Taylor & Francis
Taylor & Francis Group

Journal of Quality Technology

A Quarterly Journal of Methods, Applications and Related Topics

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/ujqt20

Toward a better monitoring statistic for profile monitoring via variational autoencoders

Nurettin Dorukhan Sergin & Hao Yan

JOURNAL OF QUALITY TECHNOLOGY
https://doi.org/10.1080/00224065.2021.1903821

Toward a better monitoring statistic for profile monitoring via variational autoencoders

Nurettin Dorukhan Sergin and Hao Yan

Arizona State University, Tempe, Arizona

ABSTRACT

Variational autoencoders have been recently proposed for the problem of process monitoring. While these works show impressive results over classical methods, the proposed monitoring statistics often ignore the inconsistencies in learned lower-dimensional representations and computational limitations in high-dimensional approximations. In this work, we first manifest these issues and then overcome them with a novel statistic formulation that increases out-of-control detection accuracy without compromising computational efficiency. We demonstrate our results on a simulation study with explicit control over latent variations, and a real-life example of image profiles obtained from a hot steel rolling process.

KEYWORDS

Deep learning; high-dimensional nonlinear profile; latent variable model; profile monitoring; variational autoencoder

1. Introduction

Profile monitoring has attracted a growing interest in the literature in the past decades for its ability to construct control charts with much better representations for certain types of process measurements (Maleki, Amiri, and Castagliola 2018; Woodall 2007; Woodall et al. 2004). A profile can be defined as a functional relationship between the response variables and explanatory variables or spatiotemporal coordinates. In this work, we focus on the case where the profiles generated from the process are high-dimensional, *that is*, the number of such explanatory variables or spatiotemporal coordinates are large. Specifically, we focus on the case where profiles are observed in a high-dimensional space, but profile-to-profile variation lies on a nonlinear low-dimensional manifold. Our motivating example of such high-dimensional profiles is presented in Figure 1, in which we exhibit a sample of surface defect image profiles collected from a hot steel rolling process.

In literature, profile monitoring techniques can be categorized by their assumptions on the type of functional relationship. For example, linear profile monitoring techniques assumed that the profiles can be represented by a linear function. The idea is to extract the slope and the intercept from each profile and monitor its coefficients (Zhu and Lin 2009). Regularization techniques can also be used in linear profile estimation. For example, Zou, Ning, and Tsung (2012) utilize a multivariate linear regression model for profiles with the LASSO penalty and use the regression coefficients for Phase-II monitoring. However, the linear assumption can be quite limiting. To address this challenge, nonlinear parametric models are normally proposed (Jensen and Birch 2009; Maleki, Amiri, and Castagliola 2018; Noorossana, Saghaei, and Amiri 2011; Williams, Woodall, and Birch 2007). These models assume an explicit family of parameterized functions and, their parameters are estimated via nonlinear regression. In both cases, the drawback of both linear and nonlinear parametric models is that they assume the parametric form is known beforehand, which might not always be the case in practice.

Another large body of profile monitoring research focuses on the type of profiles where the basis of the representation is assumed to be known, but the coefficients are unknown. For instance, to monitor smooth profiles, various nonparametric methods based on local kernel regression (Qiu, Zou, and Wang 2010; Zou, Tsung, and Wang 2008) and splines (Chang and Yadama 2010; Yan, Paynabar, and Shi 2018) are developed. To monitor the nonsmooth waveform signals, a wavelet-based mixed effect model is proposed (Paynabar and Jin 2011). However, for all the aforementioned methods, it is assumed that the nonlinear variation pattern of the profile is well captured by a known basis or kernel. Usually, there is no guidance on selecting the right basis of the representation for

CONTACT Hao Yan HaoYan@asu.edu Arizona State University, Tempe, AZ 85287.

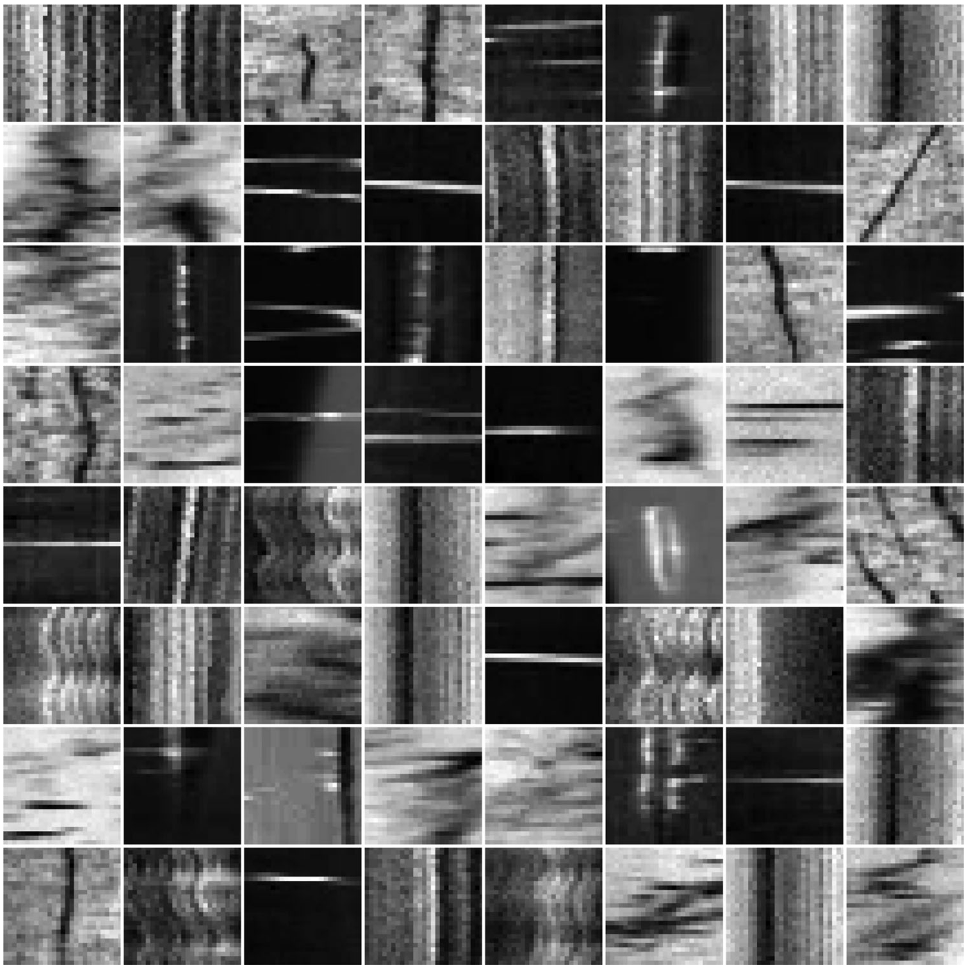


Figure 1. A collection of 64 by 64 image profiles taken from a hot steel rolling process.

the original data and it requires many trials and errors to find the right basis.

In the case that the basis of HD profiles is not known, dimensionality reduction techniques are widely used. Principal component analysis (PCA) is arguably the most popular method in this context for profile monitoring because of its simplicity, scalability, and good data compression capability. In Liu (1995), PCA is proposed to reduce the dimensionality of the streaming data where T^2 and Q charts are constructed to monitor the extracted representations and residuals, respectively. To generalize PCA methods to monitor the complex correlation among the channels of multichannel profiles, Paynabar, Zou, and Qiu (2016) propose a multivariate functional PCA method and apply change point detection methods on the function coefficients. Along this line, tensor-based PCA methods are also proposed for multichannel profiles, examples including uncorrelated multilinear PCA (Paynabar, Jin, and Pacella 2013) and multilinear PCA (Grasso, Colosimo, and Pacella 2014), and various tensor-based decomposition methods (Yan, Paynabar, and Shi 2015).

The main limitation of all the aforementioned PCA-related methods is that the expressive power of linear transformations is very limited. Furthermore, each principal component represents a global variation pattern of the original profiles, which is not efficient at capturing the local spatial correlation within a single profile. Therefore, PCA requires much larger latent-space dimensions than the dimension of the actual latent space, yielding a suboptimal and overfitting-prone representation. This phenomenon hinders profile monitoring performance.

A systematic discussion of this issue is articulated in Shi, Apley, and Runger (2016). In that work, the authors identify the problems associated with assuming a closeness relationship in the subspace that is characterized by Euclidean metrics. They successfully observe that the intra-sample variation in complex high-dimensional corpora may lie on a nonlinear manifold as opposed to a linear manifold, which is assumed by PCA and related methods. However, the authors only focus on applying manifold learning for Phase-I analysis, while the Phase-II monitoring procedure is not touched upon.

In recent years, we observe a surge in deep learning-based solutions to the problem. For instance, deep autoencoders have been proposed for profile monitoring for Phase-I analysis in Howard, Apley, and Runger (2018). In another work, Yan et al. (2016) compared the performance of contractive autoencoders and denoising autoencoders for Phase-II monitoring. Zhang et al. (2018) proposed a denoising autoencoder for process monitoring. Aside from deterministic deep neural networks, only three works (Lee et al. 2019; Wang et al. 2019; Zhang et al. 2019) proposed to use deep probabilistic latent variable models, specifically, variational autoencoders (VAE), for Phase-II monitoring. All the monitoring statistics in those works differ slightly, but they are all extensions of the classic T^2 and Q-charts of PCA. We argue that there is room for improvement for the monitoring statistic formulations in those works for several reasons, especially when high-dimensional profiles are considered. In this work, we propose a new monitoring statistic formulation to address this issue.

The contributions of this work are as follows:

- We compare the existing monitoring statistics proposed by previous works on VAE-based monitoring and unify them into the latent-space and residual-space monitoring statistics. We also prove the mathematical equivalency of these statistics with the classical T^2 and Q-charts of PCA in the linear setting.
- We highlight an important shortcoming of neural network-based encoders and how it negatively impacts the efficiency of statistics that are derived exclusively from learned latent representations. We demonstrate this on a carefully designed simulation study with explicit control over the actual latent variations.
- We explain why residual-space monitoring statistics can cover most types of process drifts in both conceptual illustration and real simulation study.
- We propose two approximations on the residual-space monitoring statistics leveraging on the first-order and second-order Taylor expansion that strikes a better balance between detection accuracy and computational feasibility than previously proposed similar statistics.
- We support our claims on both simulation and real-life case study profile datasets.

The rest of the article is organized as follows: Section 2 first introduces VAEs and reviews traditional T^2 and Q charts of PCA as well as the existing monitoring statistics proposed for VAE. Section 3 introduces our proposed monitoring statistic formulation and the rationale behind how it tackles the shortcomings of existing formulations. Section 4 introduces the simulation process used in this work as well as the manifestations of the aforementioned shortcomings. Finally, Section 5 demonstrates the advantages of the proposed methodology on a real-life case study, using images from a hot-steel rolling process.

2. Background

In this section, we review the VAE in Section 2.1. We will then review the T^2 and Q statistics for PCA methods in Section 2.2. Finally, we will briefly review the existing works profile monitoring works utilizing the VAE in Section 2.3.

2.1. Variational autoencoders

We will first review the VAE, which was first introduced by Kingma and Welling (2014). VAE soon became one of the most prominent probabilistic models in the literature. The Gaussian factorized latent variable model perspective of VAEs is crucial to understand the role of this model in the context of profile monitoring. This is why we begin with an introduction to latent variable modeling.

Let us assume we observe samples $\boldsymbol{x} \in \mathbb{R}^d$ in a high-dimensional space, generated by a multivariate random process that can be described by the density function $p(\boldsymbol{x})$. We also believe that there is redundancy in this observation and sample-to-sample variation can be explained well by a latent representation $\boldsymbol{z} \in \mathbb{R}^r$, where the latent dimension $r \ll d$. Latent variable models are powerful tools to model such complex distributions. The joint density $p(\boldsymbol{x}, \boldsymbol{z})$ is factorized into the distribution of the latent variables $p(\boldsymbol{z})$ and the conditional distribution of observed variables given latent variables $p(\boldsymbol{x} \mid \boldsymbol{z})$. A typical example of latent variable models is when the joint distribution is Gaussian factorized as in Eq. [1].

$$\begin{aligned} p(\boldsymbol{z}) &= \mathcal{N}(\boldsymbol{z}; 0, \boldsymbol{I}_r) \\ p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z}) &= \mathcal{N}(\boldsymbol{x}; \mu_{\boldsymbol{\theta}}(\boldsymbol{z}), \sigma^2 \boldsymbol{I}_d)\,. \\ p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) &= p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z}) p(\boldsymbol{z}) \end{aligned} \quad [1]$$

In the above formulation, the function $\mu_{\boldsymbol{\theta}} : \mathbb{R}^r \to \mathbb{R}^d$ is a function parameterized by $\boldsymbol{\theta}$, which describes the relationship between the latent variables and the mean of the conditional distribution. The Gaussian prior $p(\boldsymbol{z})$ is typically chosen to be standard multivariate Gaussian distribution to avoid degenerate solutions (Roweis and Ghahramani 1999) and conditional covariance is typically assumed to be isotropic $\sigma^2 I_d$ to avoid ill-defined problems. The aim is to approximate the true density $p_{\boldsymbol{\theta}}(\boldsymbol{x}) \approx p(\boldsymbol{x})$ and this approximation can be obtained through marginalization:

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

A famous member of the family of models described above is the probabilistic PCA (PPCA) (Tipping and Bishop 1999). The parameters are optimized via a maximum likelihood estimation framework and it can be solved analytically since the function $\mu_{\boldsymbol{\theta}}$ is a simple linear transformation. This enables reusing analytical results from solutions to the classical PCA problem. The assumption of PPCA that the latent and observed variables have a strictly linear relationship is restrictive. In real-world processes, this relationship is likely highly nonlinear. Deep latent variable models are a marriage of deep neural networks and latent variable models that aim to solve this problem. Deep learning has enjoyed a tremendous resurgence in the last decade due to their superior performance that was unprecedented for many tasks such as image classification (Krizhevsky, Sutskever, and Hinton 2012), machine translation (Bahdanau, Cho, and Bengio 2014), and speech recognition (Amodei et al. 2016). In theory, under sufficient conditions, a two-layer multilayer perceptron can approximate any function on a bounded region (Cybenko 1989; Hornik 1991). However, growing the width of shallow networks exponentially for arbitrarily complex tasks is not practical. It has been shown that deeper representations can often achieve better expressive power than shallow networks with fewer parameters due to the efficient reuse of the previous layers (Eldan and Shamir 2016).

VAE is arguably the most foundational member of the deep latent variable model family. The main difference between PPCA and VAE is that VAE replaces the linear transformation with a high-capacity deep neural network (called *generative* or *decoder*). This is powerful in the sense that, along with a general-purpose prior $p(\boldsymbol{z})$, deep neural networks can transform such prior to model a wide variety of densities to model the training data (Kingma and Welling 2019). Unlike PPCA, these models will not have analytical solutions due to the complex nature of the neural network used. Like most other deep learning models, their parameters are often optimized via variants of stochastic gradient descent optimizers. The problem becomes even harder given that the posterior $p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z})$ takes meaningful values only for a small subregion within the latent space $\mathbb{R}^r$. This makes sampling from the prior $p(\boldsymbol{z})$ to estimate the likelihood prohibitively expensive. Both models work around this problem using the importance sampling framework (Bishop 2006, 532), where they introduce another network (called *recognition* or *encoder*) to approximate a proposal distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})$ – parameterized by $\boldsymbol{\phi}$ – which aims to sample latent variables from a much smaller region that is more likely to produce higher posterior densities for a given input $\boldsymbol{x}$. The encoder is modeled as another Gaussian distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}; \mu_{\boldsymbol{\phi}}(\boldsymbol{x}), \sigma_{\boldsymbol{\phi}}(\boldsymbol{x}))$ where the mean and standard deviation of the proposal distribution are inferred via high capacity neural networks $\mu_{\boldsymbol{\phi}}$ and $\sigma_{\boldsymbol{\phi}}$, respectively.

One important output of a trained VAE is the likelihood estimator. Once the two networks are trained, the log-likelihood $\log p_{\boldsymbol{\theta}}(\boldsymbol{x})$ can be approximated by a Monte Carlo sampling procedure with L iterations (Kingma and Welling 2019, 30):

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) \approx \log \frac{1}{L} \sum_{l=1}^{L} \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}^{(l)})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}^{(l)} | \boldsymbol{x})}. \qquad [2]$$

However, the Monte Carlo sampling procedure is shown to be computationally inefficient and the evidence lower bound (ELBO), which is deemed a proxy to the likelihood, is often used as the objective to be optimized.

$$\begin{aligned} \text{ELBO} &\triangleq \log\big(p(\boldsymbol{x})\big) - \text{KL}\big(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) \mid\mid q^*(\boldsymbol{z}|\boldsymbol{x})\big) \\ &= \mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\theta}}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z}) + \text{KL}\big(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) \mid\mid p(\boldsymbol{z})\big), \end{aligned} \qquad [3]$$

In the equation above, $\text{KL}(\cdot \mid\mid \cdot)$ denotes the Kullback–Leibler divergence (KLD) between two distributions. The left-hand side is the quantity of interest, while the right-hand side is the tractable expression that guides the updating of parameters $\boldsymbol{\theta}, \boldsymbol{\phi}$ in an end-to-end fashion.

2.2. Review of T^2 and Q statistics in PCA

We will then review the profile monitoring statistics in the PCA. Profile monitoring via PCA is typically done using the T^2 and Q statistics (Chen et al. 2004). The Q statistic for PCA is defined as the reconstruction error between the observed profile $\boldsymbol{x}$ and the reconstructed profile $\tilde{\boldsymbol{x}}$. The geometric interpretation of Q statistics is that it quantifies how far the sample is away from the learned subspace of in-control samples. T^2 statistics on the other hand, quantifies the shift along the directions of the most dominant principal components.

The T^2 statistic and Q statistic for PCA are defined formally as follows:

$$\begin{aligned} Q(\boldsymbol{x}) &= \| \ \boldsymbol{x} - \tilde{\boldsymbol{x}} \ \|^{2} \\ T^2(\boldsymbol{x}) &= \boldsymbol{z}^\top \boldsymbol{\Sigma}_r^{-1} \boldsymbol{z} = \boldsymbol{x}^\top \boldsymbol{W}_r \boldsymbol{\Sigma}_r^{-1} \boldsymbol{W}_r^\top \boldsymbol{x} \end{aligned} \qquad [4]$$

where matrix $\boldsymbol{W}_r$ is the loading matrix, and $\boldsymbol{\Sigma}_r^{-1}$ is the inverse of the covariance matrix when only the first r principal components are kept. There are various methods to choose r such as fixing the percentage of variation explained (Chiang, Russell, and Braatz 2001, 41).

For processes with relatively small latent and residual dimensionality, the upper control limits of these statistics for the α percent Type-1 error tolerance is constructed by employing the normality assumptions of PPCA (Chiang, Russell, and Braatz 2001, 43–44). However, using such measures for high-dimensional nonlinear profiles is prohibitively error-prone as both r and d will be much higher than the assumptions of chi-square distribution can tolerate. As an alternative, nonparametric methods are typically used to estimate these limits, such as simple percentiles or kernel density estimators.

2.3. Review of previously proposed monitoring statistics proposed for VAE

In this section, we will briefly review several proposed monitoring statistics for VAEs. Three works have recently considered VAE for process monitoring, all of which propose different statistic formulations for monitoring. Zhang et al. (2019) propose H^2, which is basically the Mahalanobis distance of the mean of the proposal distribution from standard Gaussian distribution.

$$H^2 = \mu_\phi(\boldsymbol{x})^\top \mu_\phi(\boldsymbol{x}). \qquad [5]$$

In another work, Lee et al. (2019) propose two statistics: T^2 and SPE. For a given input $\boldsymbol{x}$, a single sample is drawn from the proposal distribution $\boldsymbol{z}^{(l)} \sim q_\phi(\boldsymbol{z} \mid \boldsymbol{x})$ which is used to reconstruct the input using the generative model $\boldsymbol{x}^{(l)} \sim p_\theta(\boldsymbol{x} \mid \boldsymbol{z}^{(l)})$. The proposed test statistics in this work can be formalized as follows:

$$\begin{aligned} T^2 &= (\boldsymbol{z}^{(l)} - \bar{\boldsymbol{z}})^\top S_{\boldsymbol{z}}^{-1} (\boldsymbol{z}^{(l)} - \bar{\boldsymbol{z}}) \\ SPE &= \| \ \boldsymbol{x}^{(l)} - \boldsymbol{x} \ \|_2^2 \end{aligned} \qquad [6]$$

where $\bar{\boldsymbol{z}}$ and $S_{\boldsymbol{z}}^{-1}$ are estimated over a single pass of the entire set of in-control samples. In their methodology, these two statistics work in combination and at least one positive decision from either of the two statistics is enough to claim that the process is out-of-control.

Finally, Wang et al. (2019) propose the R and D statistics by focusing on the two major components of the tractable part of the objective function of VAE shown as in Eq. [3]. The D statistic is simply the KL divergence between the prior and proposal. For R statistic, like Lee et al. (2019), they employ summary statistics over samples from proposal but also claim that sampling size can be fixed to one:

$$\begin{aligned} D &= \mathrm{KL}(q_\phi(\boldsymbol{z} \mid \boldsymbol{x}) \ || \ p(\boldsymbol{z})) \\ R &= \frac{1}{L} \sum_{l=1}^{L} - \log q_\theta(\boldsymbol{x} \mid \boldsymbol{z}^{(l)}) \end{aligned} . \qquad [7]$$

SPE in and R are essentially the same quantities up to a constant, which makes them identical in the context of monitoring. This is why we will refer to them as SPE/R throughout the rest of the article.

3. Methodology

In this section, we start by explaining how previously proposed statistics for VAE-based monitoring are modeled as extensions of their PCA-based monitoring counterparts, in Section 3.1. Then, we will reveal the pitfalls of this extension concerning the behaviors of neural networks in Section 3.2. Against the backdrop of these pitfalls, we will propose a novel monitoring statistic formulation. Lastly, we will outline the implementation details of profile monitoring procedures and neural network architectures we use in this study in Sections 3.3 and 3.4, respectively.

3.1. Relationship of the monitoring statistics for VAE and PCA

A common approach in the literature to tackle process monitoring with VAE is to extend the definitions of T^2 and Q statistics of the PCA-based monitoring to VAE. This is done by breaking the tractable portion of Eq. [3] into two terms as follows:

$$\begin{aligned} Q_{VAE} = \mathbb{E}_{\boldsymbol{z} \sim q_\theta} \log p_\theta(\boldsymbol{x} \mid \boldsymbol{z}), T^2_{VAE} \\ = \mathrm{KL}(q_\phi(\boldsymbol{z} \mid \boldsymbol{x}) \ || \ p(\boldsymbol{z})). \end{aligned} \qquad [8]$$

Either these formulations or some variant of them are typically used as the monitoring statistics. To understand the rationale behind this, we will revisit the assumptions of the model described in Eq. [1]. Let us formally represent an out-of-control distribution as a shift in $p(\boldsymbol{x})$. Since $p(\boldsymbol{x}) = \int p(\boldsymbol{x} \mid \boldsymbol{z}) p(\boldsymbol{z}) d\boldsymbol{z}$, we can anticipate two sources: a shift in the latent distribution $p(\boldsymbol{z})$ or a shift in the residual distribution $p(\boldsymbol{x} \mid \boldsymbol{z})$. The two statistics are assumed to be connected to these two sources: (1) a shift in the conditional

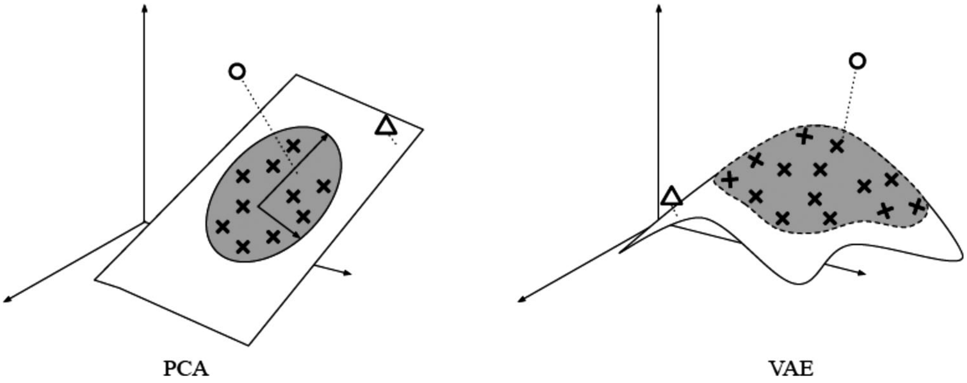


Figure 2. Illustration of the analogy between PCA and VAE. Closed regions describe the lower-dimensional manifold the in-control distribution lies in. The crosses represent the in-control samples observed in Phase-I and the gray region represents the subset of the lower-dimensional manifold where in-control samples are typically sampled from. The observation represented with a circle is typically detected with Q-statistic and the observation represented with a triangle is typically detected with T^2-statistic.

distribution $p(\boldsymbol{x} \mid \boldsymbol{z})$ can be detected monitoring $Q_{VAE} = \mathbb{E}_{z \sim q_\theta} \log p_\theta(\boldsymbol{x} \mid \boldsymbol{z})$ and (2) a shift in the latent distribution $p(\boldsymbol{z})$, can be detected monitoring $T^2_{VAE} = \mathrm{KL}(q_\phi(\boldsymbol{z} \mid \boldsymbol{x}) \mid\mid p(\boldsymbol{z}))$.

This idea is similar to utilizing both T^2 and Q-charts in the PCA-based method, where both terms play an important role in process monitoring (Kim and Lee 2003). To make this similarity more obvious, we prove that if the same ELBO framework for VAE used above is used for PPCA (see Section 2.2), we prove the equivalency of T^2 and Q statistics of PPCA and T^2_{VAE} and $Q_{V\ AE}$ of VAE in the linear settings.

Proposition 3.1. *If we use linear decoders for VAE, the models will become the Probabilistic PCA (*Tipping and Bishop 1999*) that the prior and decoding functions are normally distributed as:*

$$\begin{aligned} p(\boldsymbol{z}) &= \mathcal{N}(0, \boldsymbol{I}), \\ p_\theta(\boldsymbol{x} \mid \boldsymbol{z}) &= \mathcal{N}(\boldsymbol{W}\boldsymbol{z}, \sigma^2 \boldsymbol{I}). \end{aligned}$$

In this case, the encoder can be solved analytically as another normal distribution as $q_\phi(\boldsymbol{z} \mid \boldsymbol{x}) = \mathcal{N}(\mu_\phi(\boldsymbol{x}), \Sigma_z)$, where $\mu_\phi(\boldsymbol{x}) = \boldsymbol{M}^{-1}\boldsymbol{W}^\top \boldsymbol{x}, \Sigma_z = \sigma^2 \boldsymbol{M}^{-1}$, and $\boldsymbol{M} = \boldsymbol{W}^\top \boldsymbol{W} + \sigma^2 \boldsymbol{I}$. Then, the two monitoring statistics defined in Eq. [8] can be derived as follows:

$$\mathrm{KL}\big(q_\phi(\boldsymbol{z} \mid \boldsymbol{x}) \mid\mid p(\boldsymbol{z})\big) = \frac{1}{2} \quad \|\quad \mu_\phi(\boldsymbol{x}) \quad \|\quad ^2 + C_1 \tag{9}$$

$$\mathbb{E}_{z \sim q_\phi} \log p_\theta(\boldsymbol{x} \mid \boldsymbol{z}) \propto \quad \|\quad \boldsymbol{x} - \boldsymbol{W}\mu_\phi(\boldsymbol{x}) \quad \|\quad ^2 + C_2 \tag{10}$$

where C_1 and C_2 are constants that do not depend on $\boldsymbol{x}$. The proof is given in Appendix A.

Note that the constants do not affect the profile monitoring decision as the control limits will be translated accordingly. Thus, the test statistic T^2_{VAE} and Q_{VAE} for linear decoders (i.e., PPCA) is equivalent to the T^2-statistic and Q-statistic of PCA, respectively, residual-space.

Observe that previously proposed formulations mentioned in Section 2.3 draw inspiration – directly or indirectly – from this framework. Statistics R and SPE are based on the Q-statistic. Let us call these *residual-space statistics*, as they rely on the sum of squared differences between the signal itself and its predicted value, *that is*, residuals. The statistics H^2, T^2, and D are based on the T^2 of PCA. We call these *latent-space statistics*, as they rely exclusively on latent representations.

Figure 2 shows a graphical illustration of this analogy of residual-space statistics and latent-space statistics for PCA and VAE. Residual-space statistics quantify the distance of the observed data with respect to the learned linear or nonlinear manifold. The latent-space statistics monitor the distance within the learned manifold. In the linear case (i.e., PCA), this is the Euclidean distance. However, in the nonlinear case (i.e., VAE), this distance should be defined on the nonlinear manifold.

3.2. Proposed monitoring statistic

In this section, we will first reveal the shortcomings of the previously proposed VAE-based monitoring methodologies we presented in Section 2.3. This will lead us to the rationale behind the design of our proposed statistic, which is also included in this section after the explanation of the shortcomings.

There are two major pitfalls of the previously proposed methodologies:

1. Latent-space statistics H^2, T^2, and D or any other formulation that relies exclusively on the latent

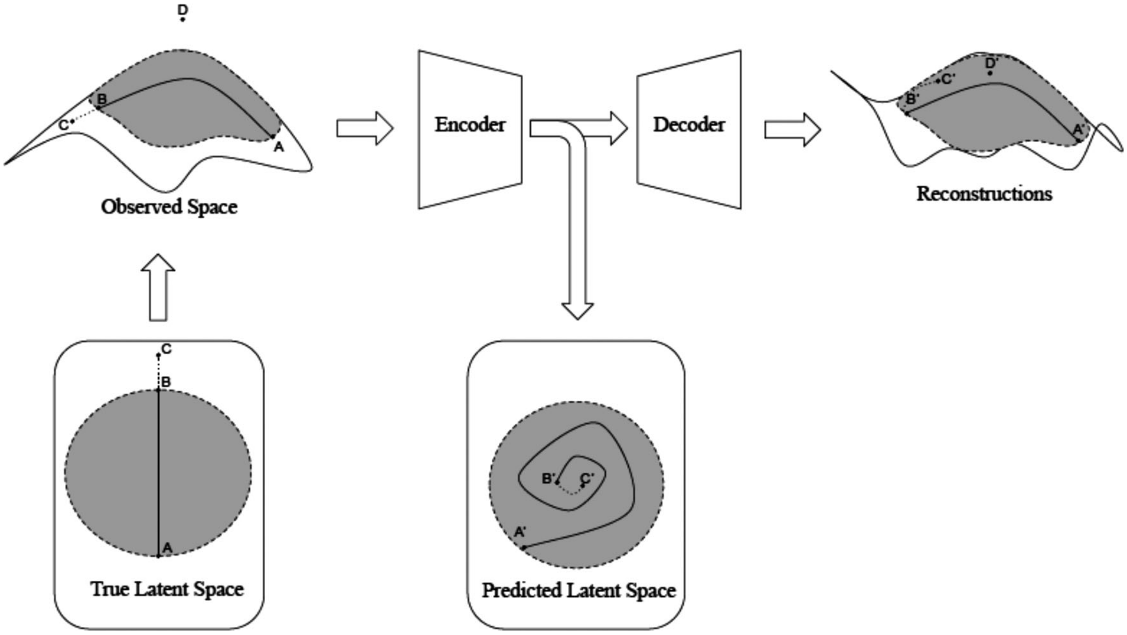


Figure 3. Illustration of incorrect latent representation phenomena and how process control fails in latent space. **Bottom left:** The true latent variations of in-control samples are generated from the gray region, which is the probable region. Point A and Point B are extreme values along a dimension of variation. Point C is generated by an out-of-control process with a shift in latent distribution. Point D is generated by an out-of-control process with a shift in the residual distribution. The predicted counterpart of each point is denoted by an apostrophe (*e.g.*, A′ for A). **Top Left:** Observations of true latent variation in the high-dimensional space that lie close to a low-dimensional manifold. **Top Middle:** The encoder and decoder of VAE trained exclusively with in-control samples (*i.e.*, the gray region in the observed space). **Bottom Middle:** Incorrectly mapped variation in the predicted latent space where the gray region is the probable region. **Top Right:** Reconstructions of the variation in high-dimensions, with a failure in extrapolation beyond the in-control region.

representation $z \sim q_\phi(z|x)$ will be unreliable for process monitoring. Thus, they should be discarded altogether from the monitoring framework since they will likely increase false alarms without contributing to the detection power in any meaningful way.

2. Residual-space statistic *SPE* and *R* rely on Monte Carlo sampling. These are not computationally feasible given how expensive calculations are on deep neural networks. An alternative approach is required to stay computationally feasible without sacrificing too much from the estimation quality of these statistics.

We will address these two shortcomings in Sections 3.2.1 and 3.2.2.

3.2.1. Unreliability of latent-space statistics for deep autoencoders

First, we focus on the unreliability of latent-space statistics. Let us first start with the case when the shift occurs in the latent distribution (i.e., $p(z)$). According to the PCA–VAE analogy illustrated in Figure 2, latent-space statistics are supposed to catch such shifts, which are represented with triangular points in the same figure. While this may work for PCA-based monitoring, we claim that such an analogy cannot be straightforwardly made for VAE. Here, we will explain the two major reasons why the latent-space statistics failed to capture the change in the latent space: "incorrect" latent representation by the encoder and failure to extrapolate by the decoder.

The first reason that latent-space statistics should not be used is that neural network-based encoders in autoencoder architectures typically learn "incorrect" latent representation. We illustrate this phenomenon in Figure 3. The line segment ABC illustrates a traversal along a latent dimension. All the samples generated along the line segment AB are sampled from the typical region of the in-control process and their latent representations are contained within the typical region of the predicted space. However, Point C is generated by an out-of-control process where there is a shift in the latent distribution but its mapping incorrectly falls within the probable region. This leads to false evidence which suggests that Point C is unlikely to be generated by an out-of-control process while in reality, it was.

The reasons as to why incorrect latent representation is learned by deep autoencoders have been studied well in the deep learning literature. Interested readers are encouraged to refer to Achille and Soatto (2018) for a discussion of the properties of ideal latent representations and to Locatello et al. (2019) for a discussion of the challenges around attaining one of these properties, namely, disentanglement. The key takeaway is that it is very likely that we end up with an imperfect mapping, especially with real-life datasets. Consequently, in Phase-II, samples generated by out-of-control processes that are characterized by a shift in the latent distribution will not be mapped consistently to the regions in the latent space, which we consider to be unusual. This will result in an increased type-II error.

A natural question to ask at this point is how we should expect to detect shifts in latent distribution if we cannot rely on latent representations. We argue that the residual-space statistics (i.e., an analog of a Q-chart) would catch such shifts too, even though its original purpose is to catch shifts in the residual space. Our argument is based on another "shortcoming" of neural networks, namely, failure to extrapolate. Deep neural networks approximate well only at a bounded domain defined by where the training set is densely samples from. In the context of our problem, this refers to the high-density region of $p(\boldsymbol{x})$, which generated the set of in-control profiles we use in Phase-I. The behavior of the function is unpredictable and often erroneous outside the training domain. In other words, it does not extrapolate well beyond the domain of training samples, which are likely to be coming from out-of-control processes. We refer interested readers to Appendix B, where we replicate this phenomenon on a toy example.

This leads to the second reason why the residual-space statistics should be used only: failure to extrapolate by the decoder. A decoder that fails to extrapolate is counter-intuitively helpful for the residual-space statistics since it will struggle with generating profiles that are in the low-density region of the in-control data distribution $p(\boldsymbol{x})$. This means that the discrepancy between the true profile and its generated counterpart will be larger for out-of-control profiles than it is for in-control profiles, regardless of the source of the shift. Overall, we conclude that the residual-space monitoring statistic would be efficient at detecting changes in the residual space and latent space. We refer the readers to Figure 3 for an illustration. Point C is generated from a shift in the latent space distribution. However, due to the "incorrect" mapping of the latent distribution, Point C' will still lie in the in-control region of the latent space. There is a significant discrepancy between Point C and reconstruction C', which can be detected by the residual-space statistics. Point D is generated from a shift in the residual space and can be captured by the residual space statistics. In conclusion, the residual-space statistic should be able to catch changes in both the residual space and latent space.

3.2.2. Improving the computational efficiency of the residual-space statistics

Now that we established our rationale behind the first shortcoming we claim to reveal, we move onto the second and focus on the previously proposed residual-based statistics: SPE and R. Both SPE and R rely on samples from the proposal distribution for the estimation of the expectation. This approach requires a large number of samples to be generated, and thus a large number of the forward passes through the decoder network, which is prohibitively expensive in terms of computation when deployed in real-life systems. To overcome this problem, we propose a Taylor expansion based approximation. First, observe that $\log p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z}) \propto \| x - \boldsymbol{\mu}_{\boldsymbol{\theta}}(z) \|_2^2 + C$ for all $\boldsymbol{x}$ and $\boldsymbol{z}$ because of the common isotropic covariance assumption. The constant C can be discarded as noneffective in terms of control charting because it would only translate the limits and the statistics by the same amount for any given $\boldsymbol{x}$ and $\boldsymbol{z}$. We call the expression $\mathbb{E}_{z \sim q_\phi} \| x - \boldsymbol{\mu}_{\boldsymbol{\theta}}(z) \|_2^2$ as the expected reconstruction error (ERE). The Taylor expansion for the first-order and second-order moment of ERE given the random variable $\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})$ can be derived analytically.

Proposition 3.2. Assume that a VAE is trained with in-control samples. The training results in the mean and diagonal covariance estimators of the proposal distribution as well as the mean estimator of the condition distribution which are denoted by $\boldsymbol{\mu}_{\boldsymbol{\phi}}, \boldsymbol{\sigma}_{\boldsymbol{\phi}}$, and $\boldsymbol{\mu}_{\boldsymbol{\theta}}$, respectively. The first and second-order Taylor Expansion (denoted by ERE_1 and ERE_2 respectively) for the function $\mathbb{E}_{z \sim q_\phi} \| \boldsymbol{x} - \boldsymbol{\mu}_{\boldsymbol{\theta}}(z) \|_2^2$ given the random variable $\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) = \mathcal{N}(\mu_{\boldsymbol{\phi}}(\boldsymbol{x}), \boldsymbol{\sigma}_{\boldsymbol{\phi}}(\boldsymbol{z}))$ and where the conditional $p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z}) = \mathcal{N}(\mu_{\boldsymbol{\phi}}(\boldsymbol{x}), \mathrm{diag}(\boldsymbol{\sigma}_{\boldsymbol{\phi}}(x)))$ can be derived analytically as

$$ERE_1 = \| \boldsymbol{x} - \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x})) \|_2^2 \qquad [11]$$

$$ERE_2 = \| \boldsymbol{x} - \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x})) \|_2^2 + \frac{1}{2}\mathrm{tr}(\mathbf{H}_z \mathrm{diag}(\boldsymbol{\sigma}_{\boldsymbol{\phi}}(x))) \qquad [12]$$

where $\mathbf{H}_z$ *is the Hessian of the function* $\| x - \boldsymbol{\mu}_{\boldsymbol{\theta}}(z) \|_2^2$ *with respect to* $\boldsymbol{z}$. *The derivation is provided in Appendix C.*

Table 1. Architecture details of deep neural networks used in this study.

Module	Architecture
Encoder	C(32, 4, 2, 1) - A() - C(32, 4, 2, 1) - A() - C(64, 4, 2, 1) - A() - C(64, 4, 2, 1) - A() - C(64, 4, 1, 0) - FC(256, 2r)
Decoder	FC(r, 256) - A() - CT(64, 4, 0, 0) - A() - CT(64, 4, 2, 1) - A() - C(32, 4, 2, 1) - CT(32, 4, 2, 1) - A() - CT(1, 4, 2, 1)

Given a trained VAE, ERE_1 can be computed efficiently by a single forward pass of the new profile from the pass $\boldsymbol{x}$ through $\boldsymbol{\mu}_\phi$ and $\boldsymbol{\mu}_\theta$ successively and calculating the squared prediction error, without the need for any sampling. ERE_2 requires the additional computation of the diagonal of the Hessian $\mathbf{H}_z$ and a relatively less expensive trace operation since the covariance is diagonal. Both ERE_1 and ERE_2 are residual-based statistics that are accurate and efficient to compute, which addresses the two shortcomings we mentioned at the beginning of this section. In our experiments, we will evaluate the effectiveness of both of these statistics in comparison to previously proposed monitoring statistics for VAE.

3.3. Profile monitoring procedure

A typical profile monitoring follows two phases: Phase-I analysis and Phase-II analysis. Phase-I analysis focuses on understanding the process variability by training an appropriate in-control mode and selecting an appropriate control limit. In our case, Phase-I analysis results in a trained model (i.e., an encoder and a decoder) and an Upper Control Limit (UCL) to help set up the control chart for each of the monitoring statistics. In Phase-II, the system is exposed to new profiles generated by the process in real-time to decide whether these profiles are in-control or out-of-control. Our experimentation plan, outlined below, is formulated to emulate this scenario to effectively assess the performance of any combination of a model, a test statistic, and a disturbance scenario to generate the out-of-control samples.

- Obtain in-control dataset $\mathcal{D}$ and partition it into train, validation and test sets $\mathcal{D}^{trn}, \mathcal{D}^{val}, \mathcal{D}^{tst}$.
- Train VAE using samples from $\mathcal{D}^{trn}$.
- Calculate test statistic for all $\boldsymbol{x} \in \mathcal{D}^{val}$ and take its 95th percentile as the UCL.
- Start admitting profiles online from the process. Calculate test statistic using the trained VAE. If the test statistic is over UCL, identify the sample as out-of-control.

We train 10 different model instances with different seeds to account for inherent randomness due to the weight initialization of deep neural networks.

3.4. Neural network architectures and training

In this work, we use convolutional neural networks for the encoders and decoders in our VAE model to represent the spatial neighborhood structures of the profiles. Introduced in LeCun et al. (1989), convolutional layers have enabled tremendous performance increase in certain neural network applications where the data are of a certain spatial neighborhood structure such as images or audio waveform. They exploit an important observation of such data, where the learner should be equivariant to translations. This is an important injection of inductive bias into the network that largely reduces the number of parameters compared to the fully connected network by the use of parameter sharing. It eventually increases the statistical learning efficiency, especially for small samples. It must be noted, however, convolutional layers are not equivariant to scale and rotation as they are to translation. Knowing what sort of inductive biases is injected into these layers is important for the understanding of disentanglement, which we will introduce later in this article.

We use the encoder–decoder structure outlined in Table 1. The layers used that build the model architectures used in this study are summarized as follows:

- C(O, K, S, P): Convolutional layer with arguments referring to the number of output channels O, kernel size K, stride S and size of zero-padding P.
- CT(O, K, S, P): Convolutional transpose layer with arguments referring to the number of output channels O, kernel size K, stride S, and size of zero-padding P.
- FC$(I,\ O)$: Fully connected layer with arguments referring to input dimension I and output dimension O.
- A(): Activation function. Leaky ReLU with a negative slope of 0.2.

Here, C(), CT(), and FC() are considered the linear transformation layers while R(), LR(), and S() are considered the nonlinear activation layers. Strided convolutions can be used to decrease the spatial dimensions in the encoders. Pooling layers are typically not recommended in autoencoder-like architectures (Radford, Metz, and Chintala 2016). Convolutional transpose layers are used to upscale latent codes back to ambient dimensions.

The sequential order of the computational graphs used for this study is summarized in Table 1. The architecture choice is directly based on the encoder-decoder architecture that was used in Higgins et al. (2017), except that we use Leaky ReLU with a negative slope of 0.2 as the activation, which is advised in Radford, Metz, and Chintala (2016) for better gradient flow. The encoder outputs $2r$ nodes, which is a concatenation of the inferred posterior mean $\boldsymbol{\mu}_{\phi}(\boldsymbol{x})$ and variance $\mathrm{diag}(\boldsymbol{\sigma}(\boldsymbol{x}))$, both are of length r. The number of epochs per training is fixed at 1000, and the learning rate and batch size are fixed at 0.001 and 64, respectively, both are chosen empirically to guarantee a meaningful convergence. Adam algorithm is used for first-order gradient optimization with parameters $(\beta_{1,}\beta_2) = (0.9, 0.999)$ as advised in Kingma and Ba (2015). The model checkpoint is saved at every epoch where a better validation loss is observed. The latest checkpoint is used as the final model.

In our experiments, the architecture and the training conditions described above are optimized with respect to the convergence performance of the VAE objective on the in-control dataset. This is because in real life, the practitioner will not have access to out-of-control samples. Consequently, the same setting worked well for both the simulation dataset and the case study dataset we considered in this article. This gives us confidence that the selection is robust from one set to the other. However, a different dataset might benefit from adjustments to the above conditions. The adjustments should be based on monitoring the convergence of the VAE objective, as the procedure will benefit from a better approximated in-control distribution.

We would like to emphasize that even we focus only on the image profiles in our article by the convolutional architectures, which will be introduced to the readers in the upcoming simulation and case study sections, the monitoring statistics we propose in Eqs. [11] and [12] can be applied to other profiles as well, which will be left as the future work.

4. Simulation study analysis and results

In this section, we will evaluate the proposed methodology via a simulation study. We will first test our claims we make in Section 3.2 in a controlled environment over the data generating process as described in Section 4.1. For every experiment mentioned in this section, we follow the procedure outlined in Section 3.3 and we use VAE models with the architecture described in Section 3.4.

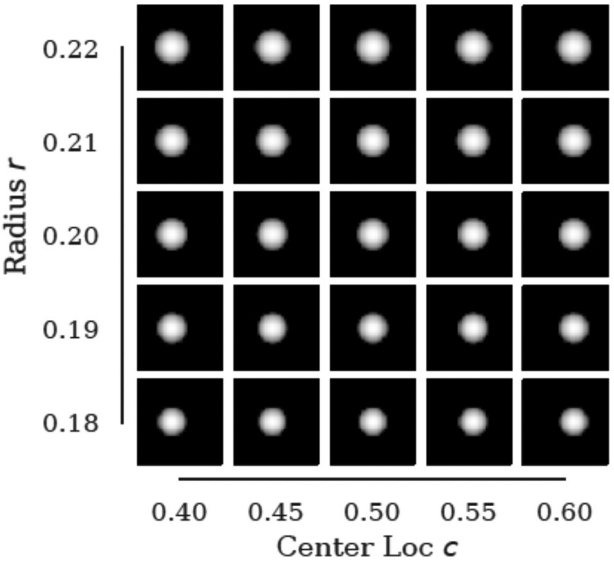


Figure 4. Dome profiles depicted as grayscale images simulated with radius and center location they coincide with on the axes.

We will then illustrate the incorrect mapping of the latent space and the extrapolation issue in Sections 4.2 and 4.3 under this controlled experiment.

4.1. Simulation setup

We first evaluate the performance of the deep latent variable models in a simulation setting where we have explicit control over the latent variations. The simulation procedure produces 2 D structured point clouds that resemble the scanned topology of a dome.

Let each pixel on a 64 by 64 grid be denoted by a tuple $\boldsymbol{p} = (p_0, p_1)$. The values of the tuples stretch from 0 to 1, equally spaced, left to right and bottom-up. Each tuple takes a value based on its location through a function $\boldsymbol{p} \mapsto f(\boldsymbol{p}; c, r) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 1 \times 10^{-2})$ is i.i.d Gaussian noise. The function f is parameterized by the horizontal location of the dome c, and the radius of the base of the dome r. The vertical location of the dome on the 2 D surface is fixed at the vertical center of the surface. Given any parameter set $\{c, r\}$, each pixel $\boldsymbol{p}$ can be evaluated with the following logic:

$$\begin{aligned} g(\boldsymbol{p}; c, r) &= 1 - \frac{(p_0 - c)}{r^2} - \frac{(p_1 - 0.5)}{r^2} \\ f(\boldsymbol{p}; c, r) &= \begin{cases} \sqrt{g(\boldsymbol{p}; c, r)} & \text{if } g(\boldsymbol{p}; c, r) \geq 0 \\ 0 & \text{if } g(\boldsymbol{p}; c, r) < 0 \end{cases}. \end{aligned} \quad [13]$$

The samples are best visualized as grayscale images, as shown in Figure 4.

The processes that generate the latent variations of in-control domes are defined as Gaussian distributions:

$$\begin{array}{l} c \sim \mathcal{N}(0.5, 1 \times 10^{-2}) \\ r \sim \mathcal{N}(0.2, 6.25 \times 10^{-4}) \end{array}. \qquad [14]$$

As our out-of-control scenarios consider the following four distribution shifts in which δ denotes the intensity of the shift:

- **Location shift:** the mean of the process that generates c is altered by an amount δ as in

$$c \sim \mathcal{N}(0.5 + \delta \times 10^{-2}, 1 \times 10^{-2})$$

- **Radius shift:** the mean of the process that generates a is perturbed by an amount δ as in

$$r \sim \mathcal{N}(0.2 + \delta \times 10^{-4}, 6.25 \times 10^{-4})$$

- **Mean shift**: all the pixels are added to an additive disturbance δ as in

$$f(\boldsymbol{p}; c, r) \leftarrow f(\boldsymbol{p}; c, r) + \delta$$

- **Magnitude shift:** all the pixels are added to a multiplicative disturbance δ as in

$$f(\boldsymbol{p}; c, r) \leftarrow f(\boldsymbol{p}; c, r) * \delta$$

Note that the location shift and radius shift represent disturbances in latent distribution $p_\delta(\boldsymbol{z})$. The other two cases, mean shift and magnitude shift, represent disturbances in the conditional distribution $p_\delta(\boldsymbol{x} \mid \boldsymbol{z})$.

We generate the training, validation, and testing sets for in-control domes as well as a set of each out-of-control scenario above. All sets have exactly 500 distinct samples. We generate these sets once, fix them, and use them for the analyses in the subsequent sections.

4.2. On the incorrect mapping of latent representations by the encoder

In this section, we will investigate the latent representations produced by the encoder and whether it can be mapped back to the "true" latent space that generates the data in the context of our simulation study.

We first train a VAE with an architecture described in Table 1 and fix the generating latent representation as $r = 2$. The training samples are generated by the in-control dome generation process as described in Section 4.1. We will use the encoder of the trained VAE for the rest of the analysis.

We can generate samples from the trained encoder by fixing one of the true latent factors and traversing along the other. The plots on the left side of Figure 5 depict the traversals of the true latent space we sample the domes from. We then push these generated examples through the encoder to obtain their respective proposal distributions. We will compare the mean of the respective proposal distributions and the true latent space. If the learned proposal distribution is mapped into a substantially different geometry by the encoders, we will describe the distribution as "incorrect".

Figure 5 shows the incorrectness in the mapping of latent representations. This incorrect mapping behavior is even worse when we are dealing with the extreme values in the true latent space. For example, from Figure 5b, we can conclude that domes with extremely small radii will likely go undetected if only the latent-space statistic is used.

Overall, the learned latent representations are typically "incorrect" especially for the samples with extreme latent variables. This, in turn, will lead to an incorrect out-of-control assignment in Phase-II analysis, if only the latent-space monitoring statistic is used.

4.3. On the extrapolation performance of the decoder

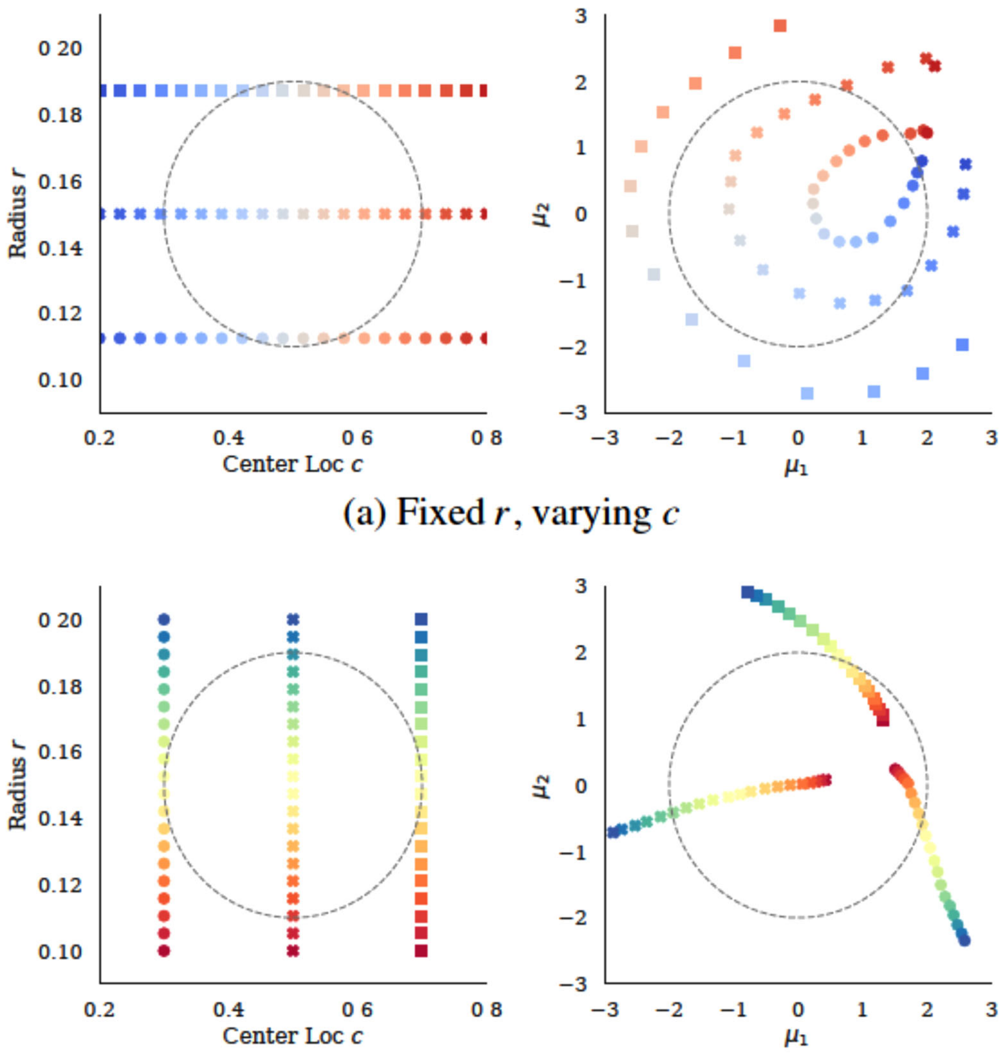
In this subsection, we will evaluate the extrapolation performance of the decoder. To demonstrate this, we showed the generated images by the decoder in Figure 6, when traveling along one axis of the latent dimension while keeping the other fixed.

Here, the decoder is trained on in-control samples described in Section 4.1, which is the same VAE described in Section 4.2

It should be cross-examined with Figure 5 above as the encoder and decoder are tightly coupled to each other. We observe two important behaviors: the posterior gets distorted beyond two or three standard deviations, and the representations are partially entangled in line with the behavior of its encoder depicted in Figure 5.

To see how this will help to detect disturbances in the latent space, we consider a dome that is extremely small in terms of the radius (i.e., small r) or at the very margins of the grid in terms of center location (i.e., center location c far from 0.5). Looking at Figure 6, we can observe that the decoder simply cannot generate such a sample because it does not extrapolate well in either of the latent dimensions. This will, in turn, produce a larger reconstruction error and can be captured by the residual-space monitoring statistic.

Recall once again that the disturbance described is purely on the latent distribution $p(\boldsymbol{z})$ and yet detected by the residual-space monitoring statistic only due to the extrapolation issue.



(a) Fixed r, varying c

(b) Fixed c, varying r

Figure 5. Figure depicting the discrepancy between the true and predicted latent representations of the encoder of a VAE with two-dimensional latent code trained with in-control samples. For each subfigure, plots on the left show where real factors of variation are sampled from and the figure on the right is what the VAE encoder infers as the mean of the proposal distribution. In all figures, the regions that are considered to be in-control are represented with a dashed circle. **Top:** Real factors of variation are generated at three fixed levels of radius r and varying values of center location c on the left figure. Corresponding inferred means are plotted on the right graph. **Bottom:** Similar to (b) but the center location c fixed at three levels and varying r.(a) Fixed r, varying c.(b) Fixed c, varying r.

4.4. On the estimation of log-likelihood under importance sampling

Earlier, we claimed that it would take too many Monte Carlo iterations to get a meaningful estimate of ERE defined as $\mathbb{E}_{z \sim q_\theta} \log p_\theta(\boldsymbol{x} \mid \boldsymbol{z})$. In this section, we test that claim on a random in-control sample $\boldsymbol{x}$ using the proposal distribution $\boldsymbol{z} \sim q_\phi(\boldsymbol{z} \mid \boldsymbol{x})$, which is obtained via the encoder of the same VAE model we have been using in this section. The results of the sampling-based estimation of ERE, first-order approximation ERE_1, and second-order approximation ERE_2 are shown in Figure 7. The key observation is that it takes at least 60 Monte Carlo iterations to get a stable and accurate estimation. At that level, the single pass through the encoder is negligible. This means using sampling will be more costly at least 60 samples to achieve the same accuracy as the first-order approximation that we suggest and at least 80 samples to get the accuracy of the second-order approximation. Another important observation is that the second-order approximation is a bit more accurate than first-order approximation since it is closer to the sample-average approximation, but their difference is quite insignificant. Furthermore, it requires much

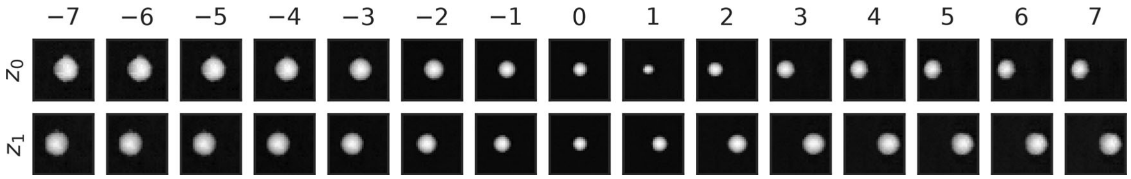


Figure 6. Latent-space traversal and the response of the decoder of a VAE with two-dimensional latent codes and trained with in-control dome samples. Each row represents which latent dimension is traversed while the other dimension is fixed at zero. Each column represents what value is assigned to that latent dimension that is represented by the row label. Each image in each cell is generated by the decoder using that specific latent variable combination.

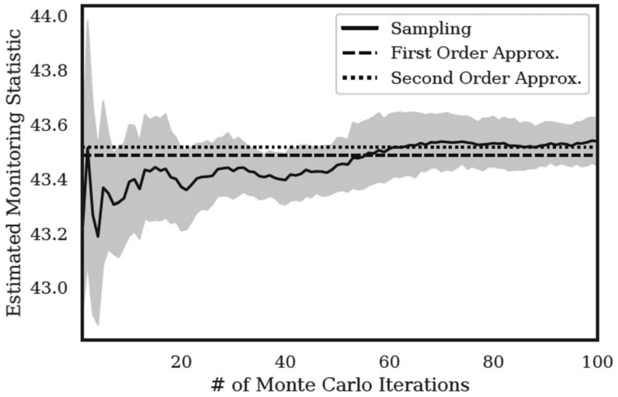


Figure 7. Estimation comparison between Monte Carlo sampling, first-order approximation and second-order approximation. A 95 percent confidence interval band is shown in the gray band and is based on simulations with 10 different seeds.

more computation for the second-order approximation, given the second-order Hessian matrix needs to be evaluated. In the next subsection, we will evaluate the performance of ERE_2 and ERE_1 in Phase-II monitoring to evaluate whether the added computational complexity for ERE_2 is justifiable.

4.5. Comparison of detection performance of proposed statistics

We now compare the proposed statistics based on the Phase II monitoring performance by how accurately they detect profiles from out-of-control processes outlined in Section 4.1. Note that for all statistics that require sampling, we obtain a single sample and calculate the statistic based on that to keep the computational demand the same for all statistics and emulate the computational constraints of a real-life case. A preliminary result we must check is the robustness of the statistics by making sure all proposed statistics have false alarm rates on the held-out in-control test set, which should also be less than the desired rate 5 percent. Table 2 demonstrates that this is the case for all of them.

Table 2. False alarm rates on the held-out dataset averaged over 10 replications per model and monitoring statistic.

Statistic	ERE_1	SPE/R	D	H^2	T^2
	0.041(0.006)	0.051(0.005)	0.044(0.004)	0.052(0.005)	0.043(0.009)

Standard deviations are in parentheses.

Through Figure 8, we observe a clear superiority of ERE_1 and ERE_2 over other methods when the disturbance is on the observable space (top row). Latent-space statistics D, H^2, and T^2 fail in this case since that they are purely computed using the proposal distribution latent variables. ERE_1 and ERE_2 also outperform SPE/R, although by a smaller margin it has with the latent variable-statistics. Between ERE_1 and ERE_2, it is hard to claim which one works better since their mean performances are quite close to each other.

For the latter two disturbances occurring purely on latent dimensions, results are presented in the bottom row of Figure 8. The key observations can be listed as follows:

- Generally ERE_1 and ERE_2, D and H^2 tend to perform better than SPE/R and T^2. A commonality between the former three is that they do not rely on random samples, supporting our argument against this practice.
- Observe the radius shift-type disturbance show in the bottom left figure. Even though H^2 performs better on positive intensities (larger radii), it completely misses negative intensities (smaller radii). We foresaw this result in Section 4.2. To reiterate, the “incorrect” mapping of the latent space and the lack of extrapolation in the encoder is the reason behind this. We would also suggest that this result can extend to all the latent-variable based statistics for deep autoencoder-based methods.
- Unlike latent-space statistics, ERE_1 and ERE_2 and SPE/R behave more robustly against varying intensities. In other words, the detection rate increases with increased intensities consistently. Among these, we observe that ERE_1 and ERE_2 consistently outperform SPE/R.
- ERE_1 and ERE_2 perform very similarly. In this case, we conclude that the second-order information does not help too much for Phase-II

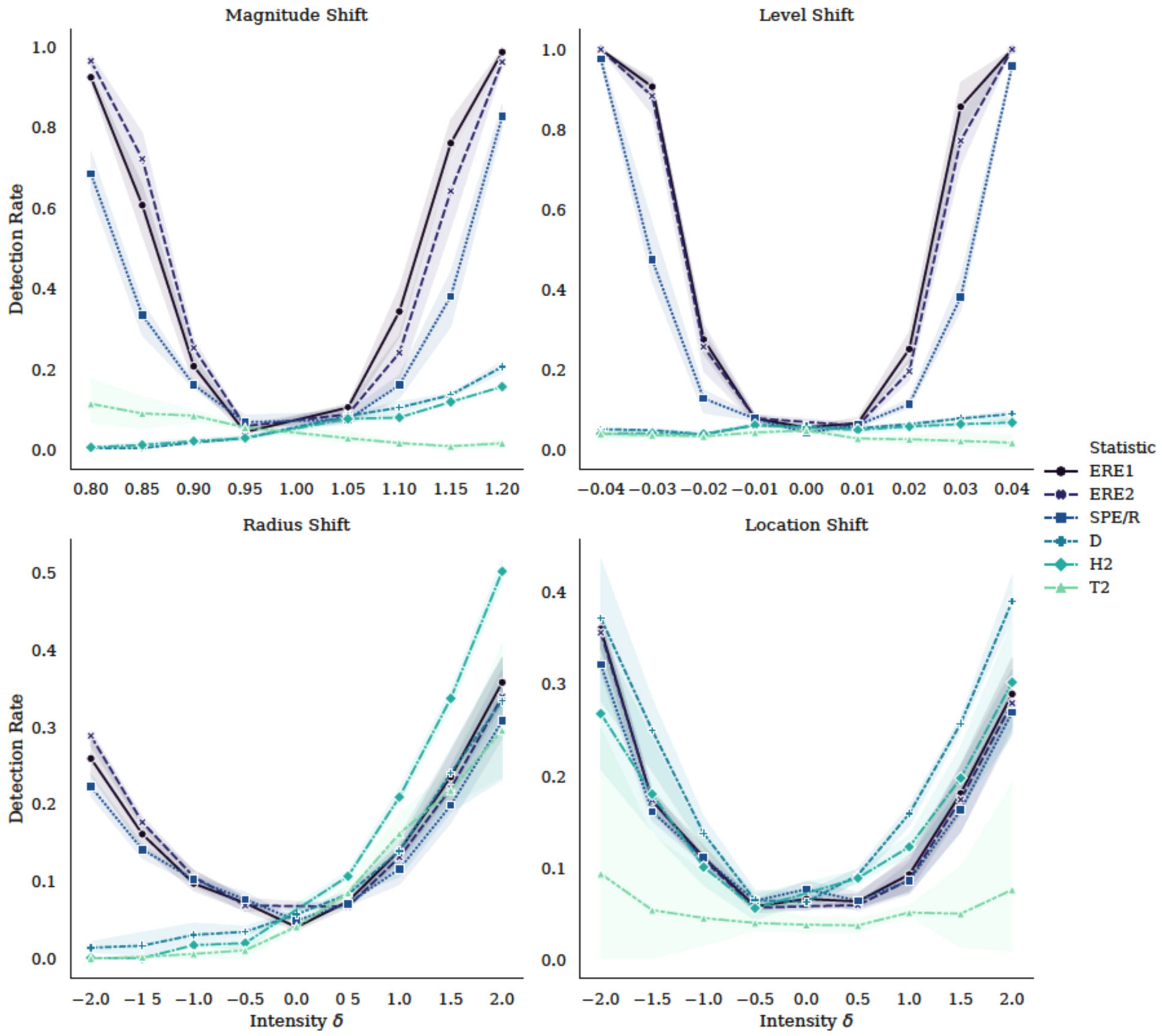


Figure 8. Fault detection rates (*y*-axis) for varying intensities (*x*-axis) of different disturbance types (quadrants). Bands represent a 95 percent confidence interval estimated around mean detection rates.

monitoring. The reason behind this is that the second-order information also comes from the encoder. However, given that the encoders are trained on in-control samples and may provide inaccurate information in the out-of-control regions, the second-order information for out-of-control samples would be biased. Therefore, it does not provide additional gain for monitoring performance.

As mentioned, in a real-life process, disturbances on the residual space is often more likely than the disturbance in the latent space. Therefore, we would recommend the use of residual-space monitoring statistics. Among all residual-space monitoring statistics, we conclude that ERE_1 perform the best, considering the accuracy, robustness, and computational demand. This will be further validated through the case study analysis.

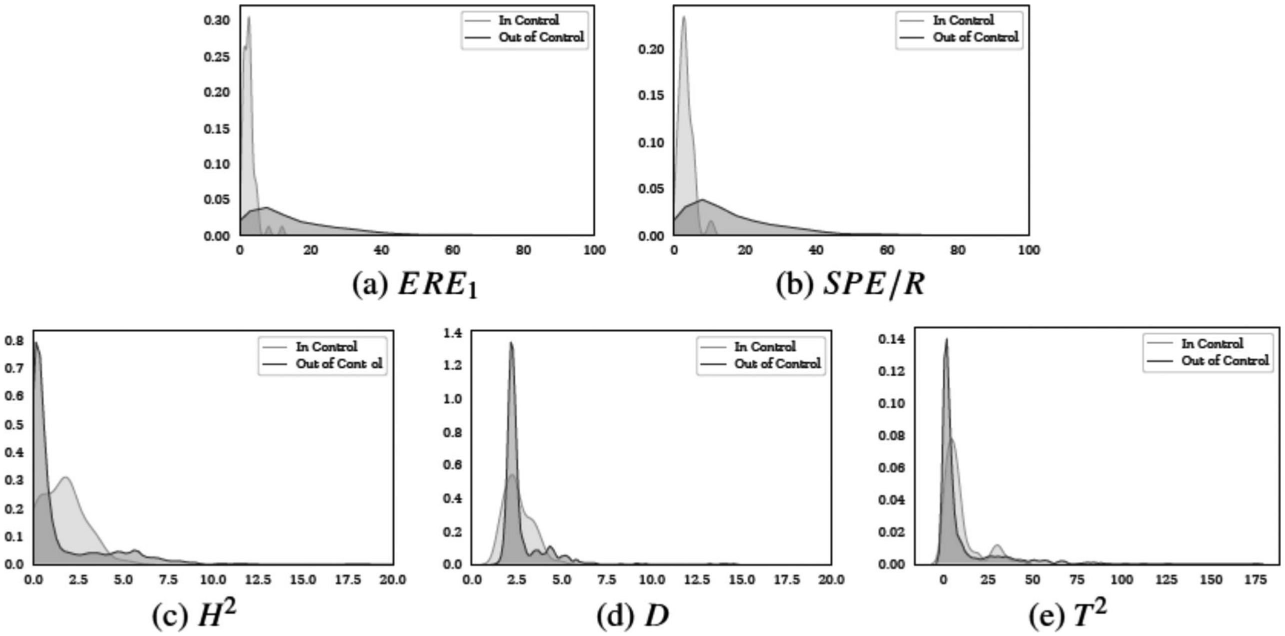
5. Case study analysis and results

In this section, we will evaluate the performance of the proposed algorithm using a real case study. Our dataset consists of defect image profiles from a hot-steel rolling process, which is shown in Figure 1. There are 13 classes of surface defect types identified by the domain engineers. Four of these classes – 0, 1, 9, and 11 – are considered minor defects and they constitute our in-control set. There are 338 images in these classes. The other nine classes make up the out-of-control cases and they have in combination 3351 images to report detection accuracy for. We randomly partition the in-control corpus to fix train, validation, and test sets with 60 percent–20 percent–20 percent relative sizes, respectively. The rest of the procedure followed is outlined in Section 3.3. Same as in the simulation study, to account for randomness in weight

Table 3. Summary of fault detection rates on out-of-control cases averaged over 10 replications per model and monitoring statistic.

Model	VAE						PCA
Statistic	D	H^2	T^2	SPE/R	ERE	ERE2	Q
Fault ID							
2	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.37(0.03)	0.44(0.06)	**0.50**(0.06)	0.00(0.00)
3	0.17(0.06)	0.23(0.04)	0.03(0.03)	0.84(0.01)	0.85(0.01)	**0.86**(0.01)	0.78(0.00)
4	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.62(0.02)	**0.75**(0.05)	0.71(0.05)	0.56(0.00)
5	0.58(0.07)	0.62(0.09)	0.00(0.00)	**1.00**(0.00)	**1.00**(0.00)	**1.00**(0.00)	0.99(0.00)
6	0.06(0.03)	0.15(0.08)	0.05(0.05)	0.79(0.01)	**0.80**(0.01)	**0.80**(0.00)	0.52(0.00)
7	0.01(0.01)	0.01(0.01)	0.00(0.00)	0.13(0.01)	**0.17**(0.01)	0.15(0.00)	0.11(0.00)
8	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.64(0.02)	**0.70**(0.07)	0.69(0.01)	0.34(0.00)
10	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.49(0.03)	**0.57**(0.05)	**0.57**(0.04)	0.29(0.00)
12	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.79(0.01)	**0.80**(0.02)	**0.80**(0.02)	0.69(0.00)
13	0.00(0.00)	0.00(0.00)	0.01(0.00)	0.71(0.04)	**0.77**(0.02)	0.76(0.02)	0.56(0.00)

Standard deviations are in parentheses. Bolded values represent the maximum average across different statistics.



(a) ERE_1

(b) SPE/R

(c) H^2

(d) D

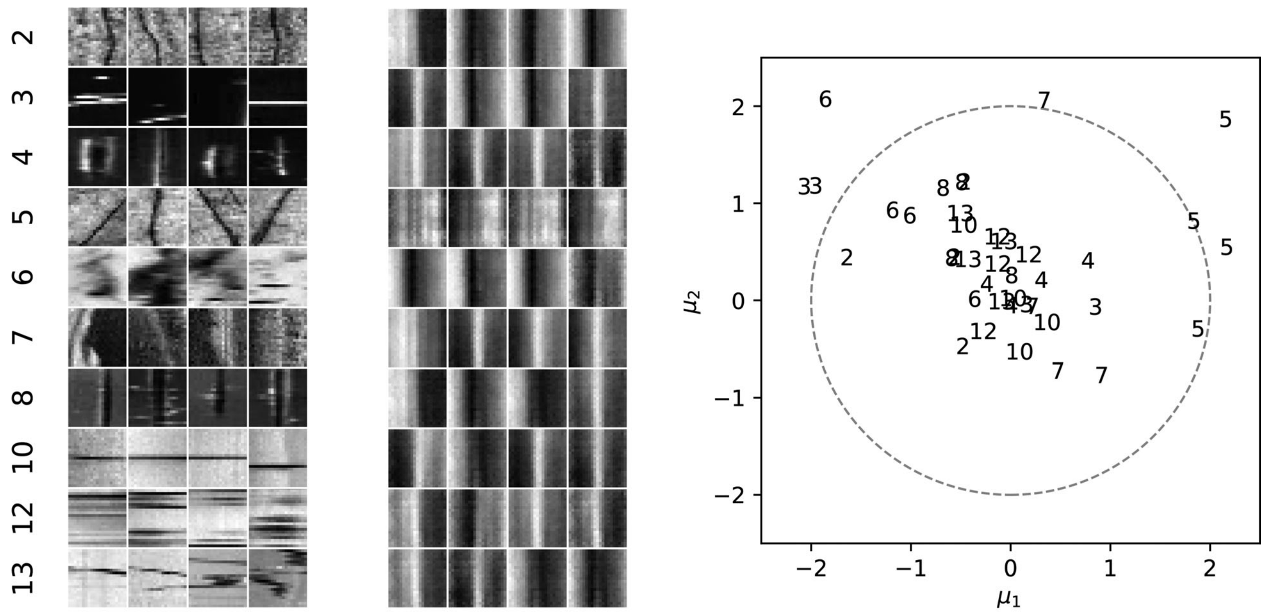
(e) T^2

Figure 9. Kernel density estimation plots of statistics obtained for in-control and out-of-control steel defect profiles, per each proposed statistic type.(a) ERE_1, (b) SPE / R, (c) H^2, (d) D, (e) T^2.

initialization, we replicate the experiment with 10 different seeds. For comparison, we also include the monitoring performance with the traditional PCA method with the same residual-space control chart, denoted as PCA-Q. The results are summarized in Table 3.

From Table 3, we can observe that ERE_1 and ERE_2 consistently outperform all other monitoring statistic formulations. The divide between residual-space statistics and latent-space statistics observed in the simulation study is further validated here too. The inferiority of latent-space statistics is much more obvious here in the real case study, as we observe for most out-of-control classes, the detection rate is simply zero. This observation further validates our claims that in practice, for deep autoencoders, the change happens in the residual space rather than the latent space. The advantage of VAE over PCA is mainly due to the better representative power and data compression ability of deep autoencoders compared to PCA. It is worth noting that the superiority of VAE over PCA for process monitoring was also demonstrated in the earlier works in various applications (Lee et al. 2019; Wang et al. 2019; Zhang et al. 2019).

To support our claim of the ineffectiveness of latent-space statistics, we refer the reader to Figure 9. We observe how well separated the statistics are for ERE_1 and SPE/R while for latent-space statistics, the obtained values are mostly overlapping. Note that we omitted ERE_2 because it was almost identical to ERE_1. To obtain a deeper understanding of the results, we point out in Figure 10 for the original images and their reconstructions. The decoder is persistent on generating samples that look like in-control rolling



(a) Original profiles (b) Reconstructions via VAE (c) Inferred means

Figure 10. Output of the VAE decoder and the encoder for randomly select rolling profiles. **Left:** Original profiles visualized. Each row is a class of defect profile and each column is randomly selected from that class. **Middle:** Reconstructions of the samples with one-to-one correspondence to the samples on the image to the left. **Right:** Inferred mean locations of each of the defects visualized on the left. Points are annotated by their class IDs.
(a) Original profiles. (b) Reconstructions via VAE. (c) Inferred means.

samples with little fidelity to how the original defect sample looks like. When Figures 10a and b are cross-examined, it is apparent why reconstruction error would be high. On the contrary, Figure 10c shows that most latent representations fall into the region that would be considered in-control from a profile monitoring perspective. We observed instances of classes 3, 5, 6, and 7 generate the latent variables in the out-of-control regions. However, even for these classes, SPE/R, ERE_1, and ERE_2 yield much better detection power than D, H^2, and T^2, as it can be seen in Table 3. In conclusion, we would like to suggest the use of ERE_1 for deep autoencoders, which is consistent with our findings in the simulation study.

Finally, we report execution time details for our proposed statistic, ERE_1. For this study, we utilized a workstation with 6-core Intel(R) Core(TM) i7-5930K CPU 3.50 GHz CPUs and 4 GeForce GTX 1080 Ti GPUs. Neural network computations are executed on a single GPU and a single CPU core is used for image input/output and preprocessing steps such as resizing to 64-by-64 and grayscale conversion wherever needed. A single GPU has 12GB memory and the model parameters take up about 730MBs. GPUs can leverage parallel computation of multiple images, therefore the remaining memory can be used to stock up images so their execution becomes parallel. An example of a batch of 128 images takes up only 63MBs more space in the GPU's memory and the per image execution time is roughly 0.8 ms. On the extreme case of using a single image per batch, per image execution time is around 2 ms on average, which satisfies the real-time monitoring constraint.

6. Conclusion

In this article, we focused on evaluating Phase-II monitoring statistics proposed so far in the literature for VAE and demonstrate that they were not performing optimally in terms of accuracy and/or computational feasibility. First, we classified these statistics into two groups and showed how they are designed as an extension to the classical statistics used for PCA. Then we pointed out that such an extension is not as straightforward as it seems due to the incorrectness of learned latent representations by VAEs and also due to the failure to extrapolate behavior. This led us to the conclusion that only residual-space statistics should be monitoring, regardless of the anticipated source of the shift in the process. We also pointed out that the residual-space statistics based on sampling will require too many samples to be computationally feasible. Finally, we proposed a novel formulation by deriving the Taylor expansion of the expected reconstruction error that addresses the computational efficiency issue in residual-space statistics.

We put our claims to the test with a carefully designed simulation study. This study demonstrated the discrepancy between the true latent variations and its learned counterparts, and its implications to the process monitoring performance of latent-space statistics. We also reinforced our claim that the derived statistics based on the residual space is overall more robust and accurate than all the other statistics proposed so far. Finally, we validated the superiority of our formulation on a real-life case study, where steel defect image profiles are used.

For future work, we hope to extend the proposed method for other types of data format. For example, for sequential profiles (e.g., time series), one-dimensional convolutional layers or a recurrent neural network for encoder and decoder structures as outlined in Chung et al. (2015) can be used. We are also curious to see how new developments in deep learning research will affect profile monitoring in high dimensions in the future. Specifically, developments in deep latent variable models and representation learning may have important implications.

Funding information

This work is partially supported by NSF DMS 1830363 and NSF CMMI 1922739. The support is gratefully acknowledged.

About the authors

Nurretin Dorukhan Sergin is a doctoral candidate at the Industrial Engineer program at Arizona State University. His current research is focused on out-of-distribution behaviors of deep neural networks and spatiotemporal modeling of urban mobility. During his master's, he did research on agent-based modeling and its application to computational social simulation problems.

Hao Yan received his BS degree in Physics from the Peking University, Beijing, China, in 2011. He also received a MS degree in Statistics, a MS degree in Computational Science and Engineering, and a PhD degree in Industrial Engineering from Georgia Institute of Technology, Atlanta, in 2015, 2016, 2017, respectively. Currently, he is an Assistant Professor in the School of Computing, Informatics, and Decision Systems Engineering at ASU. His research interests focus on developing scalable statistical learning algorithms for large-scale high-dimensional data with complex heterogeneous structures to extract useful information for the purpose of system performance assessment, anomaly detection, intelligent sampling and decision making. Dr. Yan was also the recipient of multiple awards including best paper award in IEEE TASE, IISE Transaction and ASQ Brumbaugh Award. Dr. Yan is a member of IEEE, INFORMS and IIE.

ORCID

Nurettin Dorukhan Sergin http://orcid.org/0000-0001-8522-7551
Hao Yan http://orcid.org/0000-0002-4322-7323

References

Achille, A., and S. Soatto. 2018. Emergence of invariance and disentanglement in deep representations. *Journal of Machine Learning Research* 19 (50):1–34. http://jmlr.org/papers/v19/17-646.html.

Amodei, D., S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, et al. 2016. Deep speech 2: End-to-end speech recognition in English and Mandarin. In *Proceedings of The 33rd International Conference on Machine Learning*, eds. M. F. Balcan and K. Q. Weinberger, 173–182. New York, NY: PMLR.

Bahdanau, D., K. Cho, and Y. Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bishop, C. M. 2006. *Pattern recognition and machine learning*. 1st ed. New York, NY: Springer.

Chang, S. I., and S. Yadama. 2010. Statistical process control for monitoring non-linear profiles using wavelet filtering and b-spline approximation. *International Journal of Production Research* 48 (4):1049–1068. doi: 10.1080/00207540802454799.

Chen, Q., U. Kruger, M. Meronk, and A. Leung. 2004. Synthesis of t2 and q statistics for process monitoring. *Control Engineering Practice* 12 (6):745–755. doi: 10.1016/j.conengprac.2003.08.004.

Chiang, L. H., E. L. Russell, and R. D. Braatz. 2001. *Fault detection and diagnosis in industrial systems*. 1st ed. London: Springer.

Chung, J., K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio. 2015. A recurrent latent variable model for sequential data. In *Proceedings of the Advances in Neural Information Processing Systems 28*, Vol. 28, 2980–2988. Red Hook, NY: Curran Associates Inc.

Cybenko, G. 1989. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems* 2 (4):303–314. doi: 10.1007/BF02551274.

Eldan, R., and O. Shamir. 2016. The power of depth for feedforward neural networks. In *Proceedings of the 29th Conference on Learning Theory*, eds. V. Feldman, A. Rakhlin, & O. Shamir, 907–940. New York: PMLR.

Grasso, M., B. Colosimo, and M. Pacella. 2014. Profile monitoring via sensor fusion: the use of PCA methods for multi-channel data. *International Journal of Production Research* 52 (20):6110–6135. doi: 10.1080/00207543.2014.916431.

Hershey, J. R., and P. A. Olsen. 2007. Approximating the kullback leibler divergence between gaussian mixture models. Paper presented at the 2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07, 317–320. Honolulu, HI/USA.

Higgins, I., L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. 2017. beta-vae: Learning basic visual concepts with a constrained variational framework. Paper Presented at Proceedings of the 5th International Conference on Learning Representations, ICLR 2017. Toulon, France. https://openreview.net/forum?id=Sy2fzU9gl.

Hornik, K. 1991. Approximation capabilities of multilayer feedforward networks. *Neural Networks* 4 (2):251–257. doi: 10.1016/0893-6080(91)90009-T.

Howard, P., D. W. Apley, and G. Runger. 2018. Identifying nonlinear variation patterns with deep autoencoders. *IISE Transactions* 50 (12):1089–1103. doi: 10.1080/24725854.2018.1472407.

Jensen, W. A., and J. B. Birch. 2009. Profile monitoring via nonlinear mixed models. *Journal of Quality Technology* 41 (1):18–34. doi: 10.1080/00224065.2009.11917757.

Kim, D., and I.-B. Lee. 2003. Process monitoring based on probabilistic PCA. *Chemometrics and Intelligent Laboratory Systems* 67 (2):109–123. doi: 10.1016/S0169-7439(03)00063-7.

Kingma, D. P., and J. Ba. 2015. Adam: A method for stochastic optimization. Poster presented at 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA.

Kingma, D. P., and M. Welling. 2014. Auto-encoding variational bayes. Paper presented at the Second International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 15.

Kingma, D. P., and M. Welling. 2019. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning* 12 (4):307–392. doi: 10.1561/2200000056.

Krizhevsky, A., I. Sutskever, and G. E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In Paper presented at Twenty-sixth Annual Conference on Neural Information Processing Systems, eds. P. L. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger, 1106–1114. Lake Tahoe, NV/United States.

LeCun, Y., B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural Computation* 1 (4):541–551. doi: 10.1162/neco.1989.1.4.541.

Lee, S., M. Kwak, K.-L. Tsui, and S. B. Kim. 2019. Process monitoring using variational autoencoder for high-dimensional nonlinear processes. *Engineering Applications of Artificial Intelligence* 83:13–27. doi: 10.1016/j.engappai.2019.04.013.

Liu, R. Y. 1995. Control charts for multivariate processes. *Journal of the American Statistical Association* 90 (432):1380–1387. doi: 10.1080/01621459.1995.10476643.

Locatello, F., S. Bauer, M. Lucic, G. Raetsch, S. Gelly, B. Schölkopf, and O. Bachem. 2019. Challenging common assumptions in the unsupervised learning of disentangled representations. In *Proceedings of the 36th International Conference on Machine Learning*, Vol. 97, 4114–4124. Long Beach, CA/USA: PMLR.

Maleki, M. R., A. Amiri, and P. Castagliola. 2018. An overview on recent profile monitoring papers (2008–2018) based on conceptual classification scheme. *Computers & Industrial Engineering* 126:705–728. http://www.sciencedirect.com/science/article/pii/S0360835218304789. doi: 10.1016/j.cie.2018.10.008.

Noorossana, R., A. Saghaei, and A. Amiri, eds. 2011. *Statistical analysis of profile monitoring*. Hoboken, NJ: John Wiley & Sons, Inc. doi: 10.1002/9781118071984.

Paynabar, K., and J. Jin. 2011. Characterization of non-linear profiles variations using mixed-effect models and wavelets. *IIE Transactions* 43 (4):275–290. doi: 10.1080/0740817X.2010.521807.

Paynabar, K., J. Jin, and M. Pacella. 2013. Monitoring and diagnosis of multichannel nonlinear profile variations using uncorrelated multilinear principal component analysis. *IIE Transactions* 45 (11):1235–1247. doi: 10.1080/0740817X.2013.770187.

Paynabar, K., C. Zou, and P. Qiu. 2016. A change-point approach for phase-i analysis in multivariate profile monitoring and diagnosis. *Technometrics* 58 (2):191–204. doi: 10.1080/00401706.2015.1042168.

Qiu, P., C. Zou, and Z. Wang. 2010. Nonparametric profile monitoring by mixed effects modeling. *Technometrics* 52 (3):265–277. doi: 10.1198/TECH.2010.08188.

Radford, A., L. Metz, and S. Chintala. 2016. Unsupervised representation learning with deep convolutional generative adversarial networks. Paper presented at the Fourth International Conference on Learning Representations, San Juan, Puerto Rico, May 4.

Roweis, S., and Z. Ghahramani. 1999. A unifying review of linear gaussian models. *Neural Computation* 11 (2):305–345. doi: 10.1162/089976699300016674.

Shi, Z., D. W. Apley, and G. C. Runger. 2016. Discovering the nature of variation in nonlinear profile data. *Technometrics* 58 (3):371–382. doi: 10.1080/00401706.2015.1049751.

Tipping, M. E., and C. Bishop. 1999. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 61 (3):611–622. https://www.microsoft.com/en-us/research/publication/probabilistic-principal-component-analysis/. doi: 10.1111/1467-9868.00196.

Wang, K., M. G. Forbes, B. Gopaluni, J. Chen, and Z. Song. 2019. Systematic development of a new variational autoencoder model based on uncertain data for monitoring nonlinear processes. *IEEE Access* 7:22554–22565. doi: 10.1109/ACCESS.2019.2894764.

Williams, J. D., W. H. Woodall, and J. B. Birch. 2007. Statistical monitoring of nonlinear product and process quality profiles. *Quality and Reliability Engineering International* 23 (8):925–941. doi: 10.1002/qre.858.

Woodall, W. H. 2007. Current research on profile monitoring. *Production* 17 (3):420–425. doi: 10.1590/S0103-65132007000300002.

Woodall, W. H., D. J. Spitzner, D. C. Montgomery, and S. Gupta. 2004. Using control charts to monitor process and product quality profiles. *Journal of Quality Technology* 36 (3):309–320. doi: 10.1080/00224065.2004.11980276.

Yan, H., K. Paynabar, and J. Shi. 2015. Image-based process monitoring using low-rank tensor decomposition. *IEEE Transactions on Automation Science and Engineering* 12 (1):216–227. doi: 10.1109/TASE.2014.2327029.

Yan, H., K. Paynabar, and J. Shi. 2018. Real-time monitoring of high-dimensional functional data streams via spatio-temporal smooth sparse decomposition. *Technometrics* 60 (2):181–197. doi: 10.1080/00401706.2017.1346522.

Yan, W., P. Guo, L. Gong, and Z. Li. 2016. Nonlinear and robust statistical process monitoring based on variant autoencoders. *Chemometrics and Intelligent Laboratory Systems* 158:31–40. doi: 10.1016/j.chemolab.2016.08.007.

Zhang, Z., T. Jiang, S. Li, and Y. Yang. 2018. Automated feature learning for nonlinear process monitoring – an approach using stacked denoising autoencoder and k-nearest neighbor rule. *Journal of Process Control* 64: 49–61. doi: 10.1016/j.jprocont.2018.02.004.

Zhang, Z., T. Jiang, C. Zhan, and Y. Yang. 2019. Gaussian feature learning based on variational autoencoder for improving nonlinear process monitoring. *Journal of Process Control* 75:136–155. doi: 10.1016/j.jprocont.2019.01.008.

Zhu, J., and D. K. J. Lin. 2009. Monitoring the slopes of linear profiles. *Quality Engineering* 22 (1):1–12. doi: 10.1080/08982110903344804.

Zou, C., X. Ning, and F. Tsung. 2012. LASSO-based multivariate linear profile monitoring. *Annals of Operations Research* 192 (1):3–19. doi: 10.1007/s10479-010-0797-8.

Zou, C., F. Tsung, and Z. Wang. 2008. Monitoring profiles based on nonparametric regression methods. *Technometrics* 50 (4):512–526. doi: 10.1198/004017008000000433.

Appendix A: Proof of proposition 3.1

The Kullback-Leibler divergence between two multivariate Gaussian distributions has a closed-form solution. If we define these distributions as $p_0 = N(\boldsymbol{z}; \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ and $p_1 = N(\boldsymbol{z}; \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are respective mean vectors and covariance matrices, then according to Hershey and Olsen (2007) the closed-form solution will be the following:

$$\mathrm{KL}(p_0 \mid\mid p_1) = \frac{1}{2} \left[\log \frac{\mid \boldsymbol{\Sigma}_1 \mid}{\mid \boldsymbol{\Sigma}_0 \mid} + Tr(\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\Sigma}_0) - r + (\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)^\top \boldsymbol{\Sigma}_1^{-1} (\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1) \right] \tag{A1}$$

Since $q_\phi(\boldsymbol{z} \mid \boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{x}), \boldsymbol{\Sigma}_z)$ and $p(\boldsymbol{z}) = \mathcal{N}(0, \boldsymbol{I})$, we can derive that

$$\begin{aligned} \mathrm{KL}(q_\phi(\boldsymbol{z} \mid \boldsymbol{x}) \mid\mid p(\boldsymbol{z})) &= \frac{1}{2}[-\log \mid \boldsymbol{\Sigma}_z \mid + Tr(\boldsymbol{\Sigma}_z) - r] \\ &+ \frac{1}{2}\boldsymbol{\mu}(\boldsymbol{x})^\top \boldsymbol{\mu}(\boldsymbol{x}) \\ &= \frac{1}{2}\boldsymbol{\mu}(\boldsymbol{x})^\top \boldsymbol{\mu}(\boldsymbol{x}) + C, \end{aligned} \tag{A2}$$

where $C = -\log \mid \boldsymbol{\Sigma}_z \mid + Tr(\boldsymbol{\Sigma}_z) - r$ is a constant, which does not depend on $\boldsymbol{x}$.

To derive the SPE statistics, we will derive

$$\begin{aligned} \mathbb{E}_{z\sim q_\theta} \mid\mid \boldsymbol{x} - \boldsymbol{W}\boldsymbol{z} \mid\mid^2 &= \mathbb{E}_{z\sim q_\theta}(\boldsymbol{x}^\top \boldsymbol{x} - 2\boldsymbol{z}^\top \boldsymbol{W}\boldsymbol{x} + \boldsymbol{z}^\top \boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{z}) \\ &= \boldsymbol{x}^\top \boldsymbol{x} - 2\boldsymbol{\mu}(\boldsymbol{x})^\top \boldsymbol{W}\boldsymbol{x} + \mathbb{E}_{z\sim q_\theta}(\boldsymbol{z}^\top \boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{z}) \end{aligned} \tag{A3}$$

Here, we know that

$$\begin{aligned} \mathbb{E}_{z\sim q_\theta}(\boldsymbol{z}^\top \boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{z}) &= \mathbb{E}_{z\sim q_\theta} tr(\boldsymbol{z}^\top \boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{z}) \\ &= tr(\boldsymbol{W}^\top \boldsymbol{W} \mathbb{E}_{z\sim q_\theta}(\boldsymbol{z}\boldsymbol{z}^\top)) \\ &= tr(\boldsymbol{W}^\top \boldsymbol{W}(\boldsymbol{\mu}(\boldsymbol{x})\boldsymbol{\mu}(\boldsymbol{x})^\top + \Sigma_z)) \\ &= \boldsymbol{\mu}(\boldsymbol{x})^\top \boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{\mu}(\boldsymbol{x}) + tr(\boldsymbol{W}^\top \boldsymbol{W}\Sigma_z) \end{aligned} \tag{A4}$$

Therefore, by plugging Eq. [A4] into Eq. [A3], we have

$$\begin{aligned} \mathbb{E}_{z\sim q_\theta} \mid\mid \boldsymbol{x} - \boldsymbol{W}\boldsymbol{z} \mid\mid^2 &= \boldsymbol{x}^\top \boldsymbol{x} - 2\boldsymbol{\mu}(\boldsymbol{x})^\top \boldsymbol{W}\boldsymbol{x} + \mathbb{E}_{z\sim q_\theta}(\boldsymbol{z}^\top \boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{z}) \\ &= \boldsymbol{x}^\top \boldsymbol{x} - 2\boldsymbol{\mu}(\boldsymbol{x})^\top \boldsymbol{W}\boldsymbol{x} + \boldsymbol{\mu}(\boldsymbol{x})^\top \boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{\mu}(\boldsymbol{x}) + tr(\boldsymbol{W}^\top \boldsymbol{W}\Sigma_z) \\ &= \mid\mid \boldsymbol{x} - \boldsymbol{W}\boldsymbol{\mu}(\boldsymbol{x}) \mid\mid^2 + C \end{aligned} \tag{A5}$$

where $C = tr(\boldsymbol{W}^\top \boldsymbol{W}\Sigma_z)$ that does not depend on $\boldsymbol{x}$.

Appendix B: A toy example to demonstrate out-of-distribution behavior of neural networks

Assume using a multilayer perceptron, we are trying to approximate the famous Rosenbrock function $f(x, y) = (a - x)^2 + b(y - x^2)^2$ given $(a, b) = (1, 100)$. In this small experiment, we sample tuples of two-dimensional points from a bounded region $(x_i, y_i) \in [-1, 3] \times [-2, 3]$. We use a multilayer perceptron with six hidden layers and a 100 neurons in each layer. Half of the points are used in training, and the other half is used as a validation set to optimize hyperparameters. Using the trained network, we plot the actual Rosenbrock function along with the neural network approximation in Figure A1. Notice how well the function

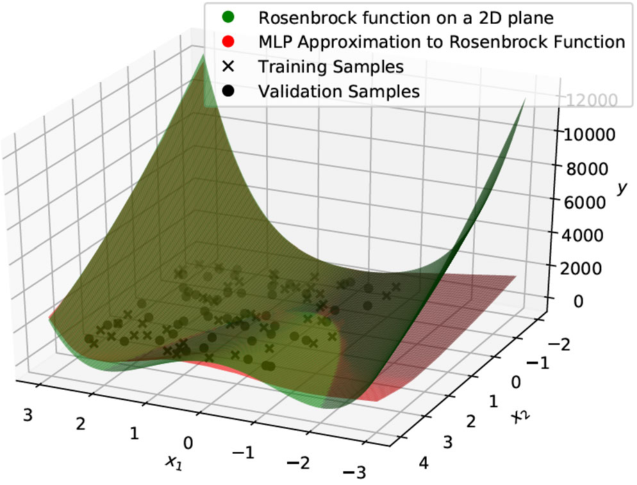


Figure A1. Rosenbrock function (green surface) approximated by an multilayer perceptron (red surface) given training (black crosses) and validation (black dots) samples form a bounded region $(x_i, y_i) \in [-1, 3] \times [-2, 3]$.

is approximated for the region $[-1, 3] \times [-2, 3]$, but there is a serious discrepancy between the approximated and the real outside of the region. This is a small yet to the point example of out-of-distribution issues with neural networks.

Appendix C: ERE testing statistic derivation

To derive the ERE_1 and ERE_2, we first define $R(z) = || \, y - \boldsymbol{\mu}_{\boldsymbol{\theta}}(z) \, || \, ^2$ as the reconstruction error (RE). The quantity we would like to approximate is $E_{z\sim q_\phi} R(\boldsymbol{z})$ where $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}), \Sigma_z)$. We are looking for the Taylor expansion of the expected RE (ERE) around $z_0 = \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x})$, that is, the first moment. For notational simplicity, we use H_z to denote the Hessian $R''(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}))$. The derivation is formalized as follows:

$$\begin{aligned} E_{z\sim q_\phi} R(\boldsymbol{z}) &= R(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x})) + R'(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x})) E_{z\sim q_\phi}\left[\boldsymbol{z} - \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x})\right]) \\ &+ \frac{1}{2} E_{z\sim q_\phi}\left[(\boldsymbol{z} - \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}))^\top \mathbf{H}_z(\boldsymbol{z} - \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}))\right] + \mathcal{O}(||(\boldsymbol{z} - \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x})||^3 \\ &\simeq R(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x})) + \frac{1}{2} E_{z\sim q_\phi}\left[(\boldsymbol{z} - \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}))^\top \mathbf{H}_z(\boldsymbol{z} - \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}))\right] \\ &= R(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x})) + \frac{1}{2} tr(\mathbf{H}_z E\left[(\boldsymbol{z} - \boldsymbol{\mu}_z)(\boldsymbol{z} - \boldsymbol{\mu}_z)^T\right]) \\ &= R(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x})) + \frac{1}{2} tr(\mathbf{H}_z \Sigma_z) \end{aligned} \tag*{[C1]}$$

Note for ERE_1, the second term $1/2(tr(\mathbf{H}_z\Sigma_z))$ is droped and we are left with $R(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}))$ only. For ERE_2, since Σ_z is a diagonal matrix, $tr(\mathbf{H}_z S_z) = tr(diag(\mathbf{H}_z) S_z) = \sum_i (\mathbf{H}_z)_{ii} (S_z)_{ii}$ holds. We can utilize this result to compute ERE_2, in a more computationally efficient manner.