physics-integrated VAEs. We empirically demonstrate that our method can learn a model that exhibits better generalization, and more importantly, can extrapolate robustly in out-of-distribution regime. In addition, we show how the direct access to the physics-grounded latent variables allows us to alter properties of generation meaningfully and explore counterfactual scenarios.

2 Physics-integrated VAEs

We first describe the structure of VAEs we consider, which comprise physics models and machine learning models such as neural nets. We suppose that the physics models can be solved analytically or numerically with a reasonable cost, and the (approximate) solution is differentiable with regard to the quantities on which the solution depends. This assumption holds in most physics models known in practice, which come in different forms such as algebraic and differential equations. If there is no closed-form solution of algebraic equations, we can utilize differentiable optimizers [3] as a layer of the model. For differential equations, differentiable integrators [see, e.g., 7] will constitute a layer. Handling non-differentiable and/or overly-complex simulators remains an important open challenge.

2.1 Example

We start with an example to demonstrate the main concepts. Let us suppose that data comprise time-series of the angle of pendulums following an ordinary differential equation (ODE):

$$\underbrace{\mathrm{d}^2\theta(t)/\mathrm{d}t^2 + \omega^2 \sin\theta(t)}_{\text{given as prior knowledge, } f_\mathrm{P}} + \underbrace{\zeta \mathrm{d}\theta(t)/\mathrm{d}t - u(t)}_{\text{to be learned by NN, } f_\mathrm{A}} = 0, \tag{1}$$

where θ is the pendulum's angle, and ω, ζ, and u are the pendulum's angular velocity, damping coefficient, and external force, respectively. We suppose that a data point x is a sequence of $\theta(t)$, i.e., $x = [\theta(0)\ \theta(\Delta t)\ \cdots\ \theta((\tau-1)\Delta t)]^\mathsf{T} \in \mathbb{R}^\tau$ for some $\Delta t \in \mathbb{R}$ and $\tau \in \mathbb{N}$, where $\theta(t)$ is the solution of (1) with a particular configuration of ω, ζ, and u. In this example, we learn a VAE on a dataset comprising such x with different configurations of ω, ζ, and u.

Suppose that the first two terms of (1) are given as prior knowledge, i.e., we know that the governing equation should contain $f_\mathrm{P}(\theta, \omega) := \ddot{\theta} + \omega^2 \sin\theta$. We will use such prior knowledge, f_P, by incorporating it in the decoder of the VAE. Since f_P misses some effects of the true pendulum system (1), we complete it by augmenting the decoder with an auxiliary function $f_\mathrm{A}(\theta, \boldsymbol{z}_\mathrm{A})$, which we model with a neural network. The VAE's latent variable will have two parts, z_P and $\boldsymbol{z}_\mathrm{A}$, respectively linked to f_P and f_A. One one hand, $\boldsymbol{z}_\mathrm{A}$ works as an ordinary VAE's latent variable since f_A is a neural net, and we suppose $\boldsymbol{z}_\mathrm{A} \in \mathbb{R}^d$, $p(\boldsymbol{z}_\mathrm{A}) := \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. On the other hand, we semantically ground z_P to the parameter of f_P, that is, $z_\mathrm{P} := \omega \in \mathbb{R}$ in this example. In summary, the augmented decoder here is $\mathbb{E}[x] = \mathrm{ODEsolve}_\theta \left[f_\mathrm{P}(\theta(t), z_\mathrm{P}) + f_\mathrm{A}(\theta(t), \boldsymbol{z}_\mathrm{A}) = 0 \right]$, where $\mathrm{ODEsolve}_\theta$ denotes some differentiable solver of an ODE with regard to θ. The encoder will have corresponding recognition networks for z_P and $\boldsymbol{z}_\mathrm{A}$. The situation in this example will be numerically examined in Section 5.1.

2.2 General formulation

We now present the concept of our physics-integrated VAEs in a general form. Note that our interest is not limited to additive cases nor ODEs. In fact, the general formulation below subsumes non-additive augmentation of various physics models (i.e., not only ODEs). The notation introduced in this section will be used to explain the proposed regularized learning method later in Section 3.

For clarity, we suppose that a VAE decoder comprises two parts: a physics-based model f_P and a trainable auxiliary function f_A. More general cases, for example with multiple trainable functions $f_{\mathrm{A},1}, f_{\mathrm{A},2}, \ldots$ used in different ways, are handled in Appendix A.

2.2.1 Latent variables and priors

We consider two types of latent variables, $\boldsymbol{z}_\mathrm{P} \in \mathcal{Z}_\mathrm{P}$ and $\boldsymbol{z}_\mathrm{A} \in \mathcal{Z}_\mathrm{A}$, which respectively will be used in f_P and f_A. The latent variables can be in any space, but for simplicity of discussion, we suppose $\mathcal{Z}_\mathrm{P}$ and $\mathcal{Z}_\mathrm{A}$ are (subsets of) the Euclidean space and set their prior distribution as multivariate normal:

$$p(\boldsymbol{z}_\mathrm{P}) \coloneqq \mathcal{N}(\boldsymbol{z}_\mathrm{P} \mid \boldsymbol{m}_\mathrm{P}, v_\mathrm{P}^2 \boldsymbol{I}) \quad \text{and} \quad p(\boldsymbol{z}_\mathrm{A}) \coloneqq \mathcal{N}(\boldsymbol{z}_\mathrm{A} \mid \boldsymbol{0}, \boldsymbol{I}), \tag{2}$$

where $\boldsymbol{m}_\mathrm{P}$ and v_P^2 are defined in accordance with prior knowledge of f_P's parameters. Note that $\boldsymbol{z}_\mathrm{P}$ will be directly interpretable as they will be semantically grounded to the parameters of the physics model f_P; for example in Section 2.1, $z_\mathrm{P} \coloneqq \omega$ was the angular velocity of a pendulum.

2.2.2 Decoder

The decoder of a physics-integrated VAE comprises two types of functions[1], $f_\mathrm{P} \colon \mathcal{Z}_\mathrm{P} \to \mathcal{Y}_\mathrm{P}$ and $f_\mathrm{A} \colon \mathcal{Y}_\mathrm{P} \times \mathcal{Z}_\mathrm{A} \to \mathcal{Y}_\mathrm{A}$. For notational convenience, we consider a functional $\mathcal{F}$ that evaluates f_P and f_A, solve an equation if any, and finally gives observation $\boldsymbol{x} \in \mathcal{X}$. $\mathcal{X}$ may be the space of sequences, images, and so on. Assuming Gaussian observation noise, we write the observation model as

$$p_\theta(\boldsymbol{x} \mid \boldsymbol{z}_\mathrm{P}, \boldsymbol{z}_\mathrm{A}) \coloneqq \mathcal{N}\big(\boldsymbol{x} \mid \mathcal{F}[f_\mathrm{A}(f_\mathrm{P}(\boldsymbol{z}_\mathrm{P}), \boldsymbol{z}_\mathrm{A})], \boldsymbol{\Sigma}_x\big). \tag{3}$$

Note that f may have other arguments besides $\boldsymbol{z}$, but they are omitted for simplicity. We denote the set of trainable parameters of f_A and f_P (and $\boldsymbol{\Sigma}_x$) by θ, while f_P may have no trainable parameters.

Let us see the semantics of $\mathcal{F}$ first in the light of the example of Section 2.1. Recall that there we considered the additive augmentation of ODE (as in [44] and other studies). It is subsumed by the expression (3) by setting $f_\mathrm{A}(f_\mathrm{P}(\boldsymbol{z}_\mathrm{P}), \boldsymbol{z}_\mathrm{A}) \coloneqq f_\mathrm{P}(\boldsymbol{z}_\mathrm{P}) + f_{\mathrm{A}'}(\boldsymbol{z}_\mathrm{A})$ and $\mathcal{F}[f] \coloneqq \mathrm{ODEsolve}[f = 0]$, where $f_{\mathrm{A}'}$ is a neural network. Let us generalize the idea. Our definition of the decoder in (3) allows not only additive augmentation of ODE but also broader range of architectures. The composition of f_P and f_A is *not* limited to be additive because we consider general function composition $f_\mathrm{A}(f_\mathrm{P}(\boldsymbol{z}_\mathrm{P}), \boldsymbol{z}_\mathrm{A})$. Moreover, the form of the physics model is *not* limited to ODEs:

- If equation $f_\mathrm{P} = 0$ has a closed-form solution S_{f_P}, then $\mathcal{F}$ is simply, e.g., $\mathcal{F}[f_\mathrm{P}, f_\mathrm{A}] \coloneqq f_\mathrm{A}(S_{f_\mathrm{P}})$.
- If an algebraic equation $f_\mathrm{P} = 0$ or $f_\mathrm{A} \circ f_\mathrm{P} = 0$ has no closed-form solution, then $\mathcal{F}$ will have a differentiable optimizer, e.g., $\mathcal{F}[f_\mathrm{P}, f_\mathrm{A}] \coloneqq f_\mathrm{A}(\arg\min \|f_\mathrm{P}\|^2)$ or $\mathcal{F} \coloneqq \arg\min \|f_\mathrm{A} \circ f_\mathrm{P}\|^2$.
- $f_\mathrm{P} = 0$ or $f_\mathrm{A} \circ f_\mathrm{P} = 0$ can be a stochastic differential equation (and $\mathcal{F}$ contains its solver), for which $\boldsymbol{z}_\mathrm{P}$ and/or $\boldsymbol{z}_\mathrm{A}$ would become a sequence encoding the realization of the process noise.

The role of f_A can also be diverse; it can work not only as a complement of physics models inside equations, but also as correction of numerical errors of solvers or optimizers, downsampling or upsampling, and observables (e.g., from angle sequence to video of a pendulum).

2.2.3 Encoder

The encoder of a physics-integrated VAE accordingly comprises two parts: posterior inference of $\boldsymbol{z}_\mathrm{P}$ and that of $\boldsymbol{z}_\mathrm{A}$. We consider the following decomposition of the approximated posterior:

$$\begin{aligned} &q_\psi(\boldsymbol{z}_\mathrm{P}, \boldsymbol{z}_\mathrm{A}, \mid \boldsymbol{x}) \coloneqq q_\psi(\boldsymbol{z}_\mathrm{A} \mid \boldsymbol{x}) q_\psi(\boldsymbol{z}_\mathrm{P} \mid \boldsymbol{x}, \boldsymbol{z}_\mathrm{A}), \\ \text{where} \quad &q_\psi(\boldsymbol{z}_\mathrm{A} \mid \boldsymbol{x}) \coloneqq \mathcal{N}\big(\boldsymbol{z}_\mathrm{A} \mid g_\mathrm{A}(\boldsymbol{x}), \boldsymbol{\Sigma}_\mathrm{A}\big), \quad q_\psi(\boldsymbol{z}_\mathrm{P} \mid \boldsymbol{x}, \boldsymbol{z}_\mathrm{A}) \coloneqq \mathcal{N}\big(\boldsymbol{z}_\mathrm{P} \mid g_\mathrm{P}(\boldsymbol{x}, \boldsymbol{z}_\mathrm{A}), \boldsymbol{\Sigma}_\mathrm{P}\big). \end{aligned} \tag{4}$$

$g_\mathrm{A} \colon \mathcal{X} \to \mathcal{Z}_\mathrm{A}$ and $g_\mathrm{P} \colon \mathcal{X} \times \mathcal{Z}_\mathrm{A} \to \mathcal{Z}_\mathrm{P}$ are recognition networks. We denote the trainable parameters of g_A and g_P (and $\boldsymbol{\Sigma}_\mathrm{A}$ and $\boldsymbol{\Sigma}_\mathrm{P}$) as ψ. This particular dependency is for our regularization method in Section 3.2, where g_P should first remove the information of $\boldsymbol{z}_\mathrm{A}$ from $\boldsymbol{x}$ and then infer $\boldsymbol{z}_\mathrm{P}$.

2.3 Evidence lower bound

The VAE is to be learned as usual by maximizing the lower bound of the marginal log likelihood known as evidence lower bound (ELBO). In our case, it is straightforward to derive:

$$\begin{aligned} \mathrm{ELBO}(\theta, \psi; \boldsymbol{x}) = &\ \mathbb{E}_{q_\psi(\boldsymbol{z}_\mathrm{P}, \boldsymbol{z}_\mathrm{A} \mid \boldsymbol{x})} \log p_\theta(\boldsymbol{x} \mid \boldsymbol{z}_\mathrm{P}, \boldsymbol{z}_\mathrm{A}) \\ &+ D_\mathrm{KL}\big[q_\psi(\boldsymbol{z}_\mathrm{A} \mid \boldsymbol{x}) \parallel p(\boldsymbol{z}_\mathrm{A})\big] + \mathbb{E}_{q_\psi(\boldsymbol{z}_\mathrm{A} \mid \boldsymbol{x})} D_\mathrm{KL}\big[q_\psi(\boldsymbol{z}_\mathrm{P} \mid \boldsymbol{x}, \boldsymbol{z}_\mathrm{A}) \parallel p(\boldsymbol{z}_\mathrm{P})\big]. \end{aligned} \tag{5}$$

[1]The distinction between f_P and f_A depends on the origin of the functional forms (and not if trainable or not). The form of f_P depends on physics' insight and thus fixed. On the other hand, the form of f_A is determined only from utility as a function appoximator, and we can use whatever useful (e.g., feed-forward NNs, RNNs, etc.).

3 Regularizing physics-integrated VAEs

We propose a regularized learning objective for physics-integrated VAEs. It comprises two types of regularizers. The first one is for harnessing unnecessary flexibility of function approximators like neural networks and presented in Section 3.1. The second ones are for grounding encoder's output to physics parameters and presented in Section 3.2. The overall objective is summarized in Section 3.3.

3.1 Harnessing trainable functions by PPC-like procedure

If the trainable component of the physics-integrated VAE (i.e., f_A) has rich expression capability, as is often the case with deep neural networks, merely maximizing the ELBO in (5) provides no guarantee that the physics-based component (i.e., f_P) will be used in a meaningful manner; e.g., f_P may just be ignored. We want to ensure that f_A does not unnecessarily dominate the behavior of the entire model and that f_P is not ignored. To this end, we borrow an idea from the *posterior predictive check* (PPC), a procedure to check the validity of a statistical model [see, e.g., 9]. Whereas the standard PPCs examine the discrepancy between model's and data's posterior predictive distributions, we compute the discrepancy between those of the physics-integrated model and its "physics-only" reduced model for monitoring and balancing the contributions of parts of the model.

For the sake of argument, suppose that a given physics model f_P is completely correct for given data. Then, the discrepancy between the original model and its "physics-only" reduced model (where f_A is somehow invalidated) should be close to zero because the decoder of both the original model (with f_P and f_A working) and the reduced model (with only f_P working) should coincide in an ideal limit with the true data-generating process. Even if f_P captures only a part of the truth, the discrepancy should be kept small, if not zero, to ensure meaningful use of the physics models in the overall model.

The "physics-only" reduced model is created as follows. Recall that the original VAE is defined by Eqs. (3) and (4). We define the decoder of the reduced model by replacing $f_\mathrm{A}\colon \mathcal{Y}_\mathrm{P}\times\mathcal{Z}_\mathrm{A}\to\mathcal{Y}_\mathrm{A}$ of (3) with a *baseline function* $h_\mathrm{A}\colon \mathcal{Y}_\mathrm{P}\to\mathcal{Y}_\mathrm{A}$. That is, the reduced observation model is

$$p^\mathrm{r}_{\theta,\theta^\mathrm{r}}(\boldsymbol{x}\mid\boldsymbol{z}_\mathrm{P},\boldsymbol{z}_\mathrm{A}) \coloneqq \mathcal{N}\big(\boldsymbol{x}\mid\mathcal{F}[h_\mathrm{A}(f_\mathrm{P}(\boldsymbol{z}_\mathrm{P}))],\boldsymbol{\Sigma}_x\big). \tag{3r}$$

We denote the set of the trainable parameters of h_A as θ^r, while it may often be empty. The corresponding encoder is defined as follows. Recall that in the original model, posterior distributions of both $\boldsymbol{z}_\mathrm{P}$ and $\boldsymbol{z}_\mathrm{A}$ are inferred in (4) and then used for reconstructing each input $\boldsymbol{x}$ in (3). On the other hand, in the "physics-only" reduced model, $\boldsymbol{z}_\mathrm{A}$ is not referred to by (3r), which makes it less meaningful to place a particular posterior of $\boldsymbol{z}_\mathrm{A}$ for each $\boldsymbol{x}$. Hence, we define the "physics-only" encoder by marginalizing out $\boldsymbol{z}_\mathrm{A}$ and using prior[2] $p(\boldsymbol{z}_\mathrm{A})$ instead. That is, the reduced posterior is

$$q^\mathrm{r}_\psi(\boldsymbol{z}_\mathrm{A},\boldsymbol{z}_\mathrm{P}\mid\boldsymbol{x}) \coloneqq p(\boldsymbol{z}_\mathrm{A})\int q_\psi(\boldsymbol{z}_\mathrm{P},\boldsymbol{z}_\mathrm{A},\mid\boldsymbol{x})\mathrm{d}\boldsymbol{z}_\mathrm{A}. \tag{4r}$$

Below we give a guideline for the choice of the baseline function, h_A:

- If the ranges of f_P and f_A are the same (i.e., $\mathcal{Y}_\mathrm{P}=\mathcal{Y}_\mathrm{A}$), then h_A can be an identity function $h_\mathrm{A}=\mathrm{Id}$. Note that in the additive case $f_\mathrm{A}\circ f_\mathrm{P}=f_\mathrm{P}+f_{\mathrm{A}'}$, where $f_{\mathrm{A}'}$ is a trainable function, replacing f_A with $h_\mathrm{A}=\mathrm{Id}$ is equivalent to replacing $f_{\mathrm{A}'}$ with $h_{\mathrm{A}'}=0$.
- If $\mathcal{Y}_\mathrm{P}\neq\mathcal{Y}_\mathrm{A}$, then h_A can be a linear or affine map from $\mathcal{Y}_\mathrm{P}$ to $\mathcal{Y}_\mathrm{A}$. For example, if $\mathcal{Y}_\mathrm{P}=\mathbb{R}^{d_\mathrm{P}}$ and $\mathcal{Y}_\mathrm{A}=\mathbb{R}^{d_\mathrm{A}}$ ($d_\mathrm{P}\neq d_\mathrm{A}$), then we can set $h_\mathrm{A}(f_\mathrm{P}(\boldsymbol{z}_\mathrm{P}))=\boldsymbol{W}f_\mathrm{P}(\boldsymbol{z}_\mathrm{P})$ where $\boldsymbol{W}\in\mathbb{R}^{d_\mathrm{A}\times d_\mathrm{P}}$.

The idea is to minimize the discrepancy between the full model and the "physics-only" reduced model. In particular, we minimize the discrepancy between the posterior predictive distributions

$$\begin{gathered} D_\mathrm{KL}\big[p_{\theta,\psi}(\tilde{\boldsymbol{x}}\mid X)\parallel p^\mathrm{r}_{\theta,\theta^\mathrm{r},\psi}(\tilde{\boldsymbol{x}}\mid X)\big],\quad\text{where}\\ p_{\theta,\psi}(\tilde{\boldsymbol{x}}\mid X)=\int p_\theta(\tilde{\boldsymbol{x}}\mid\boldsymbol{z}_\mathrm{P},\boldsymbol{z}_\mathrm{A})q_\psi(\boldsymbol{z}_\mathrm{P},\boldsymbol{z}_\mathrm{A}\mid\boldsymbol{x})p_\mathrm{d}(\boldsymbol{x}\mid X)\mathrm{d}\boldsymbol{z}_\mathrm{P}\mathrm{d}\boldsymbol{z}_\mathrm{A}\mathrm{d}\boldsymbol{x},\\ p^\mathrm{r}_{\theta,\theta^\mathrm{r},\psi}(\tilde{\boldsymbol{x}}\mid X)=\int p^\mathrm{r}_{\theta,\theta^\mathrm{r}}(\tilde{\boldsymbol{x}}\mid\boldsymbol{z}_\mathrm{P},\boldsymbol{z}_\mathrm{A})q^\mathrm{r}_\psi(\boldsymbol{z}_\mathrm{P},\boldsymbol{z}_\mathrm{A}\mid\boldsymbol{x})p_\mathrm{d}(\boldsymbol{x}\mid X)\mathrm{d}\boldsymbol{z}_\mathrm{P}\mathrm{d}\boldsymbol{z}_\mathrm{A}\mathrm{d}\boldsymbol{x}. \end{gathered} \tag{6}$$

$p_\mathrm{d}(\boldsymbol{x}\mid X)$ is the empirical distribution with the support on data $X=\{\boldsymbol{x}_1,\dots,\boldsymbol{x}_n\}$. We use $\tilde{\boldsymbol{x}}$ (instead of $\boldsymbol{x}$) just for avoiding notational confusion by clarifying the target of integral $\int\mathrm{d}\boldsymbol{x}$.

[2] It is just for defining q^r_ψ on the common support with q_ψ. Any non-informative distributions of $\boldsymbol{z}_\mathrm{A}$ are fine.

Unfortunately, analytically computing (6) is usually intractable. Hence, we take the following upper bound of (6) (a proof is in Appendix B, and further remarks are in Appendix C):

Proposition 1. *Let* p_θ *and* p_θ^{r} *be the shorthand of* $p_\theta(\tilde{\boldsymbol{x}} \mid \boldsymbol{z}_{\mathrm{P}}, \boldsymbol{z}_{\mathrm{A}})$ *in* (3) *and* $p_{\theta,\theta^{\mathrm{r}}}^{\mathrm{r}}(\tilde{\boldsymbol{x}} \mid \boldsymbol{z}_{\mathrm{P}}, \boldsymbol{z}_{\mathrm{A}})$ *in* (3r)*, respectively. Let* p_{P} *and* p_{A} *be some distributions of* $\boldsymbol{z}_{\mathrm{P}}$ *and* $\boldsymbol{z}_{\mathrm{A}}$*, e.g.,* $p(\boldsymbol{z}_{\mathrm{P}})$ *and* $p(\boldsymbol{z}_{\mathrm{A}})$ *using the priors in* (2)*, respectively. The KL divergence in* (6) *can be upper bounded as follows:*

$$D_{\mathrm{KL}}\big[p_{\theta,\psi}(\tilde{\boldsymbol{x}} \mid X) \parallel p_{\theta,\theta^{\mathrm{r}},\psi}^{\mathrm{r}}(\tilde{\boldsymbol{x}} \mid X)\big] \leq \mathbb{E}_{p_{\mathrm{d}}(\boldsymbol{x}|X)}\Big[\mathbb{E}_{q_\psi(\boldsymbol{z}_{\mathrm{P}},\boldsymbol{z}_{\mathrm{A}}|\boldsymbol{x})} D_{\mathrm{KL}}[p_\theta \parallel p_\theta^{\mathrm{r}}] + D_{\mathrm{KL}}[q_\psi(\boldsymbol{z}_{\mathrm{A}} \mid \boldsymbol{x}) \parallel p_{\mathrm{A}}] + \mathbb{E}_{q_\psi(\boldsymbol{z}_{\mathrm{A}}|\boldsymbol{x})} D_{\mathrm{KL}}[q_\psi(\boldsymbol{z}_{\mathrm{P}} \mid \boldsymbol{z}_{\mathrm{A}}, \boldsymbol{x}) \parallel p_{\mathrm{P}}]\Big]. \quad (7)$$

Definition 1. Let us denote the upper bound (7) by $\mathbb{E}_{p_{\mathrm{d}}(\boldsymbol{x}|X)} \hat{D}(\theta, \theta^{\mathrm{r}}, \psi; \boldsymbol{x})$. The regularization for harnessing unnecessary flexibility of trainable functions is defined as minimization of

$$R_{\mathrm{PPC}}(\theta, \theta^{\mathrm{r}}, \psi) \coloneqq \mathbb{E}_{p_{\mathrm{d}}(\boldsymbol{x}|X)} \hat{D}(\theta, \theta^{\mathrm{r}}, \psi; \boldsymbol{x}). \quad (8)$$

Remark 1. When multiple trainable functions are differently used in a model (e.g., inside *and* outside an equation solver), which is often the case in practice, the definition of R_{PPC} should be generalized to consider marginal contribution of every trainable function. See Appendix A.

3.2 Grounding physics encoder by physics-based data augmentation

Toward properly learning physics-integrated VAEs, minimizing R_{PPC} solely may not be enough because inferred $\boldsymbol{z}_{\mathrm{P}}$ may be still meaningless but makes R_{PPC} not that large (e.g., with solution of f_{P} fluctuating around the mean pattern of data). Though it is difficult to avoid such a local solution perfectly, we can alleviate the situation by considering additional objectives to encourage a proper use of the physics. The idea is to use the physics model as a source of information for data augmentation, which helps us to ground the output of the recognition network, g_{P} in (4), to the parameters of f_{P}.

Let $\boldsymbol{z}_{\mathrm{P}}^*$ be a sample drawn from some distribution of $\boldsymbol{z}_{\mathrm{P}}$ (e.g., prior $p(\boldsymbol{z}_{\mathrm{P}})$). We artificially generate signals $\boldsymbol{x}^*$ by feeding $\boldsymbol{z}_{\mathrm{P}}^*$ to the "physics-only" decoding process in (3r), that is, $\boldsymbol{x}_{\boldsymbol{z}_{\mathrm{P}}^*}^* \coloneqq \mathcal{F}[h_{\mathrm{A}}(f_{\mathrm{P}}(\boldsymbol{z}_{\mathrm{P}}^*))]$. We want the physics-part recognition network, g_{P}, to successfully estimate $\boldsymbol{z}_{\mathrm{P}}^*$ given the corresponding $\boldsymbol{x}_{\boldsymbol{z}_{\mathrm{P}}^*}^*$, which is necessary to say that the result of the inference by g_{P} is grounded to the parameters of f_{P}. However, in general, real data $\boldsymbol{x}$ and the augmented data $\boldsymbol{x}^*$ have different natures because f_{P} may miss some aspects of the true data-generating process. We handle this issue by considering a specific design of the physics-part recognition network, g_{P}.

Let us decompose g_{P} as $g_{\mathrm{P}}(\boldsymbol{x}, \boldsymbol{z}_{\mathrm{A}}) = g_{\mathrm{P},2}(g_{\mathrm{P},1}(\boldsymbol{x}, \boldsymbol{z}_{\mathrm{A}}))$ without loss of generality. On one hand, $g_{\mathrm{P},1}$ should transform real data $\boldsymbol{x}$ into $\boldsymbol{x}'$ such that $\boldsymbol{x}'$ *resembles the physics-based augmented signal* $\boldsymbol{x}^*$. In other words, $g_{\mathrm{P},1}$ should "cleanse" real data into a virtual "physics-only" counterpart. On the other hand, $g_{\mathrm{P},2}$ should receive such "cleansed" data $\boldsymbol{x}'$ and return the (sufficient statistics of) posterior of $\boldsymbol{z}_{\mathrm{P}}$. As $g_{\mathrm{P},2}$ works on $\boldsymbol{x}'$, which should resemble $\boldsymbol{x}^*$, we can directly self-supervise $g_{\mathrm{P},2}$ with $\boldsymbol{x}^*$. We define a couple of regularizers for setting such functionality of $g_{\mathrm{P},1}$ and $g_{\mathrm{P},2}$ as follows:

Definition 2. Let $\mathrm{sg}(\cdot)$ be the stop-gradient operator. Let $\boldsymbol{x}' \coloneqq \mathcal{F}[h_{\mathrm{A}}(f_{\mathrm{P}}(g_{\mathrm{P}}(\boldsymbol{x}, \boldsymbol{z}_{\mathrm{A}})))]$. The regularization for the physics-based data augmentation is defined as minimization of

$$R_{\mathrm{DA},1}(\psi) \coloneqq \mathbb{E}_{p_{\mathrm{d}}(\boldsymbol{x}|X) q(\boldsymbol{z}_{\mathrm{A}}|\boldsymbol{x})} \big\| g_{\mathrm{P},1}(\boldsymbol{x}, \boldsymbol{z}_{\mathrm{A}}) - \mathrm{sg}\, \boldsymbol{x}' \big\|_2^2 \quad \text{and} \quad (9)$$

$$R_{\mathrm{DA},2}(\psi) \coloneqq \mathbb{E}_{\boldsymbol{z}_{\mathrm{P}}^*} \big\| g_{\mathrm{P},2}\big(\mathrm{sg}\, \boldsymbol{x}_{\boldsymbol{z}_{\mathrm{P}}^*}^*\big) - \boldsymbol{z}_{\mathrm{P}}^* \big\|_2^2. \quad (10)$$

Remark 2. If both $g_{\mathrm{P},1}$ and $g_{\mathrm{P},2}$ work as intended (i.e., both $R_{\mathrm{DA},1}$ and $R_{\mathrm{DA},2}$ are small enough), $\boldsymbol{x}'$ is the virtual "physics-only" counterpart of $\boldsymbol{x}$. $R_{\mathrm{DA},1}$ is for ensuring the functionality of $g_{\mathrm{P},1}$ to "cleanse" $\boldsymbol{x}$ to $\boldsymbol{x}'$. $R_{\mathrm{DA},2}$ is for giving the supervision to $g_{\mathrm{P},2}$ with the augmented data $(\boldsymbol{z}_{\mathrm{P}}^*, \boldsymbol{x}^*)$.

3.3 Overall regularized learning objective

The overall regularized learning problem of the proposed physics-integrated VAEs is as follows:

$$\operatorname*{minimize}_{\theta,\theta^{\mathrm{r}},\psi} \; -\mathbb{E}_{p_{\mathrm{d}}(\boldsymbol{x}|X)} \mathrm{ELBO}(\theta, \psi; \boldsymbol{x}) + \alpha R_{\mathrm{PPC}}(\theta, \theta^{\mathrm{r}}, \psi) + \beta R_{\mathrm{DA},1}(\psi) + \gamma R_{\mathrm{DA},2}(\psi),$$

where each term appears in (5), (8), (9), and (10), respectively. Recall that θ, ψ, and θ^{r} are the sets of the parameters of the full model's decoder (3), encoder (4), and the reduced model's decoder (3r), respectively, while θ^{r} may be empty. If we cannot specify a reasonable sampling distribution of $\boldsymbol{z}_{\mathrm{P}}^*$ needed in (10), we do not compute $R_{\mathrm{DA},1}$ and $R_{\mathrm{DA},2}$ and set $\beta = \gamma = 0$; it may happen when the semantics of $\boldsymbol{z}_{\mathrm{P}}$ are not inherently grounded, e.g., when f_{P} is a *neural* Hamilton's equation [37].

4 Related work

The integration of theory-driven and data-driven methodologies has been sought in various ways. Ones in model design, which we followed, are one of the key approaches. Other approaches have also been studied; e.g., physics-informed neural nets (PINNs) [27] incorporate physics knowledge in the definition of loss function. We overview these perspectives in this section and more in Appendix D.

Physics+ML in model design Integration in model design, often called gray-box or hybrid modeling, has been a subject of study for decades [e.g., 24, 29, 36] and is still active, with deep neural networks employed in various applications [e.g., 45, 26, 21, 39, 23, 1, 2, 8, 46, 40, 32, 16, 22, 5, 33, 25, 19]. Most recent studies focus on prediction, and the generative modeling has been less investigated. Moreover, mechanisms to harness trainable components have hardly been addressed.

The work of Yin et al. [44] is notable here because they consider a mechanism to harness a trainable component to preserve the utility of physics in the model, even though it is only focused on dynamics learning for forecasting. They learn an additive hybrid ODE model $\dot{x} = f_\mathrm{P}(x) + f_\mathrm{A}(x)$, where f_P is a prescribed physics model, and f_A is a neural network. Such a model is subsumed in our architecture as exemplified in Section 2. Moreover, Yin et al. [44] propose to harness f_A by minimizing $\|f_\mathrm{A}\|^2$. Such a term also appears in one of our regularizers, R_PPC; when the observation noise is Gaussian, the first term of the rhs of (7) becomes $\mathbb{E}\|(f_\mathrm{A} \circ f_\mathrm{P}) - f_\mathrm{P}\|_2^2 = \mathbb{E}\|f_\mathrm{P} + f_{\mathrm{A}'} - f_\mathrm{P}\|_2^2 = \mathbb{E}\|f_{\mathrm{A}'}\|_2^2$. Therefore, we get a "VAE variant" of Yin et al. [44] by switching off a part of R_PPC and the other regularizers, $R_{\mathrm{DA},1}$ and $R_{\mathrm{DA},2}$. We examine cases similar to it in our experiment for comparison.

Yıldız et al. [43] and Linial et al. [20] developed VAEs whose latent variable follows ODEs. Linial et al. [20] also suggest grounding the semantics of the latent variable by providing sparse supervision on it. It is feasible only when we have a chance to observe the latent variable (e.g., with an increased cost) and may often be inherently infeasible in some problem settings including ours. In our method, we never assume availability of observation of latent variables and instead use the physics models in a self-supervised manner. While direct comparison is not meaningful due to the difference of settings, we examine a baseline close to the base model of Linial et al. [20] in our experiment for comparison.

Toth et al. [37] propose a model where the latent variable sequence is governed by the Hamiltonian mechanics with a neural Hamiltonian. While it does not suppose very specific physics models but considers general mechanics, they can also be included in our framework; that is, f_P can be a Hamilton's equation with a neural Hamiltonian. We try such a model in one of our experiments.

Physics+ML in objective design Another prevailing strategy is to define objective functions based on physics knowledge [e.g., 34, 14, 27, 12, 42, 13, 47, 30, 6]. In generative modeling, for example, Stinis et al. [35] use residuals from physics models as a feature of GAN's discriminator. Golany et al. [10] regularize the generation from GANs by forcing it close to a prescribed physics relation. These approaches are often easy to deploy, but an inherent limitation is that given physics knowledge should be complete to some extent, otherwise a physics-based loss is not well-defined.

5 Experiments

We performed experiments on two synthetic datasets and two real-world datasets, for which we prepared instances of physics-integrated VAEs. We show each particular architecture of physics-integrated VAEs and the corresponding results; some details are deferred to Appendix E. While direct comparison is impossible due to the differences of the problem settings, the baseline methods we examined (listed below) are similar to some existing methods [4, 43, 37, 20, 44].

- `NN-only` — Ordinary VAE [15, 28]; the decoder is $\mathbb{E}\boldsymbol{x} = f_\mathrm{A}(\boldsymbol{z}_\mathrm{A})$, where f_A is a neural net.
- `Phys-only` — Physics VAE; the decoder is $\mathbb{E}\boldsymbol{x} = \mathcal{F}(f_\mathrm{P}(\boldsymbol{z}_\mathrm{P}))$, while the encoder is with neural nets as usual. Almost equivalent to Aragon-Calvo and Carvajal [4] in Section 5.3.
- `NN+solver` — VAE with physics solvers; the decoder is $\mathbb{E}\boldsymbol{x} = \mathcal{F}(f_\mathrm{A}(\boldsymbol{z}_\mathrm{A}))$, where f_A is a neural net, and $\mathcal{F}$ includes some equation-solving process (e.g., ODE/PDE solver). It is similar to the methods of, for example, Yıldız et al. [43] and Toth et al. [37].
- `NN+phys` — Physics-integrated VAE learned without the regularizers (i.e., $\alpha = \beta = \gamma = 0$); almost equivalent to the base model of Linial et al. [20]. Finer ablations are also studied, among which the cases with $\beta = 0$ or $\gamma = 0$ are similar to Yin et al. [44].

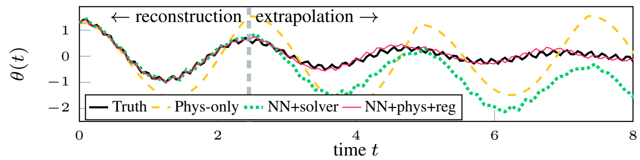

Figure 1: Reconstruction and extrapolation of a test sample of the pendulum data. Range $0 \leq t < 2.5$ is reconstruction, whereas $t \geq 2.5$ is extrapolation.

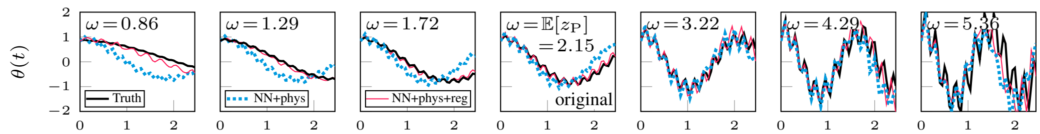

Figure 2: Counterfactual generation for the pendulum data. Horizontal axis is time t. The center panel shows the original data, and the rest is the generation with z_{P} (i.e., ω) altered while $\boldsymbol{z}_{\mathrm{A}}$ fixed.

`NN+phys+reg` Our proposal; physics-integrated VAE learned with the proposed regularizers.

We aligned the total dimensionality of the latent variables of each method (except `phys-only`); when $\dim \boldsymbol{z}_{\mathrm{A}} = d_{\mathrm{A}}$ and $\dim \boldsymbol{z}_{\mathrm{P}} = d_{\mathrm{P}}$ in `NN+phys+reg`, we set $\dim \boldsymbol{z}_{\mathrm{A}} = d_{\mathrm{A}} + d_{\mathrm{P}}$ in `NN-only` and `NN+solver`. The hyperparameters, α, β, and γ, were chosen with validation set performance. We investigated the performance sensitivity to them. No large degradation of performance was observed even if we changed the values by $\times 10$ or $\times \frac{1}{10}$ from the chosen values; details are in Appendix F.

5.1 Forced damped pendulum

Dataset We generated data from (1) with $u(t) = A\omega^2 \cos(2\pi\phi t)$. Each data-point $\boldsymbol{x}$ is a sequence $\boldsymbol{x} := [\theta_1 \ \cdots \theta_\tau] \in \mathbb{R}^\tau$, where θ_j is the value of a solution $\theta(t_j)$ at $t_j := (j-1)\Delta t$. We randomly drew a sample of the initial condition θ_1 (with $\dot{\theta}_1 = 0$ fixed) and the values of ω, ζ, A, and ϕ for each sequence. We generated 2,500 sequences of length $\tau = 50$ with $\Delta t = 0.05$ and separated them into a training, validation, and test sets with 1,000, 500, and 1,000 sequences, respectively.

Setting We set f_{P} as in Section 2.1, i.e., $f_{\mathrm{P}}(\theta, z_{\mathrm{P}}) := \ddot{\theta} + z_{\mathrm{P}}^2 \sin(\theta)$, where $z_{\mathrm{P}} \in \mathbb{R}$ should work as angular velocity ω. We augmented it by $f_{\mathrm{A},1}(\theta, z_{\mathrm{A},1})$ additively, where $f_{\mathrm{A},1}$ was a multi-layer perceptron (MLP) and $z_{\mathrm{A},1} \in \mathbb{R}$. The ODE $f_{\mathrm{P}} + f_{\mathrm{A},1} = 0$ is solved with the Euler update scheme in the model. The model has another MLP[3] $f_{\mathrm{A},2}$ with another latent variable $\boldsymbol{z}_{\mathrm{A},2} \in \mathbb{R}^2$ for further modifying the solution of the ODE. In summary, the decoding process is $\mathcal{F} := f_{\mathrm{A},2}(\mathrm{solve}_\theta[f_{\mathrm{P}}(\theta, z_{\mathrm{P}}) + f_{\mathrm{A},1}(\theta, z_{\mathrm{A},1}) = 0], \boldsymbol{z}_{\mathrm{A},2})$. The construction of the proposed regularizer for such multiple f_{A}'s is elaborated in Appendix A. We used $h_{\mathrm{A},1} = 0$ and $h_{\mathrm{A},2} = \mathrm{Id}$ as the baseline functions. The recognition networks g were modeled with MLPs. We used the initial element of each $\boldsymbol{x}$ as an estimation of the initial condition θ_1.

Results Figure 1 demonstrates a unique benefit of the hybrid modeling. We show an example of reconstruction with extrapolation. Recall that the training data comprise sequences of range $0 \leq t < 2.5$ only; so the results in $t \geq 2.5$ are extrapolation (in time) rather than mere reconstruction. We can observe that while `NN+solver` cannot extrapolate even if it is equipped with an neural ODE, `NN+phys+reg` can reconstruct and extrapolate correctly.

Figure 2 illustrates well the advantage of the proposed regularizers. We show an example of generation from learned models with z_{P} manipulated. Recall that z_{P} is expected to work as pendulum's angular velocity ω. We took a test sample with $\omega \approx \mathbb{E}[z_{\mathrm{P}}] \approx 2.15$ and generated signals with the original and different values of z_{P}, keeping the values of $\boldsymbol{z}_{\mathrm{A}}$ to be the original posterior mean. We can see that the generation from `NN+phys+reg` matches better with the signals from the true process.

Table 1 (left half) summarizes the performance in terms of the reconstruction error and the inference error of physics parameter ω on the test set. The errors are reported in mean absolute errors (MAEs). The inference error of ω is evaluated by $|\mathbb{E}[z_{\mathrm{P}}] - \omega_{\text{true}}|$. `NN+phys+reg` achieves small values in *both* reconstruction error and inference error. The MAE of ω inferred by `NN+phys` is significantly worse than the others, which indicates the importance of the proposed regularizers.

[3] We used MLP as the data are fixed length. The same holds hereafter. Extension to other networks is easy.

Table 1: Reconstruction errors and inference errors on test sets of the pendulum data and the advection-diffusion data. Averages (and SDs) over 20 random trials are reported.

		Pendulum				Advection-diffusion			
		MAE of reconst.		MAE of inferred ω		MAE of reconst.		MAE of inferred a	
	`NN-only`	0.438	(2.9×10^{-2})	–		0.0396	(2.2×10^{-4})	–	
	`Phys-only`	1.55	(7.1×10^{-4})	0.232	(5.9×10^{-3})	0.393	(9.5×10^{-4})	0.0103	(1.5×10^{-3})
	`NN+solver`	0.439	(2.3×10^{-2})	–		0.0388	(1.7×10^{-4})	–	
	`NN+phys`	0.370	(4.3×10^{-2})	1.04	(2.2×10^{-1})	0.0404	(1.2×10^{-2})	0.258	(3.2×10^{-1})
	`NN+phys+reg`	0.363	(4.8×10^{-2})	0.229	(3.8×10^{-2})	0.0437	(1.5×10^{-3})	0.00951	(6.2×10^{-3})
Ablations	$\alpha=0$	0.396	(4.3×10^{-2})	0.889	(1.9×10^{-1})	0.0461	(1.3×10^{-2})	0.0444	(1.4×10^{-2})
	$\beta=0$	0.372	(4.1×10^{-2})	0.223	(3.6×10^{-2})	0.0747	(2.4×10^{-2})	0.199	(2.3×10^{-1})
	$\gamma=0$	0.381	(4.1×10^{-2})	0.276	(4.2×10^{-2})	0.0588	(9.1×10^{-4})	0.0548	(9.4×10^{-7})

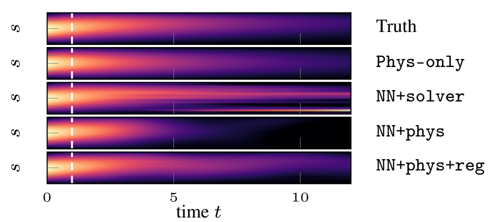

Figure 3: Reconstruction and extrapolation of a test sample of the advection-diffusion data. Range $0 \le t < 1$ is reconstruction, whereas $t \ge 1$ is extrapolation; dashed line is the border.

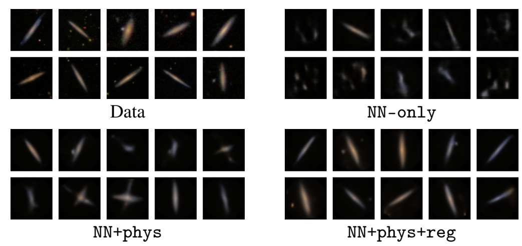

Figure 4: (*left*) Subset of the galaxy image data. (*right three*) Random generation from the `NN-only` model and the `NN+phys+reg` models.

5.2 Advection-diffusion system

Dataset We generated data from advection-diffusion PDE $\partial T/\partial t - a \cdot \partial^2 T/\partial s^2 + b \cdot \partial T/\partial s = 0$, where s is the 1-D spatial dimension. We approximated the solution $T(s,t)$ on the 12-point even grid from $s=0$ to $s=s_{\text{max}}$, so each data-point $\boldsymbol{x}$ is a sequence of 12-dim vectors, i.e., $\boldsymbol{x} := [\boldsymbol{T}_1 \ \cdots \ \boldsymbol{T}_\tau] \in \mathbb{R}^{12\times\tau}$, where $\boldsymbol{T}_j := [T(0,t_j) \ \cdots \ T(s_{\text{max}},t_j)]^\mathsf{T}$ at $t_j := (j-1)\Delta t$. We set the boundary condition as $T(0,t) = T(s_{\text{max}},t) = 0$ and the initial condition as $T(s,0) = c\sin(\pi s/s_{\text{max}})$. We randomly drew a, b, and c for each $\boldsymbol{x}$. We generated 2,500 sequences with $\tau = 50$ and $\Delta t = 0.02$ and separated them into a training, validation, and test sets with 1,000, 500, and 1,000 sequences, respectively.

Setting We set f_{P} as the diffusion PDE, i.e., $f_{\text{P}}(T, z_{\text{P}}) := \partial T/\partial t - z_{\text{P}} \partial^2 T/\partial s^2$, where $z_{\text{P}} \in \mathbb{R}$ should work as diffusion coefficient a. We augmented it by $f_{\text{A}}(T, \boldsymbol{z}_{\text{A}})$ additively, where f_{A} was an MLP and $\boldsymbol{z}_{\text{A}} \in \mathbb{R}^4$. Hence, the decoding process is $\mathcal{F} := \text{solve}_T[f_{\text{P}}(T, z_{\text{P}}) + f_{\text{A}}(T, \boldsymbol{z}_{\text{A}}) = 0]$. We used $h_{\text{A}} = 0$ as the baseline function. The recognition networks g were modeled with MLPs. We used the initial snapshot of each sequence $\boldsymbol{x}$ as an estimation of the initial condition $\boldsymbol{T}_1$.

Results Figure 3 shows an example of reconstruction with extrapolation. As the training data only comprise sequences of range $0 \le t < 1$, the remaining range $t \ge 1$ is extrapolation. Only `NN+phys+reg` (the bottom panel) achieves adequate extrapolation; `phys-only` lacks advection, `NN+solver` has unnatural artifacts, and `NN+phys` infers z_{P} (i.e., diffusion coefficient a) wrongly.

Table 1 (right half) summarizes the reconstruction and inference errors, which are consistent with the results in the pendulum example. We also show the performance of ablations of `NN+phys+reg`, where either of the regularizers was turned off (i.e., $\alpha = 0$, $\beta = 0$, or $\gamma = 0$). Not surprisingly their performance is worse than the full regularization, especially in terms of the inference error.

5.3 Galaxy images

Dataset We used images of galaxy of the Galaxy10 dataset [18]. We selected the 589 images of the "Disk, Edge-on, No Bulge" class and separated them into training, validation, and test sets with 400, 100, and 89 images, respectively. Each image is of size 69×69 with three channels. We performed data augmentation with random rotation and increased the size of the training set by 20 times.

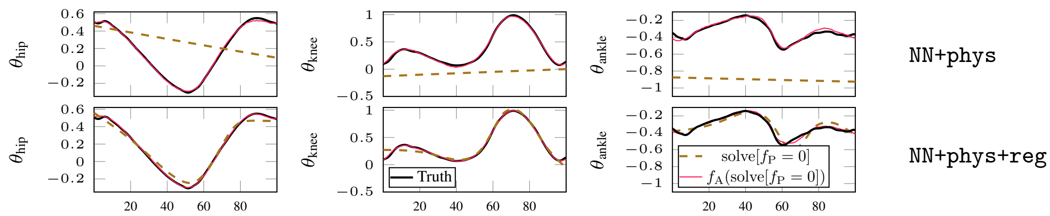

Figure 5: Reconstruction of a test sample of the gait data. Horizontal axis is normalized time.

Setting We set $f_\mathrm{P} \colon \mathbb{R}^4_{>0} \to \mathbb{R}^{69\times 69}$ as an exponential profile of the light distribution of galaxies [see 4, and references therein] whose input is $\boldsymbol{z}_\mathrm{P} := [I_0 \ A \ B \ \theta]^\mathsf{T} \in \mathbb{R}^4_{>0}$. Let $[f_\mathrm{P}(\boldsymbol{z}_\mathrm{P})]_{i,j}$ denote the (i,j)-element of the output of f_P. Then, for $1 \le i,j \le 69$, $[f_\mathrm{P}(\boldsymbol{z}_\mathrm{P})]_{i,j} := I_0 \exp(-r_{i,j})$, where $r_{i,j}^2 := (\mathsf{X}_j \cos\theta - \mathsf{Y}_i \sin\theta)^2/A^2 + (\mathsf{X}_j \sin\theta + \mathsf{Y}_i \cos\theta)^2/B^2$, and $(\mathsf{X}_j, \mathsf{Y}_i)$ is the coordinate on the 69×69 even grid on $[-1,1] \times [-1,1]$. We modify the output of f_P using a U-Net-like neural network $f_\mathrm{A} \colon \mathbb{R}^{69\times 69} \times \mathbb{R}^{\dim \boldsymbol{z}_\mathrm{A}} \to \mathbb{R}^{69\times 69\times 3}$. Thus, the decoding process is $\mathcal{F} := f_\mathrm{A}(f_\mathrm{P}(\boldsymbol{z}_\mathrm{P}), \boldsymbol{z}_\mathrm{A})$. We set $\dim \boldsymbol{z}_\mathrm{A} = 2$ for `NN+phys+reg`. We set $h_\mathrm{A} \colon \mathbb{R}^{69\times 69} \to \mathbb{R}^{69\times 69\times 3}$ to be the repeat operator along the channel axis. The encoding process is as follows: first, features are extracted from an image $\boldsymbol{x}$ by a convolutional net like [4]. The extracted features are flattened and fed to MLPs g_P and g_A.

Results Figure 4 shows an example of original data and random generation from the learned models. `NN-only` tends to generate non-realistic images, and `NN+phys` generates slightly better but still spuriously, whereas `NN+phys+reg` consistently generates galaxy-like images. More results (reconstruction, counterfactual generation, and inspection of latent variable) are deferred to Appendix F.

5.4 Human gait

Dataset We used a part of the dataset provided by [17], which contains measurements of locomotion at different speeds of 50 subjects. We extracted the angles of hip, knee, and ankle in the sagittal plane. Data originally comprise sequences of each stride normalized to be 100 steps, so each data-point $\boldsymbol{x}$ is a sequence $\boldsymbol{x} := [\boldsymbol{\theta}_1 \ \cdots \boldsymbol{\theta}_{100}] \in \mathbb{R}^{3\times 100}$, where $\boldsymbol{\theta}_j := [\theta_{\text{hip},j} \ \theta_{\text{knee},j} \ \theta_{\text{ankle},j}]^\mathsf{T}$. We used different 400, 100, and 344 sequences as training, validation, and test sets, respectively.

Setting Biomechanical modeling of gait is a long-standing problem [see, e.g., 31]. We did not choose a specific model but let f_P be a trainable Hamilton's equation as in [37, 11]. $\boldsymbol{z}_\mathrm{P} \in \mathbb{R}^{2d_\mathrm{H}}$ works as the initial conditions of it, where d_H is the dimensionality of the generalized position. We let $d_\mathrm{H} = 3$ and modeled the neural Hamiltonian with an MLP. The solution of $f_\mathrm{P} = 0$ is transformed by f_A that also takes $\boldsymbol{z}_\mathrm{A} \in \mathbb{R}^{15}$ as an argument. In summary, the decoding process is $\mathcal{F} = f_\mathrm{A}(\mathrm{solve}[f_\mathrm{P} = 0], \boldsymbol{z}_\mathrm{A})$. We set h_A to be an affine transform at each timestep, which has a weight matrix and a bias as θ^r. The recognition networks g were modeled with MLPs.

Results Figure 5 is for visually comparing the difference of the learned models' behavior due to the proposed regularizers. We compare the reconstructions by `NN+phys` and `NN+phys+reg`. The dashed lines show an intermediate of the decoding process, i.e., $\mathrm{solve}[f_\mathrm{P} = 0]$, and the red solid lines show the final reconstruction, i.e., $f_\mathrm{A}(\mathrm{solve}[f_\mathrm{P} = 0])$. Without the regularization (upper row), $\mathrm{solve}[f_\mathrm{P} = 0]$ returns almost meaningless signals, and f_A bears the most effort of reconstruction. On the other hand, with the regularization (lower row), $\mathrm{solve}[f_\mathrm{P} = 0]$ already matches well the data, and f_A modifies it only slightly. Superiority of the regularized model was also confirmed quantitatively; the average test reconstruction errors were 0.273 with `NN+phys` and 0.259 with `NN+phys+reg`.

6 Conclusion

Physics-integrated VAEs by construction attain partial interpretability as some of the latent variables are semantically grounded to the physics models, and thus we can generate signals in a controlled manner. Moreover, they have extrapolation capability due to the physics models. In this work, we proposed a regularized learning objective for ensuring a proper functionality of the integrated physics models. We empirically validated the aforementioned unique capability of physics-integrated VAEs and the importance of the proposed regularization method.

References

[1] A. Ajay, J. Wu, N. Fazeli, M. Bauza, L. P. Kaelbling, J. B. Tenenbaum, and A. Rodriguez. Augmenting physical simulators with stochastic neural networks: Case study of planar pushing and bouncing. In *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3066–3073, 2018.

[2] A. Ajay, M. Bauza, J. Wu, N. Fazeli, J. B. Tenenbaum, A. Rodriguez, and L. P. Kaelbling. Combining physical simulators and object-based networks for control. In *Proceedings of the 2019 IEEE International Conference on Robotics and Automation*, pages 3217–3223, 2019.

[3] B. Amos and J. Z. Kolter. OptNet: Differentiable optimization as a layer in neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 136–145, 2017.

[4] M. A. Aragon-Calvo and J. C. Carvajal. Self-supervised learning with physics-aware neural networks – I. Galaxy model fitting. *Monthly Notices of the Royal Astronomical Society*, 498(3): 3713–3719, 2020.

[5] F. d. A. Belbute-Peres, T. D. Economon, and J. Z. Kolter. Combining differentiable PDE solvers and graph neural networks for fluid flow prediction. In *Proceedings of the 37th International Conference on Machine Learning*, pages 2402–2411, 2020.

[6] C. Chen, G. Zheng, H. Wei, and Z. Li. Physics-informed generative adversarial networks for sequence generation with limited data. NeurIPS Workshop on Interpretable Inductive Biases and Physically Structured Learning, 2020.

[7] T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems 31*, pages 6572–6583, 2018.

[8] E. de Bézenac, A. Pajot, and P. Gallinari. Deep learning for physical processes: Incorporating prior scientific knowledge. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12): 124009, 2019.

[9] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtar, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, 3rd edition, 2013.

[10] T. Golany, D. Freedman, and K. Radinsky. SimGANs: Simulator-based generative adversarial networks for ECG synthesis to improve deep ECG classification. In *Proceedings of the 37th International Conference on Machine Learning*, pages 3597–3606, 2020.

[11] S. Greydanus, M. Dzamba, and J. Yosinski. Hamiltonian neural networks. In *Advances in Neural Information Processing Systems 32*, pages 15379–15389, 2019.

[12] X. Jia, J. Willard, A. Karpatne, J. Read, J. Zwart, M. Steinbach, and V. Kumar. Physics guided RNNs for modeling dynamical systems: A case study in simulating lake temperature profiles. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pages 558–566, 2019.

[13] S. Kaltenbach and P.-S. Koutsourelakis. Incorporating physical constraints in a deep probabilistic machine learning framework for coarse-graining dynamical systems. *Journal of Computational Physics*, 419:109673, 2020.

[14] A. Karpatne, W. Watkins, J. Read, and V. Kumar. Physics-guided neural networks (PGNN): An application in lake temperature modeling. arXiv:1710.11431, 2017.

[15] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations*, 2014.

[16] V. Le Guen and N. Thome. Disentangling physical dynamics from unknown factors for unsupervised video prediction. In *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11471–11481, 2020.

[17] T. Lencioni, I. Carpinella, M. Rabuffetti, A. Marzegan, and M. Ferrarin. Human kinematic, kinetic and EMG data during different walking and stair ascending and descending tasks. *Scientific Data*, 6(1):309, 2019.

[18] H. W. Leung and J. Bovy. Deep learning of multi-element abundances from high-resolution spectroscopic data. *Monthly Notices of the Royal Astronomical Society*, 483(3):3255–3277, 2018.

[19] L. Li, S. Hoyer, R. Pederson, R. Sun, E. D. Cubuk, P. Riley, and K. Burke. Kohn-Sham equations as regularizer: Building prior knowledge into machine-learned physics. *Phyiscal Review Letters*, 126(3):036401, 2020.

[20] O. Linial, D. Eytan, and U. Shalit. Generative ODE modeling with known unknowns. arXiv:2003.10775, 2020.

[21] Y. Long and X. She. HybridNet: Integrating model-based and data-driven learning to predict evolution of dynamical systems. In *Proceedings of the 2nd Conference on Robot Learning*, pages 551–560, 2018.

[22] N. Muralidhar, J. Bu, Z. Cao, L. He, N. Ramakrishnan, D. Tafti, and A. Karpatne. PhyNet: Physics guided neural networks for particle drag force prediction in assembly. In *Proceedings of the 2020 SIAM International Conference on Data Mining*, pages 559–567, 2020.

[23] A. Nutkiewicz, Z. Yang, and R. K. Jain. Data-driven Urban Energy Simulation (DUE-S): A framework for integrating engineering simulation and machine learning methods in a multi-scale urban energy modeling workflow. *Applied Energy*, 225:1176–1189, 2018.

[24] D. C. Psichogios and L. H. Ungar. A hybrid neural network-first principles approach to process modeling. *AIChE Journal*, 38(10):1499–1511, 1992.

[25] C. Rackauckas, Y. Ma, J. Martensen, C. Warner, K. Zubov, R. Supekar, D. Skinner, A. Ramadhan, and A. Edelman. Universal differential equations for scientific machine learning. arXiv:2001.04385, 2020.

[26] M. Raissi. Deep hidden physics models: Deep learning of nonlinear partial differential equations. *Journal of Machine Learning Research*, 19(25):1–24, 2018.

[27] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[28] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1278–1286, 2014.

[29] R. Rico-Martínez, J. S. Anderson, and I. G. Kevrekidis. Continuous-time nonlinear signal processing: A neural network based approach for gray box identification. In *Proceedings of the IEEE Workshop on Neural Networks for Signal Processing*, pages 596–605, 1994.

[30] M. Rixner and P.-S. Koutsourelakis. A probabilistic generative model for semi-supervised training of coarse-grained surrogates and enforcing physical constraints through virtual observables. arXiv:2006.01789, 2020.

[31] D. G. E. Robertson, G. E. Caldwell, J. Hamill, G. Kamen, and S. N. Whittlesey. *Research Methods in Biomechanics*. Human Kinetics, 2nd edition, 2014.

[32] M. A. Roehrl, T. A. Runkler, V. Brandtstetter, M. Tokic, and S. Obermayer. Modeling system dynamics with physics-informed neural networks based on Lagrangian mechanics. arXiv:2005.14617, 2020.

[33] U. Sengupta, M. Amos, J. S. Hosking, C. E. Rasmussen, M. Juniper, and P. J. Young. Ensembling geophysical models with Bayesian neural networks. In *Advances in Neural Information Processing Systems 33*, 2020.

[34] R. Stewart and S. Ermon. Label-free supervision of neural networks with physics and domain knowledge. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 2576–2582, 2017.

[35] P. Stinis, T. Hagge, A. M. Tartakovsky, and E. Yeung. Enforcing constraints for interpolation and extrapolation in generative adversarial networks. *Journal of Computational Physics*, 397: 108844, 2019.

[36] M. L. Thompson and M. A. Kramer. Modeling chemical processes using prior knowledge and neural networks. *AIChE Journal*, 40(8):1328–1340, 1994.

[37] P. Toth, D. J. Rezende, A. Jaegle, S. Racanière, A. Botev, and I. Higgins. Hamiltonian generative networks. In *Proceedings of the 8th International Conference on Learning Representations*, 2020.

[38] L. von Rueden, S. Mayer, K. Beckh, B. Georgiev, S. Giesselbach, R. Heese, B. Kirsch, J. Pfrommer, A. Pick, R. Ramamurthy, M. Walczak, J. Garcke, C. Bauckhage, and J. Schuecker. Informed machine learning – A taxonomy and survey of integrating knowledge into learning systems. arXiv:1903.12394v2, 2020.

[39] Z. Y. Wan, P. Vlachas, P. Koumoutsakos, and T. Sapsis. Data-assisted reduced-order modeling of extreme events in complex dynamical systems. *PLOS ONE*, 13(5):e0197704, 2018.

[40] Q. Wang, F. Li, Y. Tang, and Y. Xu. Integrating model-driven and data-driven methods for power system frequency stability assessment and control. *IEEE Transactions on Power Systems*, 34(6):4557–4568, 2019.

[41] J. Willard, X. Jia, S. Xu, M. Steinbach, and V. Kumar. Integrating physics-based modeling with machine learning: A survey. arXiv:2003.04919, 2020.

[42] Z. Yang, J.-L. Wu, and H. Xiao. Enforcing deterministic constraints on generative adversarial networks for emulating physical systems. arXiv:1911.06671, 2019.

[43] Ç. Yıldız, M. Heinonen, and H. Lähdesmäki. ODE2VAE: Deep generative second order ODEs with Bayesian neural networks. In *Advances in Neural Information Processing Systems 32*, pages 13412–13421, 2019.

[44] Y. Yin, V. Le Guen, J. Dona, I. Ayed, E. de Bézenac, N. Thome, and P. Gallinari. Augmenting physical models with deep networks for complex dynamics forecasting. In *Proceedings of the 9th International Conference on Learning Representations*, 2021.

[45] C.-C. Young, W.-C. Liu, and M.-C. Wu. A physically based and machine learning hybrid approach for accurate rainfall-runoff modeling during extreme typhoon events. *Applied Soft Computing*, 53:205–216, 2017.

[46] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser. TossingBot: Learning to throw arbitrary objects with residual physics. In *Proceedings of Robotics: Science and Systems*, 2019.

[47] J. Zhang, C. Wei, and C. Wu. Thermodynamic consistent neural networks for learning material interfacial mechanics. arXiv:2011.14172, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 2; e.g., integration of overly-complex physics models is an open challenge.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] This paper is on general methodology, and we do not think we can discuss concrete social impacts at this layer of research.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...

(a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 3 and Appendix B.
(b) Did you include complete proofs of all theoretical results? [Yes] See Appendix B.

3. If you ran experiments...
    (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
    (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 5 and Appendix E.
    (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Section 5 and Appendix F.
    (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 5 and Appendix F.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
    (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 5 and Appendix E.
    (b) Did you mention the license of the assets? [No] License of the two existing assets we used is already manifested at the original sources.
    (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
    (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
    (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
    (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
    (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
    (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]