Physics-integrated generative modeling using attentive planar normalizing flow based variational autoencoder

Sheikh Waqas Akhtar

University of Central Punjab, Lahore

sheikh.waqas@ucp.edu.pk

Abstract

Physics-integrated generative modeling is a class of hybrid or grey-box modeling in which we augment the the data-driven model with the physics knowledge governing the data distribution. The use of physics knowledge allows the generative model to produce output in a controlled way, so that the output, by construction, complies with the physical laws. It imparts improved generalization ability to extrapolate beyond the training distribution as well as improved interpretability because the model is partly grounded in firm domain knowledge. In this work, we aim to improve the fidelity of reconstruction and robustness to noise in the physics integrated generative model. To this end, we use variational-autoencoder as a generative model. To improve the reconstruction results of the decoder, we propose to learn the latent posterior distribution of both the physics as well as the trainable data-driven components using planar normalizng flow. Normalizng flow based posterior distribution harnesses the inherent dynamical structure of the data distribution, hence the learned model gets closer to the true underlying data distribution. To improve the robustness of generative model against noise injected in the model, we propose a modification in the encoder part of the normalizing flow based VAE. We designed the encoder to incorporate scaled dot product attention based contextual information in the noisy latent vector which will mitigate the adverse effect of noise in the latent vector and make the model more robust. We empirically evaluated our models on human locomotion dataset [33] and the results validate the efficacy of our proposed models in terms of improvement in reconstruction quality as well as robustness against noise injected in the model.

1 Introduction

Traditional theory-driven modeling and modern data-driven modeling approaches are usually non-overlapping but recently there has been growing interest in the merger of the two, under hybrid or grey-box modeling regime. In a broad sense this includes augmenting data-driven approach with known domain knowledge. Domain knowledge is a broad term and can refer to any information about the problem. It can be the dynamics of the physical, biological or

chemical system, behavioural aspect of an agent in a game, structural/mechanical property of a robotic system or rule, policies or constraints an autonomous system must obey while operating in an environment. The domain knowledge acts as a constraint for the model, for it to adapt itself as per the set boundaries. This is particularly significant in the presence of highly flexible and non-linear data-driven models which can easily overfit if their learning is not guided by domain knowledge. Domain-knowledge augmented data-driven models hold great promise towards robust models which have improved out-of-domain generalization capabilities. This hybrid regime can also play an important role towards model explainability because the decision or outcome of the model is semantically grounded in the domain knowledge.

Being a relatively new frontier of generative modeling, domain or (loosely speaking) physics-integrated modeling has several challenges to tackle. An important challenge is how to integrate the physics knowledge into the model learning process. Ideally, we would want to design the hybrid model such that the physics knowledge get utilized in the best possible way and would not just become redundant information in the learning process or worse cause erratic behaviour of the model. [71] [76] discussed such mechansims to integrate physics knowledge. Takeishi et.al [63] proposed a regularized learning framework which ensures effective use of the physics knowledge. This is important because in hybrid regime, it is possible that the optimizer may over-emphasize the output of data-driven model, diminishing or even completely nullifying the output of physics model. Their model showed improved generalization ability by extrapolating to out-of-distribution scenario.

Another important challenge in generative modeling is to reconstruct the high dimensional signal accurately from a low dimensional latent representation of the signal. The challenge here is to learn a latent posterior distribution from a set of limited data samples which will be able to recover the true data distribution. However, learning the true posterior latent distribution is generally intractable. We usually have to use some kind of approximation. Variational inference is an approach in which the intractable posterior distribution is approximated by a base probability distribution. Many methods have been developed which use variational inference to approximate the posterior. One such method is variational auto-encoder (VAE) which is a generative model and learns an approximate latent posterior distribution parameterized by a neural network. However the choice of the class of approximate latent posterior distribution limits the representation capacity of generative model. It is evident from a number of research efforts e.g [40] that approximate latent posterior distributions that are more faithful to the underlying structure in data, perform better. For example, if the data is a time series and has a dynamical structure associated to it, we would expect that the same structure be present in latent posterior distribution as well. Latent vectors sampled from such distribution, which harnesses this structure faithfully, would perform better than latent vectors which ignore the inherent structure in the data. There have also been studies which describe the detrimental effects of limited posterior approximation. [67] outlines two such problems. One is the under-estimation of variance of posterior distribution which can result in incorrect and unreliable predictions. The second is that the limited capacity of posterior distribution can result in biases in the MAP estimate of model parameters. A number of approaches have been developed to learn latent posteriors that are more faithful to underlying structure of data [20][14](see section related work as well).

In this work, we propose to approximate latent posterior distribution in physics integrated

VAE model using normalizing flows. Normalizing flow (NF) is a method for learning a probability distribution by transforming a base probability distribution (e.g a gaussian) through a series of invertible transformation called a flow. An advantage most relevant to generative modeling is that NF admits infinitesimal flow that is asymptotically able to recover true latent posterior distribution, overcoming the limitation of approaches like mean-field approximation in which no solution is ever able to recover the true posterior. Normalizing flow based latent posteriors have been used in generative models such as VAE with great success and its state of the art variants have achieved improved reconstruction results. Our motivation to use NF based posteriors in VAE model was to test its performance in a hybrid regime, and to evaluate how well it performs, if we approximate both data-driven and physics based latent posterior distribution using normalizing flow. We called this model NF-VAE hence forth.

Another important challenge we addressed is the presence of noise in the model. More specifically, to evaluate how the performance of model will be affected if noise is added in the feature extraction layer of the encoder. Will the noisy latent posterior be able robustly nullify the effects of noise and reconstruct the signal with high fidelity. We propose an attention based encoder architecture of NF-VAE to mitigate the effect of noise in the encoder. The idea is that if we augment the noisy latent with an additive component which is the representative of the group of latents similar to the noisy latent vector, then this would prevent the noisy latent from being too dissimilar compared to other latents of the same ilk.

2 Background

2.1 Variational Autoencoder

Variational auto-encoder (VAE) is based on amortized variational inference to approximate probability distribution $p(x)$ from which the data originated. VAE approximates data distribution $p(x)$ by a parametric distribution $p_\theta(x)$ with latent variable based generative process. Latent variables are produced inside the model and are generally not observable.

$$p_\theta(x) = \int p_\theta(x|z)p(z)dz \tag{1}$$

We assume that the prior distribution $p(z)$ on latent variable z is gaussian and posterior predictive distribution is factorized bernoulli or gaussain based on the nature of data. Per-sample parameterization is performed by a neural network called decoder. The latent posterior distribution is obtained through an amortized inference in which $p_\theta(z|x)$ is approximated by factorized gaussian distribution $q_\phi(z|x)$ and the parameters ϕ of per-sample posterior are inferred through a neural network called encoder. Both the encoder and decoder are trained end-to-end by a gradient based optimization algorithm which maximizes the sample estimate of lower bound on evidence (ELBO) given by 2

$$\frac{1}{n}\sum_{i}^{n}\log p_\theta(x_i) \geq \frac{1}{n}\sum_{i}^{n}\mathcal{L}_{\theta,\phi}(x_i) = \mathcal{L}_{\theta,\phi}$$
$$\mathcal{L}_{\theta,\phi} = \underbrace{E_{q_\phi(z|x)}\log p_\theta(x|z)}_{A} - \underbrace{KL(q_\phi(z|x)\|p(z))}_{B} \tag{2}$$

A and B in 2, respectively, are the negative reconstruction cost and the regularization term which penalizes the deviation of approximate posterior from the fixed prior $p(z)$. The gradient of 2 with respect to model parameters θ can be obtained using Monte-Carlo estimation and with respect to posterior parameters ϕ by stochastic backpropagation using reparameterization trick [27].

2.2 Normalizing Flows

Invertible networks, also called Normalizing flows [62][43], are class of likelihood-based generative models that approximate complex distributions by warping a known base distribution (e.g a gaussian noise) through an invertible/bijective function $G : \mathrm{R}^D \implies \mathrm{R}^D$. These methods use the change of variables theorem to compute exact changes in log-density of sample after going through the bijective transformation G. Given a random variable $\mathbf{z} \sim p_z(\mathbf{z})$ the log density of $\mathbf{x} = \mathbf{z}_1 = G(\mathbf{z}_0)$ follows:

$$\ln p(\mathbf{x}) = \ln p(\mathbf{z}) - \ln detJ_G(\mathbf{z}) \tag{3}$$

If we successively apply transformation map 3 on variables $\ddagger_k$ with a corresponding probability distribution $q_k(z)$, where $k \in 0, \ldots K$, we can construct an arbitrarily complex probability density given by 5:

$$\mathbf{z}_K = g_K \circ \ldots \circ g_2 \circ g_0 \tag{4}$$

$$\ln q_K(\mathbf{z}_K) = \ln q_0(\mathbf{z}_0) - \sum_{k=1}^{K} \ln \left| det \frac{\partial f_k}{\partial \mathbf{z}_{k-1}} \right| \tag{5}$$

where 4 is a shorthand notation for the composition of K transformations $g_K(g_{K-1}(\ldots))$. The path traversed by the random variables $\mathbf{z}_k$ with initial distribution $q_0(\mathbf{z}_0)$ is called the flow and the path formed by successive disbriution q_k is a normalizing flow.

The main complexity involved in computing 3 is the determinant of Jacobian which scales as LD^3, where L is number of hidden layers used and D is the dimension of hidden layers. Furthermore, computing the gradient of the Jacobian determinant also scales with $\mathcal{O}(LD^3)$ and involves computing matrix inverses that can be numerically unstable.

2.3 Planar Normalizing Flows

Rezende and Shakir [52] proposed a normalizing flow architecture with a family of bijective transformation function G of the form:

$$g(z) = \mathbf{z} + \mathbf{u}h(\mathbf{w}^T z + b) \tag{6}$$

where $\lambda = \mathbf{w} \in R^D, \mathbf{u} \in R^D, b \in \mathrm{R}$ are free parameters and $h(.)$ is a smooth element-wise non-linearity, with derivation h'. Its main advantage is the cheaper Jacobian computation which takes $O(D)$ time using the matrix determinant lemma.

$$\psi(\mathbf{z}) = h'(\mathbf{w}^T\mathbf{z} + b)\mathbf{w} \tag{7}$$

$$\left|det\frac{\partial g}{\partial \mathbf{z}}\right| = |det(\mathbf{I} + \mathbf{u}\psi(z)^T) = |1 + \mathbf{u}\psi(z)| \tag{8}$$

From 5 we conclude that density $q_K(z)$ obtained by transforming an arbitrary initial density $q_0(\mathbf{z}_0)$ through the sequence of transformation maps g_k of the form 6 is implicitly given:

$$\mathbf{z}_K = g_K \circ \ldots \circ g_2 \circ g_0 \tag{9}$$

$$\ln q_K(\mathbf{z}_K) = \ln q_0(\mathbf{z}_0) - \sum_{k=1}^{K} \ln \left|1 + \mathbf{u}_k^T \psi(\mathbf{z}_{k-1})\right| \tag{10}$$

2.4 Scaled dot product self-attention

Self-attention mechanism [69] represents an input sample as an attention-weighted sum of values of other input samples. The attention weight between two input samples is a scalar which determines how much one sample is similar to the other. The input samples may be words of a sentence, sequence of image frames in a video, values of a time series. In short, attention captures how a particular input sample is related to the other samples.

A self attention layer takes N inputs $x_1, x_2, \ldots, xN$ each of dimension $D \times 1$ and returns N output vectors of the same size. A set of values are computed for each input:

$$v_m = \beta_v + \Omega_v x_m \tag{11}$$

where $\Omega_v \in R^{D\times D}$ and $\beta_v \in R^D$ are weights and biases respectively. Then the nth output $SA_n[x_1, x_2, \ldots, x_N]$ is a weighted sum of all the values $v_1, v_2, \ldots, v_N$:

$$SA_n[x_1, x_2, \ldots, x_N] = \sum_{m=1}^{N} a[x_m, x_n] v_m \tag{12}$$

The scalar weight $a[x_m, x_n]$ is the attention that nth input pays to the mth input. The N weights $a[., x_n]$ are non-negative and sum to 1. To compute the attention weight, we apply two more non-linear transformations to the input.

$$q_n = \beta_q + \Omega_q x_n \tag{13}$$

$$k_m = \beta_k + \Omega_k x_m \tag{14}$$

where q_n and k_m are called queries and keys respectively. We can compute the dot product between queries and keys and pass the result through a softmax function:

$$a[x_m, x_n] = \texttt{Softmax}[k_m^T q_n] \tag{15}$$

$$= \frac{exp[k_m^T q_n]}{\sum_{m'=1}^{N} exp[k_{m'}^T q_n]} \tag{16}$$

So, the dot product is the measure of similarity between query and keys. Overall, attention weights are non-linear function of input. This is an example of hyper-parameter, in which one network computes the weights of another. The shared parameters of attention layer to learn are $\beta_v, \Omega_v, \beta_q, \Omega_q, \beta_k, \Omega_k$. These parameters are independent of number of inputs N.

The dot product in attention computation can have large magnitude causing inputs with large weights to dominate. Small changes in input to softmax function will have little effect on attention weight. To avoid this dot product is scaled by square root of dimension D_q of queries or keys (both have same dimensions).

Summarizing the whole process in matrix form

$$V[X] = \beta_v \mathbf{1}^T + \Omega_v X \tag{17}$$
$$Q[X] = \beta_q \mathbf{1}^T + \Omega_q X \tag{18}$$
$$K[X] = \beta_k \mathbf{1}^T + \Omega_k X \tag{19}$$
$$\tag{20}$$

where $\mathbf{1}$ is $N \times 1$ vector of ones.

Dot product self attention is computed as

$$SA[X] = V[X]\texttt{Softmax}[K[X]^T Q[X]] \tag{21}$$

Scaled dot product attention is computed as

$$SA[X] = V[X]\texttt{Softmax}\Big[\frac{[K[X]^T Q[X]]}{\sqrt{D_q}}\Big] \tag{22}$$

3 Physics Integrated generative modeling

We propose a physics-integrated variational auto-encoding architecture for generative modeling of a dynamical system. The encoder part encodes the input signal into a latent representation. This latent approximation of the original signal is sampled from a learned posterior distribution. The decoder decodes the latent variables to reconstruct the original signal. The fidelity of reconstruction is profoundly influenced by the family of distribution, from which the latent vector is sampled. VAE uses gaussian distribution to sample latent vector in order to allow for efficient inference. However, this results in poor reconstruction results. We propose to use an attentive normalizing flow based posterior approximation to improve the reconstruction results. Normalizing flow is a generative modeling approach and has previously been used for latent posterior approximation in VAE with promising results [43]. We will refer to such normalizing flow based VAE as NF-VAE.

We now present the architecture of NF-VAE, with a brief discussion on its components:

3.1 NF-VAE

We have used NF-VAE as a hybrid generative model to learn a dynamical system. The dynamical system is such that there is a known part. We know its dynamics or physics. The

other part is unknown and we don't know it dynamics. The encoder of NF-VAE produces two sets of latent variables. One for which the dynamics is known. Since we know the dynamics, the decoder of this part will just require us to solve an IVP using some numerical integrator which will take initial solution value and physics based latent vector as input. The other set of latent vector (name auxiliary variable) will belong to the unknown dynamics. Its decoder will be a neural network, mapping auxiliary latent vector to the solution of unknown ODE part in the forward process. The final output will be the sum of both decoder outputs.

3.1.1 Latent Variables and Prior

We have two types of latent variables: physics based $z_P \in \mathcal{Z}_P$ and auxiliary latent variables $z_{aux} \in \mathcal{Z}_{aux}$. $\mathcal{Z}_P$ and $\mathcal{Z}_{aux}$ are assumed to be in Euclidean space and prior distributions on z_P and z_{aux} are assumed to be multivariate normal.

$$\begin{aligned} p(z_P) &= \mathcal{N}(z_p|\mathbf{m}_P, \mathbf{\Sigma}_P) \\ p(z_{aux}) &= \mathcal{N}(z_{aux}|\mathbf{m}_{aux}, \mathbf{\Sigma}_{aux}) \end{aligned} \tag{23}$$

$\mathbf{m}_P$, $\mathbf{\Sigma}_P$ and $\mathbf{m}_{aux}, \mathbf{\Sigma}_{aux}$ are obtained using feature extraction neural networks $\texttt{MLP}_{\mathbf{m}_P}$, $\texttt{MLP}_{\mathbf{\Sigma}_P}$, $\texttt{MLP}_{\mathbf{m}_{aux}}$ and $\texttt{MLP}_{\mathbf{\Sigma}_{aux}}$ respectively.

3.1.2 Encoder

The encoder learns latent posterior distribution of z_P and z_{aux} as:

$$q_\psi(z_P, z_{aux}|x) = q_\psi(z_{aux}|x) q_\psi(z_P|x, z_{aux}) \tag{24}$$

where

$$q_\psi(z_{aux}|x) = \ln q_K(z_K^{aux}) = \ln q_0(z_0^{aux}) - \sum_{k=1}^{K} |1 + u_{k,aux}^T \psi_{k,aux}(z_{k-1}^{aux})| \tag{25}$$

$$q_0(z_0^{aux}) = \mathcal{N}(z_0^{aux}|\mathbf{m}_{aux}, \mathbf{\Sigma}_P) \tag{26}$$

$$\mathbf{m}_{aux} = \texttt{MLP}_{\mathbf{m}_{aux}}(x) \tag{27}$$

$$\mathbf{\Sigma}_{aux} = \texttt{MLP}_{\mathbf{\Sigma}_{aux}}(x) \tag{28}$$

$$q_\psi(z_P|x, z_{aux}) = \ln q_K(z_K^P) = \ln q_0(z_0^P) - \sum_{k=1}^{K} |1 + u_{k,P}^T \psi_{k,P}(z_{k-1}^{aux})| \tag{29}$$

$$q_0(z_0^P) = \mathcal{N}(z_0^P|\mathbf{m}_P, \mathbf{\Sigma}_P) \tag{30}$$

$$\tilde{x} = x + \mathbf{m}_{aux} \tag{31}$$

$$\mathbf{m}_P = \texttt{MLP}_{\mathbf{m}_P}(\tilde{x}) \tag{32}$$

$$\mathbf{\Sigma}_P = \texttt{MLP}_{\mathbf{\Sigma}_P}(\tilde{x}) \tag{33}$$

The feature extracting network produces latent prior z_0 and K normalizing flow maps from $f_0, \ldots, f_K$ of the form 7, which transform prior latent z_0 to z_K using 9. Features extraction and normalizing flow network makeup the recognition network or encoder of VAE. The parameters of recognition network are trained using stochastic backpropagation [51].

We also used mixing process to learn physics latent z_p grounded in auxiliary latent z_{aux} 31. This is a concatentation of mean of z_{aux} with x before feeding it to feature extraction network. This is important because the decoder uses both latents to reconstructs a single output. These latents cannot be completely unrelated or disjoint. The physics based latent should be grounded in auxiliary latent so that both outputs f_P and f_A of the decoder are aligned with each other, being exlusive parts of one whole thing and not some unrelated outputs.

3.1.3 Decoder

The decoder consists of two types of functions $f_p : \mathcal{Z}_p \to \mathcal{Y}_p$ and $f_{aux} : \mathcal{Z}_{aux} \to \mathcal{Y}_{aux}$. We consider functional $\mathcal{F}$ which evaluates the two functions f_p and f_{aux}. f_p, represents the numerically integrated dynamics of the physics model (an ODE) whereas f_{aux} represents the solution of the unknown part of the dynamical system as a neural network which learns to map z_{aux} to the output. The observation x is the sum of both f_p and f_{aux}. It may be a sequence of images or a time series. We assume observation has an gaussian noise with known variance σ_n^2 in it, hence it is also a gaussian:

$$p_\theta(x|z_p, z_{aux}) = \mathcal{N}(x|\mathcal{F}[f_p, f_{aux}; z_p, z_{aux}], \sigma_n^2\mathbf{I}). \tag{34}$$

$$f_p = \texttt{ODESolver}(\frac{df_p}{dt}; t_0, t_T, x_0) \tag{35}$$

$$\frac{df_p}{dt} = \texttt{MLP}_{p-decoder}(z_p) \tag{36}$$

$$f_{aux} = \texttt{MLP}_{aux-decoder}(z_{aux}) \tag{37}$$

$$\tag{38}$$

Trainable parameters of f_p and f_a are denoted by θ. We assume additive relation between f_p and f_{aux} such that $\mathcal{F}[f_p, f_{aux}; z_p, z_{aux}] = f_p + f_{aux}$. The role of f_{aux} is complementary to the physics model in this setup. However, it can be much more than that, for example, it can also work as a correction of numerical error of ode-solver or optimizer. It can also act as side information e.g. sequence of images or video of dynamical system in its operation.

3.1.4 Objective Function

Following variational principle [21], we can derive a lower bound on marginal log-likelihood. This bound is often referred to as evidence lower bound (ELBO) or negative free energy.

$$\ln p_\theta(x) = \ln \Sigma_{\mathbf{z}\sim q_\psi(z_p,z_{aux}|x)} p_\theta(x|\mathbf{z})p(\mathbf{z}) \tag{39}$$

$$= \ln \Sigma_{\mathbf{z}} \frac{q_\psi(z_p, z_{aux}|x)}{q_\psi(z_p, z_{aux}|x)} p_\theta(x|z_p, z_{aux})p(z_p, z_{aux}) \tag{40}$$

$$= \ln \Sigma_{\mathbf{z}} \frac{q_\psi(z_{aux}|x)q_\psi(z_P|x, z_{aux})}{q_\psi(z_{aux}|x)q_\psi(z_P|x, z_{aux})} p_\theta(x|z_p, z_{aux})p(z_p)p(z_{aux}) \tag{41}$$

$$\begin{aligned} = \ln \Sigma_{z_p|(z_{aux},x)}\Sigma_{z_{aux}} \frac{p(z_p)}{q_\psi(z_P|x, z_{aux})} + \ln \Sigma_{z_p|(z_{aux},x)}\Sigma_{z_{aux}} \frac{p(z_{aux})}{q_\psi(z_{aux}|x)} \\ + \ln \Sigma_{z_p|(z_{aux},x)}\Sigma_{z_{aux}} p_\theta(x|z_p, z_{aux})q_\psi(z_{aux}|x)q_\psi(z_P|x, z_{aux}) \end{aligned} \tag{42}$$

$$\geq E_{z_{aux}}[-D_{KL}\{q_\psi(z_P|x, z_{aux})\|p(z_p)\}] - D_{KL}[q_\psi(z_{aux}|x)\|p(z_{aux})] + E_{\mathbf{z}} \ln p_\theta(x|z_p, z_{aux}) \tag{43}$$

$$= \texttt{ELBO}(\theta, \psi; \mathbf{x}) \tag{44}$$

In the last inequality we used Jensen's inequality to obtain ELBO. ELBO provides a unified objective for optimization of the model with respect to latent variables. The third term of last equation is the reconstruction error. The first and second terms are KL-divergences between approximate latents z_p and z_{aux} respectively with their corresponding priors. The divergence terms act as regularizer and try to keep the learned posterior close to prior. We can maximize the ELBO and hence the log-likelihood by minimizing the divergence terms.

3.1.5 Takeiski Regularizers

Working with two different sets of latent variable for generative modeling comes with several challenges. For example, it is possible that the trainable part of the decoder which reconstructs the unknown dynamics using a neural network, dominates in such a way that it renders the known physics model completely useless. Maximizing ELBO does not guarantee that physics knowledge is being used in an effective manner. Another challenge is that the physics based latent produced by encoder z_p somehow becomes meaningless such that the reconstructed solution f_p of the physics model fluctuates around the mean pattern of data. In this situation, even if the decoder effectively uses the physics model, the optimizer still would not be able to escape the local minima. To alleviate these problems, we used two additional regularizers (namely R_{T1} and R_{T2}) proposed by Takeshi et. al (see section 3 of [63]) in the objective function. Overall, the regularized objective function to optimize is:

$$\text{minimize}_{\theta,\psi} - \mathbb{E}_{p_d(x|X)}\texttt{ELBO} + \alpha R_{T1} + \beta R_{T2} \tag{45}$$

where $p_d(x|X)$ is the empirical distribution with support on data $X := \{x_1, x_2, \ldots, x_n\}$. α and *beta* are hyperparameters to control penalization by regularizers. Their optimal values were selected on the basis of performance on validation set.

3.1.6 Learning

We used Adam optimizer [28] to learn the model.

3.2 Attentive NF-VAE

We propose a modification in NF-VAE architecture to incorporate per-sample contextual information based on scaled dot product attention mechanism. This involves an amalgamation of latent posterior with attention weighted latent posteriors. The attention weighted posteriors serves as a contextual information about the input sample, relative to other samples of the batch. Attention weighted posterior captures the relationship of posterior of a sample with posteriors of all the other samples of the batch. Incorporation of contextual attention posterior has the effect of bringing a posterior closer to the group of other similar posteriors. This will be particularly beneficial in case if the latent posterior has noise, missing values or if it is an outlier. Thus, minimizing the adverse effect of training the decoder with a noisy or outlier latent posterior. In comparison to NF-VAE, the architecture of Attentive NF-VAE has modification in the encoder only. Rest of the modules are the same as NF-VAE.

3.2.1 Encoder of Attentive NF-VAE

After getting z_p and z_{aux} from the NF-VAE encoder as described in 3.1.2, we pass each latent vector to its respective scaled dot product self-attention layer $\texttt{SA}_p$ and $\texttt{SA}_{aux}$ 2.4. We call attention weighted outputs of attention layers as z_p^{att} and z_{aux}^{att}. These attention weighted output are combined with z_p and z_{aux} as following to give us attentive latent vectors.

$$z_p^{att} = \texttt{SA}_p(z_p) \tag{46}$$

$$z_p = z_p + z_p \odot z_p^{att} \tag{47}$$

$$z_{aux}^{att} = \texttt{SA}_{aux}(z_{aux}) \tag{48}$$

$$z_{aux} = z_{aux} + z_{aux} \odot z_{aux}^{att} \tag{49}$$

where $\odot$ represent element-wise multiplication.

4 Related Work

Harnessing structure in generative modeling
There have been several studies to harness different notions of structure in the generative model, with the aim to generate data more faithful to the true data distribution and improve the reconstruction fidelity. For example, mean-field approximations of latent posterior distribution incorporates a basic form of dependency with the latent variables [6]. In [20][21], posterior distribution was specified as a mixture model with continuous latent variables. Posterior distribution with continuous variables [27][55][7][19] and discrete [66] have also been studied. Dynamical structure in latent variables was harnessed using normalizing flows in [52][12][56][31][66][30][44][29][80][19]. Graph structure [11], molecules [38][18], point cloud [79], and part-model for motion synthesis using normalizing flows [17] were also studied. Normalizing flow based posterior approximation for local [52][30][68][64] and global [37] variables have also been developed. Normalizing flows can also be used to learn posterior distribution conditioned on side information [78]. [3] used attention mechanism to build more expressive variation inference model by explicitly modeling nearby and distant interaction in the latent

space.
Based on how physics knolwedge can be utilized in the data-driven model, we can classify hybrid modeling into two main categories:
Physics-integrated hybrid modeling
These are the methods in which physics information is incorporated in the model design or architecture. Such models have mostly been studied in the context of prediction and generation for various applications [36][36][10][81][83][73][42][2][1][54][16][41][9][57][48][34][47][24]. Further details can be found in some excellent survey papers [72][77][23]. We now mention some works closest to ours in setting and construction. Takeishi et. al [63] proposed a regularized hybrid model that augmented data-driven learning in VAE with physics model. Their regularizers ensure an effective utilization of physics knowledge and prevents the output of physics model from being stuck in a local minima. Yin et.al [80] also proposed a regularized hybrid model. They regularized the norm $\|f_A\|^2$ to control the flexibility of data-driven output f_A. [82][35], although, were data-driven models, they assumed latents follow an ODE. [65] proposed a model in which latent variables, governed by hamilton mechanics, were modeled using hamiltonian neural network [15]. There have also been many studies on how to integrate both physics and data-driven output components e.g they can be combined additively [50][39][13][54][16][70][47] or can be combined in some composite way [46][36][32][10][34][25][9][5]. Integration can also be designed in such a way that f_A acts as a corrector to compensate for the inaccuracy or partial availability of physics knowledge or an unmodeled phenomena [81][58][74][83][45].
Physics-inspired hybrid modeling
These are the methods in which the physics knowledge is used to define the objective function of the model. [61][76][49][59][8][75][60][22][53]. These methods assume availability of complete physics knowledge because with only partial knowledge, the objective function will also be incomplete and hence will not perform well.

5 Experiments on Human locomotion modeling

Locomotion is the movement of body from one place to another performed through a complex interaction of neuro-muscular, skeletal and sensory system. Human locomotion mainly includes walking and running. Modeling of human locomotion is an important challenge to be able to quantify possible deviation of the walking pattern of a patient from the physiological profiles of healthy persons, in order to do objective quantitative assessment of walking abnormalities, to develop new rehabilitation protocols and assistive devices, to personlize them according to the needs of patient and to verify their efficacy over time.

5.1 Dataset

We used a subset of dataset [33] which contains kinematic, kinetic and EMG measurements of locomotion at different speeds of 50 healthy subjects (25 males, 25 females, age range: 6–72 years, body mass: 18.2–110 kg, body height: 116.6–187.5 cm). Data consisted of sequences of stride, time normalized to 100 points as a percentage of stride duration. At each time point, we extracted 3 measurements: angles of hip, knee and ankle in sagittal plane. So, each data

sample $\mathbf{x}$ is a sequence $\mathbf{x} := [\omega_1, \omega_2, \ldots, \omega_{100}] \in \mathbb{R}^{3\times 100}$ where $\omega_j := [\omega_{hip,j}, \omega_{knee,j}, \omega_{ankle,j}]^T$. A batch of N data samples will be tensor of dimension $[N \times M \times t]$, where N is the number of samples in the batch, $M = 3$ is the number of measurements at each time point and $t = 100$ is the number of time points of each sample. We used 400, 100, and 344 data samples respectively for training, validation and testing.

5.2 Models

We experimented with the following VAE models in table 1 for generative modeling on Human locomotion dataset [33]. In Ordinary VAE and Physics VAE, we used a baseline

Ord-VAE [51][27]	Ordinary VAE in which encoder neural network produces latent vector z_A. Decoder is also a neural network which reconstructs x from z_A i.e., $\mathbb{E}\mathbf{x} = f_A(z_A)$. No physics knowledge is used.
Phy-VAE [4]	Encoder neural network produces latent vector z_P. Decoder is a physics engine which takes z_p and x_0 as initial conditions and generates the solution for time t by solving a known ODE. $\mathbb{E}\mathbf{x} = f_P(z_P)$
Ord+Phy+R VAE [63]	Encoder produces two sets of latent vectors z_A and z_P using two different neural networks. Physics based latent z_P is reconstructed by a physics engine which know a part of dynamical system. Auxiliary latent z_A is reconstructed by a neural network. Both reconstructed parts are added to give the final output. $\mathbb{E}\mathbf{x} = f_P(z_P) + f_A(z_A)$. Learning is regularized to ensure meaningful, robust latent production.
NF+Phy+R VAE(proposed)	Encoder produces two sets of latent vectors z_A and z_P using two different normalizing flow based neural networks. Decoder is same as in [63]. Learning is regularized.
Attentive-NF+Phy+R VAE(proposed)	Encoder produces two sets of attentive latent vectors z_A and z_P using two different normalizing flow based neural networks. Attentive latents use contextual information capturing similarity with other clean latents. We hypothesize that embedding such contextual information in a latent makes it robust to outliers. Decoder is the same as in [63]. Learning is regularized.

Table 1: Models used in the experiments

architecture similar to methods in [51][27][4]. Direct comparison is not possible since our problem setting is different. With [63], we make a direct comparison. Our problem setting is the same and we have kept the same architecture in the decoder network for a fair comparison. Our proposed method differs with [63] in the encoder network.

5.2.1 Architecture and Training details of NF-VAE and Att-NF-VAE

Latent Variables:
We used $\mathbf{z}_P \in \mathbb{R}^2$ and $\mathbf{z}_A \in \mathbb{R}^{15}$ as physics based and auxiliary latents respectively.

Learned Prior
We use two distinct MLP networks for learned prior distribution $\mathcal{N}(\mu_P, \Sigma_P)$ and $\mathcal{N}(\mu_A, \Sigma_A)$.
Physics Prior generation:

$$\underbrace{M \times t + dim\_z_A}_{\text{Input layer}} \rightarrow \underbrace{512 \rightarrow 512 \rightarrow M \times t \rightarrow 512 \rightarrow 512 \rightarrow 512}_{\text{feature extraction layer (Feature)}}$$

$$\text{Feature} \longrightarrow 128 \rightarrow \underbrace{2}_{\mu_P \in R^2} \longrightarrow \underbrace{\mathcal{N}(\mu_P, \Sigma_P)}_{\text{Latent Generation}} \sim z_P^{\text{prior}}$$

$$\text{Feature} \longrightarrow 128 \rightarrow \underbrace{2}_{\Sigma_P \in R^2} \longrightarrow \underbrace{\mathcal{N}(\mu_P, \Sigma_P)}_{\text{Latent Generation}} \sim z_P^{\text{prior}}$$

Auxiliary Prior generation:

$$\underbrace{M \times t}_{\text{Input layer}} \rightarrow \underbrace{512 \rightarrow 512 \rightarrow 512}_{\text{feature extraction layer (Feature)}}$$

$$\text{Feature} \longrightarrow 64 \rightarrow 32 \rightarrow \underbrace{15}_{\mu_A \in R^{15}} \longrightarrow \underbrace{\mathcal{N}(\mu_A, \Sigma_A)}_{\text{Latent Generation}} \sim z_A^{\text{prior}}$$

$$\text{Feature} \longrightarrow 64 \rightarrow 32 \rightarrow \underbrace{15}_{\Sigma_A \in R^{15}} \longrightarrow \underbrace{\mathcal{N}(\mu_A, \Sigma_A)}_{\text{Latent Generation}} \sim z_A^{\text{prior}}$$

Encoder (NF-VAE)
In normalizing flow VAE (NF-VAE) encoder, we use prior latents z_P^{prior} and z_A^{prior} sampled from prior distribtions and apply K inverse transformations (i.e flows) g_k sequentially to transform them to $z_{P,K}$ and $z_{A,K}$.
NF Layer (Physics):
$z_P^{\text{prior}} \in \mathbb{R}^2 \overset{g_{P,1}}{\rightarrow} z_{P,1} \overset{g_{P,2}}{\rightarrow} z_{P,2} \ldots \overset{g_{P,K}}{\rightarrow} z_{P,K}$ where $g_{P,k}$ is defined as in 9
Similarly.
NF Layer (Auxiliary):
$z_A^{\text{prior}} \in \mathbb{R}^{15} \overset{g_{A,1}}{\rightarrow} z_{A,1} \overset{g_{A,2}}{\rightarrow} z_{A,2} \ldots \overset{g_{A,K}}{\rightarrow} z_{A,K}$ where $g_{A,k}$ is defined as in 9
Encoder (Attentive NF-VAE)
In Attentive NF-VAE encoder, we apply a scaled dot-product attention layer after NF-layer which gives us the attentive context vector 46. We incorporate this context vector with the latent vector $z_{P,K}$ and $z_{A,K}$ of NF layer as in 47 and 49 respectively, to get latents z_P and z_A.
Decoder:
In proposed models, decoder consists of two parts, one neural network based which uses z_A and outputs f_A and the other physics model based decoder which uses z_P as input and outputs f_P. Reconstructed output is $\hat{x} = f_P + f_A$.
Auxiliary Decoder f_A:
$z_A \in R^{15} \rightarrow 512 \rightarrow 512 \rightarrow f_A \in M \times t$

Physics Decoder f_P:
The decoder includes a physics engine, which takes concatenated latent z_P and initial value x_0 as input, generates the dynamics using a known physics model. Then, a solver numerically integrates the dynamics to give us the reconstructed output. We modeled ∂f_P with a trainable Hamilton equation parameterized by a neural network [15].

$$\partial f_P\Big([\mathbf{p}^T \quad \mathbf{q}^T]^T, z_P\Big) = \Big[-\frac{\partial \mathcal{H}^T}{\partial \mathbf{q}} \quad \frac{\partial \mathcal{H}^T}{\partial \mathbf{p}} \Big]^T \tag{50}$$

where $\mathbf{p} \in \mathbb{R}^{d_H}$ is a generalized position, and $\mathbf{q} \in \mathbb{R}^{d_H}$ is a generalized momentum, and $\mathcal{H} : \mathbb{R}^{d_H}$ is a Hamiltonian or total energy of the system. We take $d_H = 1$ and model $\mathcal{H}$ with an MLP with two hidden layers of size 128.
Architecture of decoder is as following:

$$\underbrace{M \times t + dim\_z_P}_{\text{Input Layer}} \rightarrow \quad \underbrace{\partial f_P}_{\text{Physics model50}} \quad \rightarrow \quad \underbrace{\int}_{\text{ODE-Solver}} \quad \rightarrow f_P \in M \times t$$

Training settings: We trained for 50 epochs with a batch size of 100. We set Adam optimizer with a learning rate= 10^{-3}, weight decay = 10^{-6} and eps = 10^{-3}. Regularization hyper-parameters α and β were set to 10^{-2} and 10^{-1} respectively. For NF-VAE and Att-NF-VAE, auxiliary and physics latent priors were transformed by 12 and 5 flows respectively to get auxiliary and physics latents.

5.2.2 Results

We report the mean absolute error (MAE) of the VAEs models on test data. Table 2 empirically demonstrate the efficacy of proposed methods over other benchmarks methods. Normalizing flow based NF-VAE with physics knowledge and regularization comes out to be the best among the lot, closely followed by Attention based NF-VAE. Physics VAE is the worst performer. Apparently, just using physics knowledge (i.e dynamcis) to reconstruct the signal is not very useful unless we integrate it with the data-driven model. Additionally, learning a rich and more expressive latent posterior distribution (NF-VAE & Att-NF VAE) is more beneficial than assuming it to be gaussian (Ord VAE).

Ord VAE	Phy VAE	Ord+Phy+R VAE	NF+Phy+R VAE(Proposed)	Att-NF+Phy+R VAE(Proposed)
0.2050	12.8470	0.4104	0.15671	0.2015

Table 2: mean absolute error (MAE) of reconstruction on human gait test data

The reconstruction result of a test sample using VAEs model is shown in fig 1.

5.2.3 Performance with noisy features

We study the effect of noise on the top 4 VAE models of table 2. We add noise by randomly zeroing some features in the last feature extraction layer. We do this by selecting first a percentage of randomly chosen samples in a batch which will be corrupted and then by

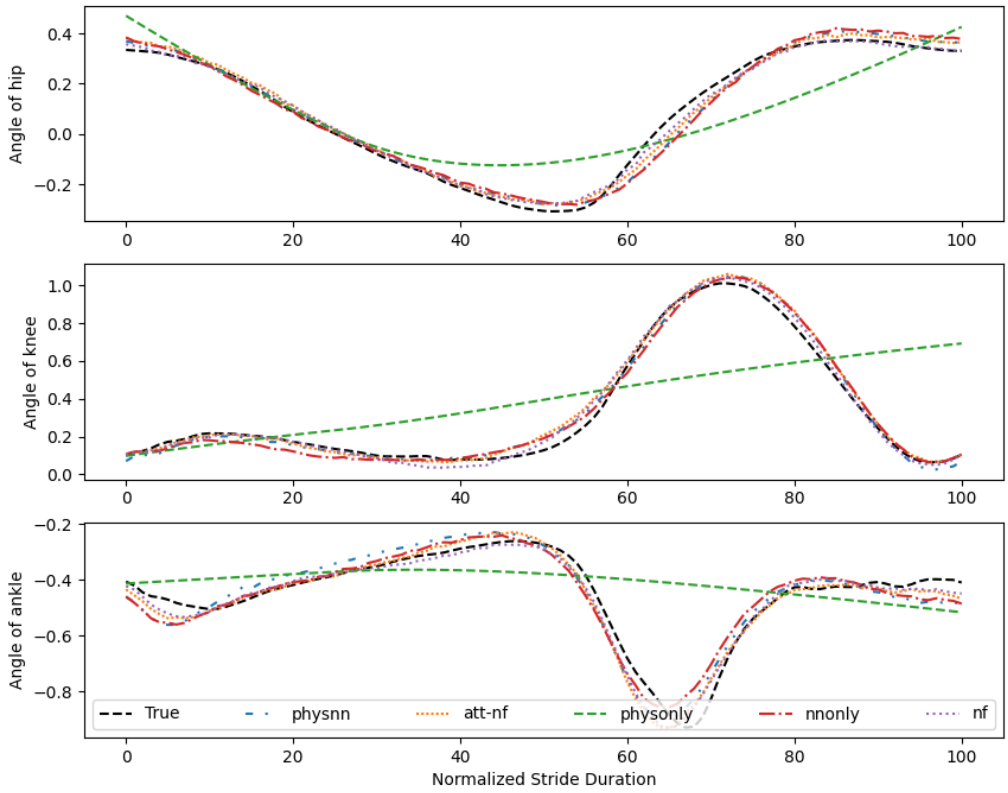

Figure 1: Reconstruction of a test sample of locomotion data.

zeroing a percentage of randomly chosen features in the selected samples. We trained the models for 10 epochs.

Method	% of noisy samples	% of noisy features in each corrupt sample				
		5%	10%	25%	50%	75%
Ord VAE	5%	1.8197	1.8229	1.8259	1.8539	1.8832
	10%	1.8204	1.8213	1.8344	1.8734	1.9125
	25%	1.8186	1.8240	1.8573	1.9185	1.9377
	50%	1.8196	1.8317	1.8826	1.9516	1.9612

Table 3: Effect of noise on MAE of Ordinary VAE

Effect of noise on Ord VAE: Observing the trend in the mae values for various noise concentrations in table 3 suggests that mae increase both by increasing the feature noise as well as the number of noisy samples. However, increase in mae by increasing feature noise is less severe when the % of noisy samples is small.

Method	% of noisy samples	% of noisy features in each corrupt sample				
		5%	10%	25%	50%	75%
Ord+Phy+R VAE	5%	1.8019	1.7934	1.7926	1.8163	1.8043
	10%	1.8094	1.7920	1.7929	1.7900	1.8093
	25%	1.7951	1.7907	1.8115	1.8298	1.8784
	50%	1.8026	1.7747	1.8560	1.9310	1.9308

Table 4: Effect of noise on MAE of Ord+Phy+R VAE

Effect of noise on Ord+Phy+R VAE: Observing the first two rows in table 4 suggest that for upto 10% of corrupt samples the mae does not increase drastically by increasing feature noise as in Ord VAE. Last two rows show gradual increase in mae on increasing feature noise, similar to Ord VAE but overall the mae values are less than Ord VAE. This establishes that Ord+Phy+R is more robust to noise than ordinary VAE.

Method	% of noisy samples	% of noisy features in each corrupt sample				
		5%	10%	25%	50%	75%
NF+Phy+R VAE	5%	1.6084	1.6975	1.6543	1.6885	1.7605
	10%	1.7210	1.7695	1.8173	1.6958	1.7433
	25%	1.5323	1.5045	1.6697	1.7263	1.7446
	50%	1.3733	1.5907	1.4378	1.8650	1.8921

Table 5: Effect of noise on MAE of NF+Phy+R VAE

Effect of noise on NF+Phy+R VAE: Similar to previous two models, in NF+Phy+R VAE (see table 5), mae increases on increasing feature noise but the mae values are less compared to both Ord and Ord+Phy+R VAE models. We can deduce that it is more robust to both Ord and Ord+Phy+R VAE models.

Method	% of noisy samples	% of noisy features in each corrupt sample				
		5%	10%	25%	50%	75%
Att-NF+Phy+R VAE	5%	1.7679	1.5756	1.5455	1.8246	1.7387
	10%	1.8367	1.8349	1.6837	1.6754	1.7942
	25%	1.7348	1.6667	1.7764	1.8517	2.0694
	50%	1.9215	1.9074	1.7199	1.8315	1.8289

Table 6: Effect of noise on MAE of Att-NF+Phy+R VAE

Effect of noise on Att-NF+Phy+R VAE: Table 6 shows that Att-NF+Phy+R VAE has a unique behaviour on increasing feature noise (in columns) compared to other models. On increasing feature noise, mae first decreases for upto 25% corruption in features, then increases. This can be attributed to the inclusion of attention based contextual information in the latents. Under moderate feature corruption (upto 25%), the contextual information added to a latent would bring it closer to the uncorrupted latents, but when feature noisy

is severe, contextual information becomes too corrupted itself, we see mae increasing as in other models. Overall, mae values are higher compared to NF+Phy+R VAE but less than Ord+Phy+R VAE.
To summarize, NF+Phy+R VAE model is the most robust to noise but Att-NF+Phy+R has a unique behaviour of decreasing mae on increasing (upto 25%) feature noise.

5.2.4 Ablation Studies

We study the effect and contribution of different factors on the performance of proposed models.

Effect of latents

We first study the contribution of latents z_P and z_A, when only one is present.

NF-VAE (only z_A)	0.1944
NF-VAE (only z_P)	13.3996
Att-NF-VAE (only z_A)	0.2347
Att-NF-VAE (only z_P)	13.9726

Table 7: Effect of Latents on MAE, when only one is present

This suggests that using only data-driven approach without any physics knowledge is much better in performance than using just physics knowledge.
When both latents are present, where one is NF or Att-NF based, other latent is an MLP.

NF-VAE $\begin{pmatrix} z_A \rightarrow \text{NF} \\ z_P \rightarrow \text{MLP} \end{pmatrix}$	0.1677
NF-VAE $\begin{pmatrix} z_A \rightarrow \text{MLP} \\ z_P \rightarrow \text{NF} \end{pmatrix}$	0.4617
Att-NF-VAE $\begin{pmatrix} z_A \rightarrow \text{Att-NF} \\ z_P \rightarrow \text{MLP} \end{pmatrix}$	0.2383
Att-NF-VAE $\begin{pmatrix} z_A \rightarrow \text{MLP} \\ z_P \rightarrow \text{Att-NF} \end{pmatrix}$	0.4640

Table 8: Effect of Latents on MAE, when one is NF/Att-NF based, other is an MLP

When both latents are present, where one is NF based and other Att-NF based.

NF-VAE $\begin{pmatrix} z_A \rightarrow \text{NF} \\ z_P \rightarrow \text{Att-NF} \end{pmatrix}$	0.1772
NF-VAE $\begin{pmatrix} z_A \rightarrow \text{Att-NF} \\ z_P \rightarrow \text{NF} \end{pmatrix}$	0.1769

Table 9: Effect of Latents on MAE, when one is NF based, other is a Att-NF based

Analyzing the results in tables 7 to 9 and table 2 suggest that NF based latents perform the best.

Effect of Regularization

We discuss the effects of two Takeishi regularizers 3.1.5 on the mae performance of proposed models. Setting a hyper-parameter = 0 nullifies the its effect. Comparing the result in table 10 with table 2 in which both regularizers were used, suggests that their inclusion improves the performance.

NF-VAE	$\alpha = 0$	0.1715
	$\beta = 0$	0.1766
	$\alpha = 0, \beta = 0$	0.1775
Att-NF-VAE	$\alpha = 0$	0.2869
	$\beta = 0$	0.2050
	$\alpha = 0, \beta = 0$	0.2087

Table 10: Effect of regularizers on MAE of proposed models

6 Conclusion

We tackled two challenges in physics-integrated generative modeling. First to improve the reconstruction performance by harnessing the dynamical structure of latent posterior distribution. Second to improve the robustness against model noise by augmenting attention based contextual information in the construction of latent posterior. Empirical evaluations validate our proposed improvements in the architecture of VAE model. In future studies, it would be interesting to investigate additional structure in the encoder (e.g latents having time-dependent dynamics). Our hypothesis is that continuous normalizing flow based posterior distribution can harness this structure. It would also be interesting to extend hybrid generative model with a complex and structured observation process, for example, partial and noisy observations [26], or observations at irregular times [55].

References

[1] 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 *In Proceedings of the 2019 IEEE International Conference on Robotics and Automation*, page 3217–322, 2019.

[2] 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 *In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*, page 3066–3073, 2018.

[3] Ifigeneia Apostolopoulou, Ian Char, Elan Rosenfeld, and Artur Dubrawski. Deep attentive variational inference. In *International Conference on Learning Representations*, 2021.

[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] A. Behjat, C. Zeng, R. Rai, I. Matei, D. Doermann, and S. Chowdhury. A physics-aware learning architecture with input transfer networks for predictive modeling. *Applied Soft Computing*, 96:106665, 2020.

[6] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians'. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

[7] E. Challis and D. Barber. Affine independent variational inference. In *In NIPS*, 2012.

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

[9] 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 *In Proceedings of the 37th International Conference on Machine Learning*, page 2402–2411, 2020.

[10] 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.

[11] Zhiwei Deng, Megha Nawhal, Lili Meng, and Greg Mori. Continuous graph flow. 2019. arXiv:1908.02436.

[12] L. Dinh, D. Krueger, and Y. Bengio. Nice: Non-linear independent components estimation. 2014. arXiv:1410.8516.

[13] M. Déchelle, J. Donà, K. Plessis-Fraissard, P. Gallinari, and M. Levy. Bridging dynamical models and deep networks to solve forward and inverse problems. In *NeurIPS workshop on Interpretable Inductive Biases and Physically Structured Learning*, 2020.

[14] S. Gershman, M. Hoffman, and D. Blei. Nonparametric variational inference. In *ICML*, 2012.

[15] S. Greydanus, M. Dzamba, and J. Yosinski. Hamiltonian neural networks. In *In Advances in Neural Information Processing Systems*, 2019.

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

[17] Gustav Eje Henter, Simon Alexanderson, and Jonas Beskow. Moglow: Probabilistic and controllable motion synthesis using normalising flows. 2019. arXiv:1905.06598.

[18] Shion Honda, Hirotaka Akita, Katsuhiko Ishiguro, Toshiki Nakanishi, and Kenta Oono. Graph residual flow for molecular graph generation. 2019. arXiv:1909.13521.

[19] Emiel Hoogeboom, Jorn W. T. Peters, Rianne van den Berg, , and Max Welling. Integer discrete flows and lossless compression. In *In Advances in Neural Information Processing Systems*, 2019.

[20] T. S. Jaakkola and M. I. Jordan. Improving the mean field approximation via the use of mixture distributions. In *Learning in graphical models*, pages 163–173. Springer, 1998.

[21] M. I. Jordan, Z. Ghahramani, and T. S. Jaakkola. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[22] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, , and L. Yang. Physics-informed machine learning. *Nature Reviews Physics*, 2021.

[23] A. Karpatne, G. Atluri, J. Faghmous, M. Steinbach, A. Banerjee, A. Ganguly, S. Shekhar, N. Samatova, , and V. Kumar. Theory-guided data science: A new paradigm for scientific discovery from data. *IEEE Transactions on Knowledge and Data Engineering*, 29(10):2318–2331, 2017.

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

[25] S. Karra, B. Ahmmed, , and M. K. Mudunuru. Adjointnet: Constraining machine learning models with physics-based codes. 2021. arXiv:2109.03956.

[26] Varun A. Kelkar, Rucha Deshpande, Arindam Banerjee, and Mark Anastasio. Ambientflow: Invertible generative models from incomplete, noisy measurements. *Transactions on Machine Learning Research*, 2024.

[27] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

[28] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations*, 2015.

[29] Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *In Advances in Neural Information Processing Systems*, page 10215–10224, 2018.

[30] Diederik P. Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *In Neural Information Processing Systems*, page 4743–4751, 2016.

[31] Ivan Kobyzev, Simon Prince, and Marcus A. Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[32] F. Lanusse, P. Melchior, and F. Moolekamp. Hybrid physical-deep learning model for astronomical inverse problems. 2019. arXiv:1912.03980.

[33] 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.

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

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

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

[37] Christos Louizos and Max Welling. Multiplicative normalizing flows for variational bayesian neural networks. In *In Proceedings of the 34th International Conference on Machine Learning*, page 2218–2227, 2017.

[38] Kaushalya Madhawa, Katushiko Ishiguro, Kosuke Nakago, and Motoki Abe. Graph-nvp: An invertible flow model for generating molecular graphs. 2019. arXiv:1905.11600.

[39] V. Mehta, I. Char, W. Neiswanger, Y. Chung, A. O. Nelson, M. D. Boyer, E. Kolemen, and J. Schneider. Neural dynamical systems: Balancing structure and flexibility in physical prediction. 2020. arXiv:2006.12682.

[40] A. Mnih and K. Gregor. Neural variational inference and learning in belief networks. In *ICML*, 2014.

[41] 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 *In Proceedings of the 2020 SIAM International Conference on Data Mining*, page 559–567, 2020.

[42] 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*, 225:225:1176–1189, 2018.

[43] George Papamakarios, Eric T. Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22:57:1–57:64, 2019.

[44] George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. In *In Advances in Neural Information Processing Systems*, page 2338–2347, 2017.

[45] Pitchforth, T. Rogers, U. Tygesen, and E. Cross. Grey-box models for wave loading prediction. *Mechanical Systems and Signal Processing*, 159:107741, 2021.

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

[47] Z. Qian, W. R. Zame, L. M. Fleuren, P. Elbers, and M. van der Schaar. Integrating expert odes into neural odes: Pharmacology and disease progression. 2021. arXiv:2106.02875.

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

[49] 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.

[50] R. Reinhart, Z. Shareef, and J. Steil. Hybrid analytical and data-driven modeling for feed-forward robot control. *Sensors*, 17(2):311, 2017.

[51] 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*, 2014.

[52] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1530–1538, Lille, France, 07–09 Jul 2015. PMLR.

[53] 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. 2020. arXiv:2006.01789.

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

[55] Yulia Rubanova, Ricky T. Q. Chen, and David Duvenaud. Latent ordinary differential equations for irregularly sampled time series. In *Advances in Neural Information Processing Systems*, page 5321–5331, 2019.

[56] T. Salimans, D.P. Kingma, and M. Welling. Markov chain monte carlo and variational inference: Bridging the gap. In *In ICML*, 2015.

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

[58] S. K. Singh, R. Yang, A. Behjat, R. Rai, S. Chowdhury, and I. Matei. Pi-lstm: Physics-infused long short-term memory metwork. In *In Proceedings of the 18th IEEE International Conference on Machine Learning and Applications*, pages 34–41, 2019.

[59] 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.

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

[61] 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.

[62] Esteban G. Tabak and Cristina V. Turner. A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164, 2013.

[63] Naoya Takeishi and Alexandros Kalousis. Physics-integrated variational autoencoders for robust and interpretable generative modeling. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[64] Jakub M. Tomczak and Max Welling. Improving variational auto-encoders using householder flow. In *NeurIPS Workshop on Bayesian Deep Learning*, 2016.

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

[66] Dustin Tran, Keyon Vafa, Kumar Krishna Agrawal, Laurent Dinh, and Ben Poole. Discrete flows: Invertible generative models of discrete data. In *In Advances in Neural Information Processing Systems*, 2019.

[67] R. E. Turner and M. Sahani. Two problems with variational expectation maximisation for time-series models. In D. Barber, T. Cemgil, and S. Chiappa, editors, *Bayesian Time series models*, chapter 5, pages 109–130. Cambridge University Press, Cambridge, 2011.

[68] Rianne van den Berg, Leonard Hasenclever, Jakub M. Tomczak, and Max Welling. Sylvester normalizing flows for variational inference. In *The 34th Conference on Uncertainty in Artificial Intelligence*, 2018.

[69] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, 2017.

[70] F. A. Viana, R. G. Nascimento, A. Dourado, and Y. A. Yucesan. Estimating model inadequacy in ordinary differential equations with physics-informed neural networks. *Computers & Structures*, 245:106458, 2021.

[71] 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. 2020. arXiv:1903.12394v2.

[72] L. von Rueden, S. Mayer, R. Sifa, C. Bauckhage, , and J. Garcke. Combining machine learning and simulation to a hybrid modelling approach: Current and future directions. In *In Advances in Intelligent Data Analysis XVIII, number 12080 in Lecture Notes in Computer Science*, pages 548–560. Springer, 2020.

[73] 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.

[74] 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.

[75] S. Wang, Y. Teng, and P. Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021.

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

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

[78] Christina Winkler, Daniel E. Worrall, Emiel Hoogeboom, and Max Welling. Learning likelihoods with conditional normalizing flows. 2019. arXiv:1912.00042.

[79] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge J. Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *In Proceedings of the International Conference on Computer Vision*, 2019.

[80] 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 *In Proceedings of the 9th International Conference on Learning Representations*, 2021.

[81] C.-C. Young, W.-C. Liu, and M.-C. Wu. A physics-aware learning architecture with input transfer networks for predictive modeling. *Applied Soft Computing*, 53:205–216, 2017.

[82] Ç. Yıldız, M. Heinonen, and H. Lähdesmäki. Ode2vae: Deep generative second order odes with bayesian neural networks. In *In Advances in Neural Information Processing Systems*, page 13412–13421, 2019.

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