Variational autoencoder for the identification of piecewise models *

Manas Mejari, Marco Forgione, Dario Piga †

November 2, 2022

Abstract

The article presents a *variational autoencoder* (VAE) tailored for the identification of hybrid *piecewise* models in input-output form. We show that using a specialized autoencoder structure, the *latent space* can provide an interpretable representation in terms of the modes of the underlying hybrid system. In particular, we use categorical encoding of the discrete latent variables whose distribution is approximated via the *encoder* neural network, characterizing a partition of the regressor space, while the *decoder* consists of a set of neural networks, each corresponding to a local submodel of the piecewise hybrid system. By employing variational Bayesian framework for inference, the constitutive terms of the *evidence lower bound* (ELBO) are derived analytically with the chosen VAE architecture. The ELBO loss consists of a reconstruction error term and a regularization term over the latent modes. This loss is optimized in order to train the encoder-decoder networks concurrently via back-propagation. The developed framework is not restricted to simple *piecewise affine* (PWA) models and it can be straightforwardly extended to general class of piecewise non-linear systems over non-polyhedral domains.

1 Introduction

In recent years, the availability of flexible open-source software with automatic differentiation capabilities along with the hardware support for parallelization has sparked renewed interest of the Automatic Control community in using deep learning tools for modeling dynamical systems [14]. Many recent contributions have developed algorithms using neural networks for the identification of *non-linear* dynamical systems [6, 15, 20, 23], characterized by continuous variables.

*This work has been supported by HASLER STIFTUNG under the project *INHALE: Interpretable Neural networks for Hybrid dynamicAL systEms.*

†The authors are with IDSIA Dalle Molle Institute for Artificial Intelligence, USI-SUPSI, Via la Santa 1, CH-6962 Lugano-Viganello, Switzerland. (e-mail: {manas.mejari, marco.forgione, dario.piga}@supsi.ch).

However, very few works have focused on developing a deep learning framework for modeling *hybrid* systems, which operate concurrently in continuous as well as discrete domains.

Hybrid models have proven to be a powerful tool to describe the behavior of many real-world systems which are characterized by different operating regions. The system can switch, either smoothly or abruptly, among these regions, giving rise to data composed of combination of system's behavior at each operating mode. For the analysis and controller synthesis of such systems, it is necessary to decompose the data into separate clusters, each representing the underlying mode and to identify system's local behavior around each operating point. To this end, many approaches have been proposed over the years to learn hybrid models from data (see, survey paper [7] for an overview), which can describe the local behaviors and uncover different operating regions of the system. Most of the conventional approaches for hybrid system identification are restricted to simple hybrid structures (such as piecewise affine maps), often requiring *linear* separability assumptions of the data clusters. These methods do not utilize powerful deep learning tools, which can potentially be exploited to recognize *non-linear* clustering patterns and more complex local dynamics. In this paper, we aim at developing a framework for learning a class of hybrid dynamical models called *piecewise models*, utilizing tools from modern deep learning.

A piecewise model consists of a set of local submodels, each defined over a specific region of the regressor space. Depending upon the parameterization of local submodels as well as its corresponding region, the following nomenclature is used in the literature [12]: (i) *piecewise affine* (PWA): affine local models over polyhedral domain; (ii) *piecewise non-linear* (PWN): non-linear submodels over polyhedral domain; (iii) *non-linearly piecewise affine* (NPWA): affine submodels over non-polyhedral domain; (iv) *nonlinearly piecewise nonlinear* (NPWN): non-linear submodels over non-polyhedral domain.

Learning piecewise models from data is an NP-hard problem [11], which requires estimating the sub-models as well as a partition of the regressor space. Over the years several heuristics have been developed for learning piecewise models (albeit, not utilizing deep learning), which include: the recently proposed optimization-based algorithm [1] for handling numeric as well as categorical data; the bounded-error approach [2]; recursive clustering-based methods [3, 4, 17], mixed-integer programming algorithms [21, 18], Bayesian inference [9, 19], among many others. The aforementioned contributions and most of the existing approaches proposed in the literature are restricted to PWA models having *affine* submodel parameterization over *polyhedral* partitioning. Very few works have addressed the identification of non-linear submodels and non-polyhedral domains, *e.g.,* in [12], kernel regression and *support vector machines* are employed to identify PWN and NPWA models. Kernel methods (for submodel estimation) combined with optimization-based strategies (for clustering) are proposed in [13, 16] to learn PWN models.

In this paper, we propose a method based on *variational autoencoder* (VAE) to learn piecewise models. The VAE was first introduced in the seminal work [10], generalizing the deterministic autoencoder [8], to the families of probabilistic

models with variational Bayesian inference. In our work, we consider specific architectures for the *encoder* and *decoder* networks in VAE, in order to take into account the *hybrid* nature of the underlying system. The encoder approximates a distribution of the discrete latent modes, characterizing a partition of the regressor space, while the decoder consists of a set of neural networks, each corresponding to a local submodel of the piecewise hybrid system. With categorical encoding of the latent modes, we compute the expectation terms in the ELBO loss analytically. This avoids sampling and re-parameterization trick employed in the conventional VAE [10], reducing the overall training cost. In summary, concurrent solution of intertwined regression and classification problems required in the hybrid piecewise model identification is obtained via simultaneous training of the encoder-decoder networks by minimizing the ELBO loss.

To the best of our knowledge, neural networks have been considered for piecewise models only in [5], where a neural network classifier to recognize the partition of the regressor space is proposed for identification of NPWA models. Our work differs from [5] in the following ways: (i) in [5], an expectation-maximization algorithm is employed in a *frequentists* setting to derive maximum-likelihood (ML) estimate of the model parameters. Our method is based on amortized variational inference in a *Bayesian* framework, which allows incorporating prior information of the latent mode probabilities via a *prior* distribution; (ii) the method in [5] is restricted to *affine* parameterization of the submodels (PWA and NPWA models), while our work is applicable to non-linear submodels using decoder neural networks, *i.e.*, identification of more general classes of piecewise models such as PWN and NPWN; (iii) the ELBO loss derived in this paper consists of a regularization term in the form of mode entropy, which allows to control potential mode collapses during training as well as to impose regular structure over the latent space. No such regularization term is considered in the ML loss in [5].

The article is organized as follows. The type of hybrid model considered in this work are described in Section 2. The identification problem is formalized in Section 3. The proposed variational autoencoder architecture tailored for piecewise models is presented in Section 4. In Section 5, we derive the evidence lower bound loss which is used to train the VAE network. Finally, the effectiveness of the proposed method is demonstrated via four case studies reported in Section 6.

2 Hybrid piecewise models

In this section, we describe types of piecewise hybrid models which we aim to identify. We consider models in input-output form with inputs denoted as $x_t \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$, and measured outputs as $y_t \in \mathcal{Y} \subset \mathbb{R}^{n_y}$.

2.1 Piecewise affine model

The PWA model $f : \mathcal{X} \to \mathcal{Y}$ is described as follows:

$$f(x_t) = \begin{cases} \theta_1' \begin{bmatrix} 1 \\ x_t \end{bmatrix} & \text{if } x_t \in \mathcal{X}_1, \\ \vdots & \vdots \\ \theta_K' \begin{bmatrix} 1 \\ x_t \end{bmatrix} & \text{if } x_t \in \mathcal{X}_K, \end{cases} \tag{1}$$

where $K \in \mathbb{N}$ is the number of *modes* (*i.e.*, the number of affine functions defining f), and $\theta_i \in \mathbb{R}^{(n_x+1)\times n_y}$ is the parameter vector associated to the i-th affine submodel. The regions $\mathcal{X}_i \subseteq \mathcal{X}$ can be either polyhedral (PWA model) or non-polyhedral (NPWA model). The set $\{\mathcal{X}_i\}_{i=1}^s$ forms a *complete* partition[1] of the regressor space $\mathcal{X}$.

Remark 1 (PWA-ARX model) *If the data is generated from a* dynamical *system $\mathcal{S}$, the regressor x_t can be defined in terms of past n_a outputs and past n_b (exogenous) input samples,* i.e., $x_t = [y_{t-1}', \cdots, y_{t-n_a}', u_{t-1}', \cdots, u_{t-n_b}']' \in \mathbb{R}^{(n_y n_a + n_u n_b)}$. *This corresponds to a* piecewise affine autoregressive with exogenous input *(PWA-ARX) model.*

2.2 Probability weighted affine model

The *probability weighted affine* (PrA) model provides a smooth relaxation of the PWA model [22], where the individual models are composed by probabilistic weighting functions as follows:

$$y_t = f_{\text{pr}}(x_t) + v_t, \tag{2a}$$

$$= \sum_{i=1}^{K} p_t^i \theta_i' \begin{bmatrix} 1 \\ x_t \end{bmatrix} + v_t, \tag{2b}$$

where p_t^i denotes the probability that the regressor x_t belongs to mode i and is parametrized by the softmax function

$$p_t^i = \frac{\exp(\eta_i' \begin{bmatrix} 1 \\ x_t \end{bmatrix})}{1 + \sum_{i=1}^{K-1} \exp(\eta_i' \begin{bmatrix} 1 \\ x_t \end{bmatrix})}, \tag{2c}$$

where $\eta_i \in \mathbb{R}^{n_x+1}$ is an unknown parameter vector characterizing the partition of the regresor space. Indeed, it is possible to transform PrA model (2) to PWA model (1), using, e.g, the rule $x_t \in \mathcal{X}_i, \Leftrightarrow \ i = \arg\max_{i=1,\ldots,K} \ p_t^i$.

[1] The collection $\{\mathcal{X}_i\}_{i=1}^K$ is a complete partition of $\mathcal{X}$ if $\bigcup_{i=1}^K \mathcal{X}_i = \mathcal{X}$ and $\mathcal{X}_i^\circ \cap \mathcal{X}_j^\circ = \emptyset$, $\forall i \neq j$, with $\mathcal{X}_i^\circ$ denoting the interior of $\mathcal{X}_i$.

2.3 Piecewise non-linear model

Piecewise non-linear models represent a non-linear extension of PWA (1) ones, and are defined as follows:

$$f(x_t) = \begin{cases} g_1(x_t;\theta_i) & \text{if } x_t \in \mathcal{X}_1, \\ \vdots & \vdots \\ g_K(x_t;\theta_K) & \text{if } x_t \in \mathcal{X}_K, \end{cases} \tag{3}$$

where $g_i : \mathcal{X}_i \to \mathcal{Y}$ are local *non-linear* maps with parameters θ_i. As stated before, the regions $\mathcal{X}_i \subseteq \mathcal{X}$ can be either polyhedral (PWN model) or non-polyhedral (NPWN model).

3 Learning problem

We consider a training dataset $\mathcal{D} = \{x_t, y_t\}_{t=1}^T$ consisting of T input-output samples generated from the following system $\mathcal{S}$

$$y_t = f(x_t) + v_t, \tag{4}$$

where $v_t \sim \mathcal{N}(0, \sigma_v^2 I_{n_y})$ is a multivariate zero-mean white Gaussian noise with diagonal covariance, statistically independent of the input x_t.

3.1 Piecewise regression problem

Given a dataset $\mathcal{D}$, the piecewise regression problem entails the following tasks:

T1 Computation of the parameters $\Theta = [\theta_1, \cdots, \theta_K]$ defining the local affine functions in the PWA map f in (1); or equivalently f_{pr} in (2b); or equivalently functions $\{g_i(x_t;\theta_i)\}_{i=1}^K$ in (3);

T2 Classification of the regressors x_t into clusters and subsequent characterization of the partition $\{\mathcal{X}_i\}_{i=1}^K$ of the regressor space.

In this work, we fix the number of modes K a-priori. In case the parameter K is unknown, it can be chosen via cross validation, trading-off between accuracy *vs* model complexity.

3.1.1 Active mode:

In this paper, we will make use of a K dimensional discrete latent variable z_t, termed as the *active mode* at time t, having a 1-of-K representation, such that

$$z_t^i = \begin{cases} 1 & \text{if } \ x_t \in \mathcal{X}_i, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Thus, the i-th component z_t^i of the vector z_t is 1 if and only if the regressor x_t belongs to the i-th region $\mathcal{X}_i$. The sequence of discrete active modes $\{z_t\}_{t=1}^T$, dictates the overall clustering of the regressor vectors $\{x_t\}_{t=1}^T$ to corresponding regions, which can be used to compute a partition of the regressor space $\mathcal{X}$. Note that the mode sequence $\{z_t\}_{t=1}^T$ is *unknown* and is to be estimated.

3.2 Bayesian problem formulation

In the rest of the paper, we consider a *Bayesian* setting in which it is assumed that the output samples in $\mathcal{D}$ are generated from an (unknown) true distribution $y_t \sim p_\Theta(y_t|x_t)$ with parameters Θ of the piecewise hybrid system. The active mode z_t in (5) represents latent encoding of the output y_t, which is assumed to be jointly distributed with y_t according to an unknown joint distribution $p_\Theta(z_t, y_t|x_t)$. In other words, the data generating system is modelled as $p_\Theta(y_t|x_t) = \sum_i^K p_\Theta(z_t^i, y_t|x_t) = \sum_i^K p_\Theta(y_t|z_t^i, , x_t)p_\Theta(z_t^i|x_t)$. Accordingly, the posterior over the latent modes is given by $p_\Theta(z_t|y_t, x_t) = \frac{p_\Theta(y_t|x_t,z_t)p_\Theta(z_t|x_t)}{p_\Theta(y_t|x_t)}$, where $p_\Theta(y_t|x_t, z_t)$ and $p_\Theta(z_t|x_t)$ denote the *likelihood* and the *prior*, respectively.

With this setting, the piecewise regression problem is formalized as follows:

Problem 1 *We aim to estimate the parameters Θ of the piecewise model defining the data-generating distribution $p_\Theta(y_t|x_t)$ (corresponding to task **T1**) and to estimate the unknown posterior distribution $p_\Theta(z_t|y_t, x_t)$ over the latent modes, consequently identifying the mode sequence $\{z_t\}_{t=1}^N$ which characterizes a partition of the regresor space (i.e., task **T2**).*

4 Amortized variational inference

For inference of the piecewise models, we adopt the *variational autoencoder* (VAE) introduced in [10]. Here, we briefly describe the concepts employed in VAE. Let y be the observed data and z denote the latent variables. In a *Bayesian* formalism, the probabilistic model is characterized by the likelihood $p_\theta(y|z)$ and a prior $p_\theta(z)$ over the latent variables, with model parameters θ. Then, the posterior distribution $p_\theta(z|y)$ over z is obtained through Bayes rule: $p_\theta(z|y) = \frac{p_\theta(y|z)p_\theta(z)}{p_\theta(y)}$. However, often the evidence $p_\theta(y)$ is intractable to compute, and consequently the posterior distribution $p_\theta(z|y)$ is not directly accessible. VAE overcomes the intractability of the posterior by introducing a variational distribution $q_\phi(z)$ parameterized via a neural network (viz. *amortized* variational inference), chosen in order to approximate the true unknown posterior $p_\theta(z|y)$. In particular, the parameters of $q_\phi(z)$ are given by a (probabilistic) *encoder* network, while the *decoder* neural network characterizes the likelihood $p_\theta(y|z)$. The goal is then to minimize the KL divergence between the variational distribution and the true posterior $\mathrm{D}_{\mathrm{KL}}(q_\phi(z)||p_\theta(z|y))$ (in order to enforce regularization over latent space) and to maximize log marginal likelihood $\log(p_\theta(y))$ (for learning model parameters θ), i.e., training the encoder-decoders networks by maximizing the following loss over the network parameters ϕ, θ,

$$\mathcal{L}_{\theta,\phi}(y) = \log(p_\theta(y)) - \mathrm{D}_{\mathrm{KL}}(q_\phi(z)||p_\theta(z|y)).$$

Note that both the terms above are intractable to compute. Nonetheless, the loss $\mathcal{L}_{\theta,\phi}(y)$ can be equivalently written as

$$\mathcal{L}_{\theta,\phi}(y) = \mathbb{E}_{z\sim q_\phi(z)}\left[\log(p_\theta(y|z))\right] - \mathrm{D}_{\mathrm{KL}}(q_\phi(z)||p_\theta(z)),$$

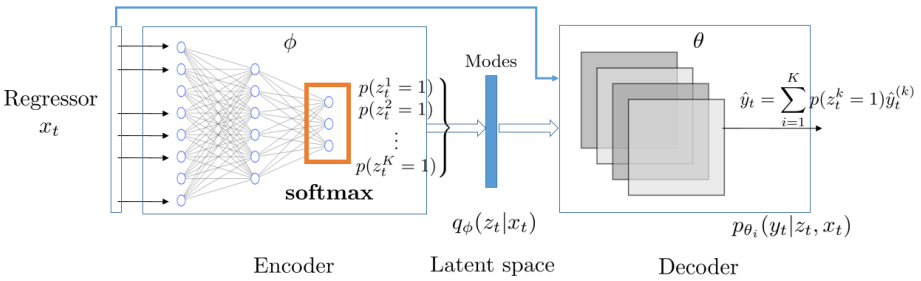

Figure 1: Autoencoder architecture for PWA regression.

which is the so called *evidence lower bound* (ELBO) loss as $\log(p_\theta(y)) \geq \mathcal{L}_{\theta,\phi}(y)$.

The first term in the ELBO loss represents the 'reconstruction error' while the second term is the KL divergence between the prior and the approximate posterior, both of which can now be computed. In order to perform back-propagation through both the encoder and decoder at once, the *re-parameterization trick* is used which allows generating latent samples $z \sim q_\phi(z)$, and thus, to compute gradients of the expectations in the ELBO loss.

In this work, we adapt and specialize the VAE concept for piecewise regression. In particular, the encoder characterizes a partition of the regressor space by recognizing active modes z_t, while the weights and biases of the decoder network represent the parameters Θ of the local submodels. In order to the compute posterior over the discrete latent modes z_t and to estimate the parameters Θ, we derive the corresponding ELBO loss for the piecewise models. By parameterizing encoder posterior with a categorical distribution, the expectation terms in the ELBO loss can be computed analytically, and thus, we avoid sampling and re-parameterization trick employed in the conventional VAE, reducing the computational effort during training.

4.1 VAE architecture for piecewise regression

We consider VAE network architecture shown in Fig. 1.

Encoder:

The encoder $\mathcal{NN}_e(x_t; \phi)$ is a feed-forward neural network with trainable parameters ϕ and with the regressor x_t fed as input feature. The last layer of the encoder consists of a *softmax* activation function taken over K modes. Thus, the output of the encoder $\mu_t \in \mathbb{R}^K$ represents the probability of each mode at time t, given by

$$\mu_t = \mathcal{NN}_e(x_t; \phi), \tag{6}$$

where

$$\mu_t^i = p(z_t^i = 1), \quad \mu_t^i \in [0, 1], \quad i = 1, \ldots, K, \quad \sum_{i=1}^{K} \mu_t^i = 1$$

with z_t being the discrete latent variable characterizing the active mode as defined in (5).

Decoder:

For PWA model, we define a decoder consisting of K independent neural networks, each corresponding to an affine submodel in the PWA map (1). Each of the K network consists of a single *linear* layer, weights (and biases) of which correspond to the parameters $\theta_i \in \mathbb{R}^{n_x}, i = 1, \ldots, K$ of the local affine function. The feature inputs to the i-th network are the regressor x_t and the probability of the i-th mode μ_t^i obtained from the encoder. The output of the i-th network is then given by $\hat{y}_t^i = \hat{\theta}_i^{'} \begin{bmatrix} \mu_t^i \\ x_t \end{bmatrix} + b_i$ with $\hat{\theta}_i^{'}$ and b_i denoting the weights and bias of the network respectively. Given that the current active mode is i with probability $\mu_t^i = 1$, the i-th network's weights and biases correspond to the parameters θ_i of the affine submodel in (1).

The decoder output is the weighted sum of the individual network's outputs given by

$$\hat{y}_t = \sum_{i=1}^{K} \mu_t^i \theta_i^{'} \begin{bmatrix} 1 \\ x_t \end{bmatrix} \tag{7}$$

This is equivalent to the probability weighted affine (PrA) model (2) with probabilities given by the encoder. As noted before, such PrA model provides a smooth relaxation of PWA model in (1), such that the deterministic partition is replaced by probabilistic boundaries. This particular structure of the decoder along with categorical parameterization of the encoder distribution allows us to compute the expected likelihood over latent space analytically.

Alternatively, in order to learn PWN and NPWN models, we can consider a *non-linear* decoder consisting of K independent feed-forward MLP neural networks having weights θ_i, with inputs x_t, μ_t^i and output $\hat{y}_t^i = \mathcal{NN}_d(x_t, \mu_t^i; \theta_i)$. Each network corresponds to local *non-linear* submodel and the decoder output is given by the probability weighted sum of K non-linear network outputs, $\hat{y}_t = \sum_{i=1}^{K} \mu_t^i \mathcal{NN}_d(x_t, \mu_t^i; \theta_i)$.

5 ELBO loss

In order to train the VAE for piecewise models described above, in this section, we derive the *evidence lower bound* loss $\mathcal{L}(\phi, \Theta)$ to be optimized over the encoder and decoder parameters.

5.1 Prior

Let us define the prior over the latent modes z_t as the following categorical distribution,

$$p(z_t|x_t) = \prod_{i=1}^{K} (\pi_t^i)^{z_t^i} \tag{8}$$

where $\pi_t^i \in [0,1]$ is the prior probability of the i-th mode, *i.e.*, $\pi_t^i = p(z_t^i = 1)$. Recall that z_t is K dimensional 1-of-K encoded categorical variable with the property that exactly one element has value 1 and the others have the value 0, thus, probability mass function $p(z_t|x_t) = p(z_t^i = 1)$ for the *active* mode i.

This allows incorporating prior knowledge about the initial clustering pattern of the system. The prior probabilities can also be estimated via unsupervised k-means algorithm or estimating a Gaussian mixture model. Possible choices for prior distribution are provided in the following. *Uniform prior:* We can set $\pi_t^i = \frac{1}{K}$, *i.e.*, $p(z_t|x_t) = \frac{1}{K}$, to weigh all modes equally.

Dirichlet hyperprior: Let $\alpha = [\alpha_1 \ldots, \alpha_K]$ be concentration hyperparameter of a Dirichlet distrbution $\mathrm{Dir}(K|\alpha)$. The prior probabilities are then given by

$$\pi_t = [\pi_t^1, \ldots, \pi_t^K] \sim \mathrm{Dir}(K|\alpha) \sim \frac{1}{B(\alpha)} \prod_{i=1}^{K} (\pi_t^i)^{\alpha_i - 1},$$

The choice of hyperparameter α determines the priorities assigned to a specific mode.

5.2 Likelihood

For PWA model structure, the *likelihood* of the output observation at time t is given as follows

$$\begin{aligned} p_\Theta(y_t|z_t, x_t) = \; & p_{\theta_i}(y_t|x_t, z_t^i = 1) \\ & = \mathcal{N}(y_t; \theta_i^{'} \left[\begin{smallmatrix} 1 \\ x_t \end{smallmatrix} \right], \sigma_v^2 I_{n_y}) \\ & = \prod_{i=1}^{K} (\mathcal{N}(y_t; \theta_i^{'} \left[\begin{smallmatrix} 1 \\ x_t \end{smallmatrix} \right], \sigma_v^2 I_{n_y}))^{z_t^i}, \end{aligned} \tag{9}$$

where we recall, σ_v^2 is the variance of the measurement noise. Since the noise samples v_t in (4) are assumed to be i.i.d., the overall likelihood is composed of product over the likelihood of individual observations given as follows,

$$p_\Theta(y_1, \cdots, y_T|z_1, \cdots, z_T, x_1, \cdots, x_N) = \prod_{t=1}^{T} \prod_{i=1}^{K} (\mathcal{N}(y_t; \theta_i^{'} \left[\begin{smallmatrix} 1 \\ x_t \end{smallmatrix} \right], \sigma_v^2 I_{n_y}))^{z_t^i} \tag{10}$$

The likelihood for PWN and NPWN model class (3) can be derived in a similar manner.

5.3 Variational distribution

The encoder provides a distribution over discrete latent variable z_t parameterized as the following categorical distribution,

$$q_\phi(z_t|x_t) = \prod_{i=1}^{K} (\mu_t^i)^{z_t^i} \tag{11}$$

where $\mu_t^i \in [0,1], \;\; i = 1, \dots K$ are the probabilities given by the encoder output (see (6)) and ϕ are the weights of the encoder network. Note that the true unknown posterior $p_\Theta(z_t|x_t, y_t)$ is approximated by the variational distribution $q_\phi(z_t|x_t)$ given by the encoder.

5.4 Evidence lower bound

Next, we derive the expression for the evidence lower bound. We recall that the Kullback–Leibler (KL) divergence between the distribution $q_\phi(z_t|x_t)$, given by the probabilistic encoder network and the true (unknown) posterior $p_\Theta(z_t|y_t, x_t)$ can be written as,

$$\begin{aligned} &\mathrm{D}_{\mathrm{KL}}(q_\phi(z_t|x_t)||p_\Theta(z_t|x_t, y_t)) = \\ &\log(p_\Theta(y_t|x_t)) + \mathrm{D}_{\mathrm{KL}}(q_\phi(z_t|x_t)||p(z_t|x_t)) - \mathbb{E}_{z_t \sim q_\phi(z_t|x_t)}\left[\log p_\Theta(y_t|z_t, x_t)\right] \end{aligned} \tag{12}$$

where $p_\Theta(y_t|x_t)$ is the *marginal* likelihood or the *evidence*. Note that the left hand side of (12) is the KL divergence between the two distributions, it is always non-negative. Thus, the evidence is lower bounded by the following term,

$$\log(p_\Theta(y_t|x_t)) \geq \mathbb{E}_{z_t \sim q_\phi(z_t|x_t)}\left[\log p_\Theta(y_t|z_t, x_t)\right] - \mathrm{D}_{\mathrm{KL}}(q_\phi(z_t|x_t)||p(z_t|x_t)) \tag{13}$$

The right hand side of (13) is the *evidence lower bound* (ELBO) of the data sample at t.

Re-writing (12), we define the ELBO loss $\mathcal{L}_{\Theta,\phi}(x_t, y_t)$ as follows

$$\begin{aligned} &\mathcal{L}_{\Theta,\phi}(x_t, y_t) \\ &= \log(p_\Theta(y_t|x_t)) - \mathrm{D}_{\mathrm{KL}}(q_\phi(z_t|x_t)||p_\Theta(z_t|x_t, y_t)) && \text{(14a)} \\ &= \mathbb{E}_{z_t \sim q_\phi(z_t|x_t)}\left[\log p_\Theta(y_t|z_t, x_t)\right] - \mathrm{D}_{\mathrm{KL}}(q_\phi(z_t|x_t)||p(z_t|x_t)) && \text{(14b)} \end{aligned}$$

Note that from (14a), maximizing the ELBO loss $\mathcal{L}_{\Theta,\phi}$ implies maximization of the log marginal likelihood $\log(p_\Theta(y_t|x_t))$ for model learning from data and forcing the approximate posterior $q_\phi(.)$ towards the true one by minimizing the KL distance $\mathrm{D}_{\mathrm{KL}}(q_\phi(\cdot)||p_\Theta(z_t|x_t, y_t)$, in order to have regular latent structure. However, both of these terms are intractable to compute. Nonetheless, this objective is achieved by considering the ELBO loss given in (14b), where the constitutive terms can be computed analytically. Thus, the goal is to maximize the ELBO loss $\mathcal{L}_{\Theta,\phi}$ given in (14b) w.r.t. both the encoder parameters ϕ and the decoder parameters Θ.

We now compute both the terms of the loss $\mathcal{L}_{\Theta,\phi}$ in (14b) for PWA model.

Reconstruction error: The first term in (14b) is the averaged log likelihood of the model over the approximate posterior distribution. This term is equivalent to "reconstruction or fitting error" in an autoencoder which drives learning of the model from data.

Substituting the expression of the likelihood (9), we have

$$
\begin{aligned}
&\mathbb{E}_{z_t \sim q_\phi(z_t|x_t)} \left[\log p_\Theta(y_t|z_t, x_t)\right] \\
&= \mathbb{E}_{z_t \sim q_\phi(z_t|x_t)} \left[\log \prod_{i=1}^{K} (\mathcal{N}(y_t; \theta_i^{'} \begin{bmatrix} 1 \\ x_t \end{bmatrix}, \sigma_v^2 I_{n_y}))^{z_t^i}\right] \\
&= \mathbb{E}_{z_t \sim q_\phi(z_t|x_t)} \left[\sum_{i=1}^{K} z_t^i \log(\mathcal{N}(y_t; \theta_i^{'} \begin{bmatrix} 1 \\ x_t \end{bmatrix}, \sigma_v^2 I_{n_y}))\right] \\
&= \mathbb{E}_{z_t \sim q_\phi(z_t|x_t)} \left[\frac{-1}{2\sigma_v^2} \sum_{i=1}^{K} z_t^i \left\| y_t - \theta_i^{'} \begin{bmatrix} 1 \\ x_t \end{bmatrix} \right\|^2 + \text{const}\right] \\
&= \frac{-1}{2\sigma_v^2} \sum_{i=1}^{K} \mu_t^i \left\| y_t - \theta_i^{'} \begin{bmatrix} 1 \\ x_t \end{bmatrix} \right\|^2 + \text{const} \qquad (15)
\end{aligned}
$$

Given the categorical approximate posterior distribution $q_\phi(z_t|x_t)$ of the encoder in (11), the last equality in (15) is obtained using the fact that $\mathbb{E}_{z_t \sim q(z_t|x_t)} \left[z_t^i\right] = \mu_t^i$ for the categorical variable z_t.

Equivalently, for the probability weighted affine model (PrA) of the decoder, we have,

$$
\mathbb{E}_{z_t \sim q_\phi(z_t|x_t)} \left[\log p_\Theta(y_t|z_t, x_t)\right] = \frac{-1}{2\sigma_v^2} \left\| y_t - \sum_{i=1}^{K} \mu_t^i \theta_i^{'} \begin{bmatrix} 1 \\ x_t \end{bmatrix} \right\|^2, \qquad (16)
$$

and for piecewise non-linear models (PWN and NPWN), we have,

$$
\mathbb{E}_{z_t \sim q_\phi(z_t|x_t)} \left[\log p_\Theta(y_t|z_t, x_t)\right] = \frac{-1}{2\sigma_v^2} \left\| y_t - \sum_{i=1}^{K} \mu_t^i \mathcal{NN}_d(x_t, \mu_t^i; \theta_i) \right\|^2, \qquad (17)
$$

where $\mathcal{NN}_d(\cdot)$ is the non-linear MLP decoder network.

Regularization over latent variables: The second term in (14b) is KL divergence between the (approximate) posterior (11) and prior over latent variables defined in (8). This acts as a regularization term and provides a consistent structure to the latent space.

Substituting (8) and (11), in the second term of (14b) we have,

$$
\begin{aligned}
&\mathrm{D}_{\mathrm{KL}}(q_\phi(z_t|x_t)||p(z_t|x_t)) \\
&= \mathbb{E}_{z_t\sim q_\phi(z_t|x_t)}\left[\log\left(\frac{q_\phi(z_t|x_t)}{p(z_t|x_t)}\right)\right] \\
&= \mathbb{E}_{z_t\sim q_\phi(z_t|x_t)}\left[\log\left(\frac{\prod_{i=1}^{K}(\mu_t^i)^{z_t^i}}{\prod_{i=1}^{K}(\pi_t^i)^{z_t^i}}\right)\right] \\
&= \mathbb{E}_{z_t\sim q_\phi(z_t|x_t)}\left[\sum_{i=1}^{K} z_t^i\log(\mu_t^i) - z_t^i\log(\pi_t^i)\right] \\
&= \sum_{i=1}^{K}\mu_t^i\log\left(\frac{\mu_t^i}{\pi_t^i}\right) \qquad (18)
\end{aligned}
$$

The last equality follows as $\mathbb{E}_{z_t\sim q_\phi(z_t|x_t)}\left[z_t^i\right] = \mu_t^i$.

For a uniform prior, we have $\pi_t^i = \frac{1}{K}$, thus the KL term simplifies to

$$
\sum_{i=1}^{K}\mu_t^i\log\left(\frac{\mu_t^i}{\pi_t^i}\right) = \sum_{i=1}^{K}\mu_t^i\log\left(\mu_t^i\right) - \log(1/K)
$$

The first term can be interpreted as the (negative) *mode entropy* which allows to control potential mode collapse during the training. We note that $\lim_{\mu_t^i\to 0}\ \mu_t^i\log\left(\mu_t^i\right) = 0$, thus, the regularization term is set to 0 for non-active modes having small probability values.

Substituting the expressions of fitting error (15) and regularization loss (18) in the ELBO (14b) and averaging over all data samples, we get the following loss function

$$
\mathcal{L}_{\Theta,\phi}(\{x_t, y_t\}_{t=1}^{T}) = \frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{K}\left(-\mu_t^i\left\|y_t - \theta_i^{'}\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right]\right\|^2 - \lambda\mu_t^i\log\left(\frac{\mu_t^i}{\pi_t^i}\right)\right), \quad (19)
$$

where λ is a regularization hyper-parameter to achieve trade-off between reconstruction error and regularization loss. The encoder and decoder networks are trained at once to compute the parameters ϕ, Θ via back-propagation of the loss (19) over training data. The estimate ϕ defines an encoder which approximates the posterior $p_\Theta(z_t|y_t, x_t)$ characterizing the partition of the regressor space, while decoder parameters Θ give the parameters of the local submodels of the piecewise hybrid model, thus, solving Problem 1.

6 Numerical examples

The effectiveness of the proposed technique is evaluated on four examples, namely, PWA function regression, a benchmark PWA-ARX identification, a nonlinearly piecewise NPWA-ARX identification example and PWNL model

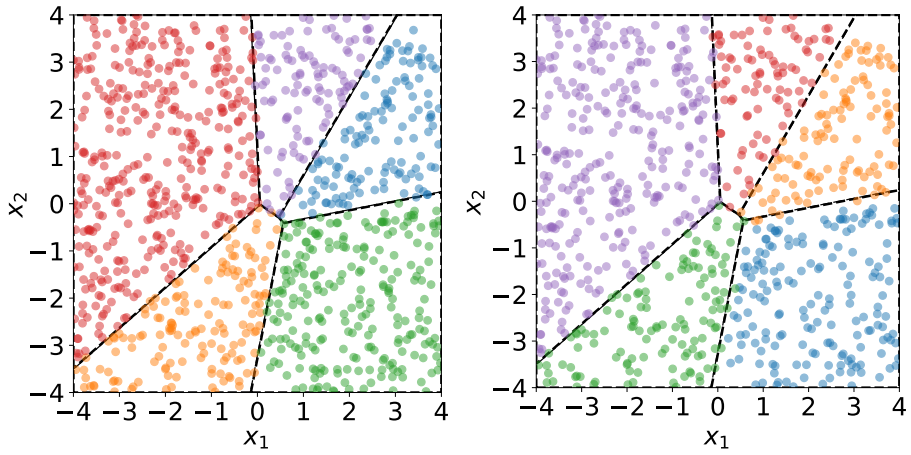

Figure 2: True partition induced by (21) (left panel) and estimated clusters (right panel).

learning. All computations are carried out on an i7 1.9-GHz Intel core processor with 32 GB of RAM. The codes are implemented with `PyTorch 1.12.1` for the training of the neural networks.

The quality of the trained models is assessed in terms of their ability to recognize the partition of the regressor space and their predictive capability quantified via R^2 score computed on a test dataset:

$$R^2 = \left(1 - \frac{\sum_{t=1}^{T}(y_t - \hat{y}_t)^2}{\sum_{t=1}^{T}(y_t - \bar{y})^2}\right) \times 100\ \%, \tag{20}$$

where y is the measured output, $\hat{y}$ is the estimated model output and $\bar{y}$ is the average value of y, *i.e.*, $\bar{y} = \frac{1}{T}\sum_{t=1}^{T} y_t$.

For all the examples reported in this section, we have chosen a uniform prior $p(z_t) = \frac{1}{K}$. The number of hidden layers and number of nodes in each layer of encoder-decoder networks are tuning hyper-parameters of the VAE, which are chosen via cross-validation.

6.1 Example 1: Piecewise affine function

In this example, we consider the following PWA data-generating function characterized by $K = 5$ modes,

$$f(x) = \max\left\{ \begin{bmatrix} 0.8031 \\ 0.0219 \\ -0.3227 \end{bmatrix}' \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}, \begin{bmatrix} 0.0942 \\ -0.5617 \\ -0.1622 \end{bmatrix}' \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}, \right.$$
$$\left. \begin{bmatrix} 0.9462 \\ -0.7299 \\ -0.7141 \end{bmatrix}' \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}, \begin{bmatrix} -0.4799 \\ 0.1084 \\ -0.1210 \end{bmatrix}' \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}, \begin{bmatrix} 0.5770 \\ 0.1574 \\ -0.1788 \end{bmatrix}' \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix} \right\}. \tag{21}$$

Table 1: True *vs* estimated model parameters (weights and biases of the decoder network).

Mode	True	Estimated
1	$[\,0.8031,\ 0.0219,\ -0.3227\,]$	$[\,0.8021,\ 0.0226,\ -0.3209\,]$
2	$[\,0.0942,\ -0.5617,\ -0.1622\,]$	$[\,0.0938,\ -0.5613,\ -0.1614\,]$
3	$[\,0.9462,\ -0.7299,\ -0.7141\,]$	$[\,0.9462,\ -0.7298,\ -0.7141\,]$
4	$[\,-0.4799,\ 0.1084,\ -0.1210\,]$	$[\,-0.4798,\ 0.1083,\ -0.1205\,]$
5	$[\,0.5770,\ 0.1574,\ -0.1788\,]$	$[\,0.5777,\ 0.1567,\ -0.1768\,]$

We gather a dataset consisting of 1000 target-feature samples from (21), with features $x \in \mathbb{R}^2$ uniformly distributed in the box $[\,-4,\ 4\,] \times [\,-4,\ 4\,]$. We use $N = 800$ samples for training and remaining 200 samples as a test dataset. In Fig. 2 (left panel), the true partition induced by the PWA function (21) is shown.

The encoder is chosen as a single-hidden-layer linear network with 8 neurons. The output layer consists of 5 nodes, corresponding to the number of modes in (21) with a *softmax* activation. The decoder consists of 5 linear networks, each with a single layer corresponding to the parameters of the affine functions in (21). The VAE is trained by maximizing the loss function (19) setting $\lambda = 1 \cdot 10^{-4}$, by employing *Adam* algorithm with a learning rate $1 \cdot 10^{-2}$ and number of stochastic gradient descent (SGD) iterations fixed to $20 \cdot 10^3$. The time required for training is 79.6 sec.

The final clustering of the features obtained with the trained VAE is shown in the right panel of Fig 2. It can be observed that in the noise-free setting, the model is able to recognize the underlying partition of the feature space very accurately.

Weights and biases of the decoder network are reported in Table 1. The weights closely match with the true parameters of the affine function. Thus, VAE is able to accurately identify the PWA data-generating function, both in terms of recognizing the underlying partition as well as recovering the true model parameters.

Next, the predictive capability and robustness of the model is evaluated using a *noisy* dataset. In particular, the feature data $x_{1,t}, x_{2,t}$ and target outputs y_t are corrupted by zero-mean white Gaussian noise with variance σ^2, *i.e.*, $\epsilon_t^{x_1}, \epsilon_t^{x_2}, \epsilon_t^{y} \sim \mathcal{N}(0, \sigma^2)$. In Table 2, we report the R^2 scores of the trained model computed on the test data for varying noise-levels $\sigma = \{0.0, 0.01, 0.1, 0.2\}$. In the ideal case of $\sigma = 0.0$, the prediction accuracy is 100%, while high R^2 scores are achieved even in the presence of high noise levels. It is evident that the learned model is robust against the noise and is able to reconstruct the output with a high accuracy.

Table 2: R^2 scores with noisy data $(x_i + \epsilon^{x_i}, y + \epsilon^y)$ where $\epsilon^{x_i}, \epsilon^y \sim \mathcal{N}(0, \sigma^2)$, i = 1,2.

Test	$\sigma = 0.00$	$\sigma = 0.01$	$\sigma = 0.10$	$\sigma = 0.20$
R^2 (%)	100.00 %	99.94 %	98.58 %	94.63 %

6.2 Example 2: Identification of PWA-ARX model

In this case study, we consider the following benchmark PWA-ARX system [2], characterized by $K = 3$ modes

$$y_t = \begin{cases} \begin{bmatrix} -0.4 & 1 & 1.5 \end{bmatrix} x_t + e_t, & \text{if } \begin{bmatrix} 4 & -1 & 10 \end{bmatrix} x_t < 0, \\ \begin{bmatrix} 0.5 & -1 & -0.5 \end{bmatrix} x_t + e_t, & \text{if } \begin{bmatrix} 4 & -1 & 10 \end{bmatrix} x_t \geq 0, \\ & \quad \& \begin{bmatrix} 5 & 1 & -6 \end{bmatrix} x_t \leq 0, \\ \begin{bmatrix} -0.3 & 0.5 & -1.7 \end{bmatrix} x_t + e_t, & \text{if } \begin{bmatrix} 5 & 1 & -6 \end{bmatrix} x_t > 0, \end{cases} \tag{22}$$

with regressor $x_t = \begin{bmatrix} y_{t-1} & u_{t-1} & 1 \end{bmatrix}'$. The training dataset consist of $T = 6000$ input/output samples gathered from the system (22), with the input signal u generated from a uniform random distribution taking values in $[-4, 4]$. The noise e_t corrupting the output is a zero-mean white Gaussian noise $e_t \sim \mathcal{N}(0, \sigma^2)$ with $\sigma = 0.5$, corresponding to the *signal-to-noise ratio* $\text{SNR} = 10 \log \frac{\sum_{t=1}^{T} (y_t - e_t)^2}{\sum_{t=1}^{T} e_t^2} =$ 16 dB.

For estimation with VAE, we consider a single-hidden-layer network architecture for the encoder with *relu* activation function in the hidden layer and *softmax* activation at the output layer. The number of nodes in the hidden layer of the encoder is set to 8, while the number of output layer nodes is set to $K = 3$ corresponding to the true number of modes. The decoder consists of 3 *linear* networks, each with a single layer having dimension equal to the dimension of the parameters of local affine function in (22). The VAE is trained maximizing the loss function (19) with $\lambda = 1 \cdot 10^{-3}$. The learning rate is set to $1 \cdot 10^{-3}$ and number of SGD iterations are fixed to $20 \cdot 10^3$. The required training time is 99.6 sec.

An independent noise-free test dataset of 2000 samples is generated to asses the performance of the PWA model. The R^2 scores obtained on the test data computed using *one-step-ahead* predicted output is R^2 $(1-\text{step}) = 99.43\%$ and using *simulated* output is R^2 (sim) $= 85.15\%$[2]. The obtained scores show that the true output is accurately reconstructed by the estimated model. The parameters of the local affine models (decoder weights and biases) are reported in Table 3, which closely match the true system parameters for all 3 modes.

[2]for computing R^2 (sim), *estimated* past outputs are considered to construct the regressor in order to predict the current output, while R^2 (1-step) is computed with *measured* past outputs in the regressor.

Table 3: True *vs* estimated model parameters (weights and biases of the decoder network).

Mode	True	Estimated
1	$[-0.4,\ 1,\ 1.5]$	$[-0.39,\ 0.99,\ 1.57]$
2	$[0.5,\ -1,\ -0.5]$	$[0.50,\ -1.00,\ -0.52]$
3	$[-0.3,\ 0.5,\ -1.7]$	$[-0.31,\ 0.50,\ -1.67]$

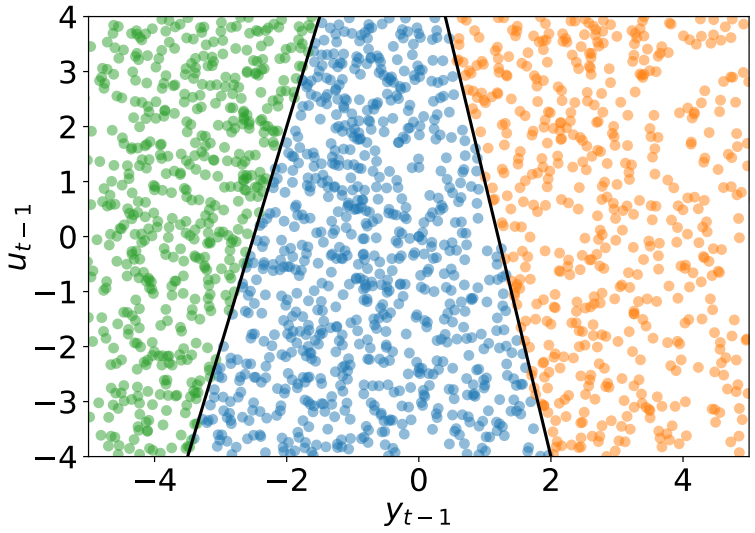

Figure 3: PWA-ARX model: True partition (solid black lines) *vs* estimated clustering of the regressor space.

Table 4: R^2 scores computed based on one-step-ahead prediction (1-step) and with simulated output (sim) for varying noise levels.

Test data	$\sigma = 0.1$ SNR = 30 dB	$\sigma = 0.3$ SNR = 20 dB	$\sigma = 0.7$ SNR = 13 dB
R^2 (1-step)	99.53 %	99.65 %	99.28 %
R^2 (sim)	97.19 %	96.02 %	82.43 %

The estimated clustering pattern of the regressor space is depicted in Fig. 3. It can be seen that the true underlying partition of the data-generating system (22) has been accurately recognized by the trained encoder network with each regressor being assigned to the correct operating mode. Finally, the ro-

bustness of the model against different noise conditions is tested. The R^2 scores computed on test data using one-step-ahead prediction and using simulation are reported in Table 4. The results show that the model is able to reconstruct the output fairly accurately even in the presence of high noise levels.

6.3 Example 3: Identification of NPWA-ARX model

In this example, we consider estimation of *nonlinearly* piecewise affine system. Specifically, we consider the following data-generating system as a slight modification to the PWA-ARX system (22), which can be conceived as an extension to the nonlinearly piecewise affine model [12, 5]:

$$y_t = \begin{cases} \begin{bmatrix} -0.4 & 1 & 1.5 \end{bmatrix} x_t + e_t, & \text{if } \begin{bmatrix} 4 & -1 & 10 \end{bmatrix} x_t < 0, \\ \begin{bmatrix} 0.5 & -1 & -0.5 \end{bmatrix} x_t + e_t, & \text{if } \begin{bmatrix} 4 & -1 & 10 \end{bmatrix} x_t \geq 0, \\ & \quad \& \begin{bmatrix} 5 & 1 & -6 \end{bmatrix} x_t \leq 0, \\ \begin{bmatrix} -0.4 & 1 & 1.5 \end{bmatrix} x_t + e_t, & \text{if } \begin{bmatrix} 5 & 1 & -6 \end{bmatrix} x_t > 0, \end{cases} \tag{23}$$

where $e \sim \mathcal{N}(0, \sigma^2)$ with $\sigma = 0.3$, which corresponds to the SNR = 19 dB. A training dataset of 6000 samples and a noise-free test dataset of 2000 samples is gathered.

Here, the system with $K = 2$ local affine models is assumed to introduce *nonlinear* partitioning requirements. We remark that the problem could be solved by considering a PWA-ARX model with $K = 3$ modes and *linearly separable* clusters. However, in this example, we aim to infer nonlinearly PWA-ARX (NPWA-ARX) model by considering only $K = 2$ modes with a nonlinear boundary between mode 1 and mode 2. Note that the system is defined by the same dynamics occurring over two regions of the regressor space, although, identification methods which rely on *linear* partition of the regression space would require three modes as the regions are not linearly separable.

For estimation with VAE, we consider encoder network having a single-hidden-layer with *relu* activations in the hidden layer and *softmax* activation at the output layer. The number of nodes in the hidden layer of the encoder is set to 10, while the number of output layer nodes is set to $K = 2$. The decoder consists of 2 *linear* networks, each with a single layer having dimension equal to the dimension of the parameters of local affine function in (23). The VAE is trained by maximizing the loss function (19) with $\lambda = 1 \cdot 10^{-3}$. The learning rate is set to $1 \cdot 10^{-3}$ and number of SGD iterations are fixed to $20 \cdot 10^3$. The required training time is 92.5 sec.

The estimated clustering pattern is shown in Fig. 4. It can be seen from the figure that despite nonlinear partitioning induced by the data-generating system, the encoder is able to recognize the underlying operating regions of the two dynamics very accurately. The parameters of the local affine models (decoder weights and biases) are reported in Table 3, which closely match the true system parameters for both modes. Finally, the R^2 score obtained on the test data

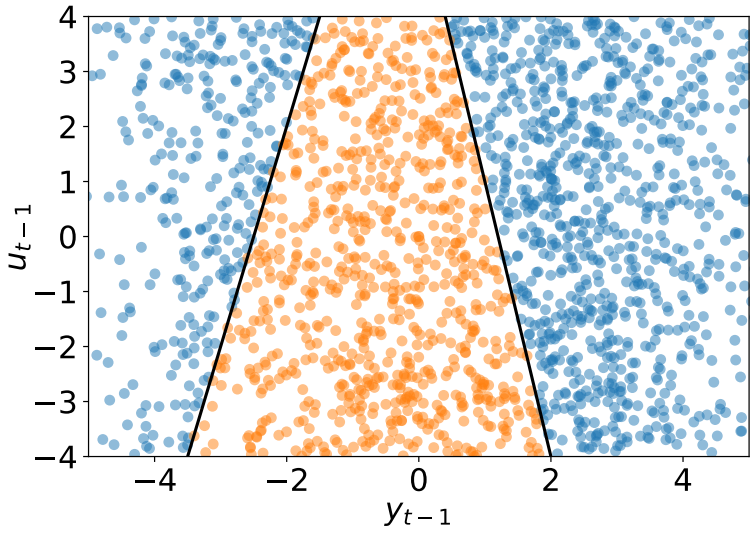

Figure 4: NPWA-ARX model: True partition (solid black lines) *vs* estimated clustering of the regressor space.

Table 5: True *vs* estimated model parameters (weights and biases of the decoder network).

Mode	True	Estimated
1	$[-0.4,\ 1,\ 1.5]$	$[-0.4032,\ 1.0006,\ 1.5034]$
2	$[0.5,\ -1,\ -0.5]$	$[0.5047,\ -1.0012,\ -0.5054]$

computed using *one-step-ahead* predicted output is $R^2\ (1-\text{step}) = 99.68\%$ and using *simulated* output $R^2\ (\text{sim}) = 96.77\%$, which shows the estimated model is able to reconstruct the output with high accuracy.

6.4 Example 4: Identification of PWNL model

In this example we consider the following PWNL data-generating function [16],

$$y_t = \begin{cases} -2 + 0.1\log(1-x_t) + e_t, & \text{if } x_t \leq 0, \\ -1 + 0.01x_t + \sin(x_t)/x_t + e_t, & \text{if } 0 < x_t \leq 10, \\ 2 + 0.2\exp\left(\frac{-1}{2}\frac{(x_t-15)^2}{5}\right)x_t + e_t, & \text{if } 10 < x_t \leq 20, \end{cases} \tag{24}$$

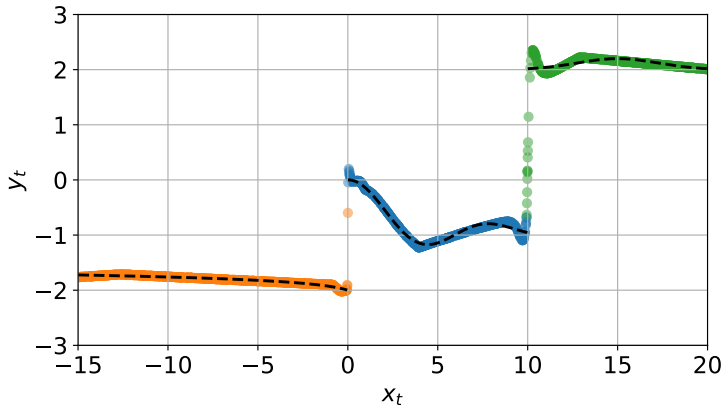

Figure 5: PWNL model: True function (dashed black lines) *vs* estimated submodels and clustering.

where $e \sim \mathcal{N}(0, \sigma^2)$ with $\sigma = 0.3$, which corresponds to the SNR = 15 dB. A training dataset of 6000 samples and a noise-free test dataset of 2000 samples is gathered from (24) with x_t uniformly distributed in $[-15, 20]$.

For inference, we consider fully-connected encoder network having a single-hidden-layer with *relu* activations in the hidden layer and *softmax* activation at the output layer. The number of nodes in the hidden layer of the encoder is set to 16, while the number of output layer nodes is set to $K = 3$. The decoder consists of 3 fully connected networks, each having a single hidden layer with number of nodes set to 8 and *relu* activation. The VAE is trained by maximizing the loss function (19) with $\lambda = 1 \cdot 10^{-2}$. The learning rate is set to $1 \cdot 10^{-3}$ and number of SGD iterations are fixed to $20 \cdot 10^3$. The required training time is 155.7 sec.

The estimated function along with the estimated clustering pattern is shown in Fig. 5. It can be seen from the figure that partition of the input data in the three regions is accurately recognized by the VAE. At the same time, the local non-linear functions have been approximated fairly accurately by the trained decoder networks. Finally, the R^2 scores obtained over the test dataset is $R^2(\text{test}) = 99.68\%$, which shows high prediction accuracy of the estimated model.

7 Conclusion and future works

We have presented a framework for learning piecewise models using specialized variational autoencoder. In contrast to the traditional black-box deep learning models, the developed VAE is interpretable, in the sense that the latent space can be interpreted in terms of the modes of the underlying hybrid system while the decoder represents local submodels. The developed approach is effective to

identify a general class of piecewise models as demonstrated in numerical case studies. Future works involve extension of the proposed method to PWA state-space models and investigating other variants of VAE, *e.g.*, vector quantized (VQ-VAE) for data-driven modeling and control of hybrid systems.

References

[1] A. Bemporad. A piecewise linear regression and classification algorithm with application to learning and model predictive control of hybrid systems. *IEEE Transactions on Automatic Control*, pages 1–16, 2022.

[2] A. Bemporad, A. Garulli, S. Paoletti, and A. Vicino. A bounded-error approach to piecewise affine system identification. *IEEE Transactions on Automatic Control*, 50(10):1567–1580, 2005.

[3] K. Boukharouba, L. Bako, and S. Lecoeuche. Identification of piecewise affine systems based on dempster-shafer theory. In *Proc. 15th IFAC Symposium on System Identification*, pages 1662–1667, Saint-Malo, France, 2009.

[4] V. Breschi, D. Piga, and A. Bemporad. Piecewise affine regression via recursive multiple least squares and multicategory discrimination. *Automatica*, 73:155–162, 2016.

[5] A. Brusaferri, M. Matteucci, and S. Spinelli. Identification of probability weighted ARX models with arbitrary domains. *arXiv:2009.13975*, 2020.

[6] M. Forgione and D. Piga. Model structures and fitting criteria for system identification with neural networks. In *Proc. of the 14th IEEE International Conference on Application of Information and Communication Technologies (AICT)*, pages 1–6, Tashkent, Uzbekistan, 2020.

[7] A. Garulli, S. Paoletti, and A. Vicino. A survey on switched and piecewise affine system identification. In *Proc. of the 16th IFAC Symposium on System Identification*, pages 344–355, Brussels, Belgium., 2012.

[8] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[9] A. L. Juloski, S. Weiland, and W. P. M. H. Heemels. A Bayesian approach to identification of hybrid systems. *IEEE Transactions on Automatic Control*, 50(10):1520–1533, 2005.

[10] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. *arXiv:1312.6114v10*, 2014.

[11] F. Lauer. On the complexity of piecewise affine system identification. *Automatica*, 62:148–153, 2015.

[12] F. Lauer and G. Bloch. Switched and piecewise nonlinear hybrid system identification. In *Hybrid Systems: Computation and Control*, pages 330–343, Berlin, Heidelberg, 2008. Springer.

[13] F. Lauer and G. Bloch. Piecewise smooth system identification in reproducing kernel hilbert space. In *53rd IEEE Conference on Decision and Control*, pages 6498–6503, 2014.

[14] L. Ljung, C. Andersson, K. Tiels, and T. B. Schön. Deep learning and system identification. In *Proc. of the 21st IFAC World Congress*, pages 1175–1181, Berlin, Germany, 2020.

[15] D. Masti and A. Bemporad. Learning nonlinear state–space models using autoencoders. *Automatica*, 129:109666, 2021.

[16] M. Mazzoleni, V. Breschi, and S. Formentin. Piecewise nonlinear regression with data augmentation. In *Proc. of the 19th IFAC Symposium on System Identification SYSID*, pages 421–426, Padova, Italy., 2021.

[17] M. Mejari, V. Breschi, and D. Piga. Recursive bias-correction method for identification of piecewise affine output-error models. *IEEE Control Systems Letters*, 4(4):970–975, 2020.

[18] M. Mejari, V. V. Naik, D. Piga, and A. Bemporad. Identification of hybrid and linear parameter-varying models via piecewise affine regression using mixed integer programming. *International Journal of Robust and Nonlinear Control*, 30(15):5802–5819, 2020.

[19] D. Piga, A. Bemporad, and A. Benavoli. Rao-Blackwellized sampling for batch and recursive Bayesian inference of Piecewise Affine models. *Automatica*, 117:109002, 2020.

[20] D. Piga, M. Forgione, and M. Mejari. Deep learning with transfer functions: new applications in system identification. In *Proc. of the 19th IFAC Symposium System Identification SYSID*, pages 415–420, Padova, Italy, 2021.

[21] J. Roll, A. Bemporad, and L. Ljung. Identification of piecewise affine systems via mixed-integer programming. *Automatica*, 40(1):37–50, 2004.

[22] S. Taguchi, T. Suzuki, S. Hayakawa, and S. Inagaki. Identification of probability weighted multiple ARX models and its application to behavior analysis. In *Proc. of the 48h IEEE Conf. on Decision and Control (CDC)*, pages 3952–3957, Shanghai, China, 2009.

[23] Y. Wang. A new concept using LSTM neural networks for dynamic system identification. In *Proc. of the 2017 American Control Conference (ACC)*, pages 5324–5329, Seattle, WA, USA, 2017.