Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2022.019521

ARTICLE

Enhancing the Effectiveness of Trimethylchlorosilane Purification Process Monitoring with Variational Autoencoder

Jinfu Wang[1], Shunyi Zhao[1,*], Fei Liu[1] and Zhenyi Ma[2]

[1]Key Laboratory of Advanced Process Control for Light Industry (Ministry of Education), Jiangnan University, Wuxi, 214122, China

[2]Hangzhou JASU Environmental Monitoring Co., Ltd., Hangzhou, 311999, China

[*]Corresponding Author: Shunyi Zhao. Email: shunyi.s.y@gmail.com

Received: 27 September 2021 Accepted: 30 December 2021

ABSTRACT

In modern industry, process monitoring plays a significant role in improving the quality of process conduct. With the higher dimensional of the industrial data, the monitoring methods based on the latent variables have been widely applied in order to decrease the wasting of the industrial database. Nevertheless, these latent variables do not usually follow the Gaussian distribution and thus perform unsuitable when applying some statistics indices, especially the T^2 on them. Variational AutoEncoders (VAE), an unsupervised deep learning algorithm using the hierarchy study method, has the ability to make the latent variables follow the Gaussian distribution. The partial least squares (PLS) are used to obtain the information between the dependent variables and independent variables. In this paper, we will integrate these two methods and make a comparison with other methods. The superiority of this proposed method will be verified by the simulation and the Trimethylchlorosilane purification process in terms of the multivariate control charts.

KEYWORDS

Process monitoring; variational autoencoders; partial least square; multivariate control chart

1 Introduction

With the increasing demand for product quality and its performance under different industrial environments, the industrial process has become more sophisticated in structure and process flow [1,2]. Therefore, it has been more challenging to ensure their security and reliability while using the model-based modeling approach based on physical and mathematical details [3]. As a substitute for the model-based approach, the data-based modeling approach is regarded as the statistical process control (SPC), which makes outstanding performance on monitoring univariable processes [4,5]. Nevertheless, the behavior of SPC is unsatisfactory while applying to the multivariable processes. For instance, the Type I error rate raises with the number of variables increasing [6]. With the diversification of system parameters, the consequence of this problem has been more and more severe and brings multivariable statistical process control (MSPC) into our sight [7–9].

This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The most commonly used MSPC charts are Hotelling T^2 chart and squared prediction error (SPE) chart. The value of T^2 is calculated as the following Eq. (1).

$$T^2 = (\mathbf{z} - \bar{\mathbf{z}})^T \mathbf{S_z}^{-1} (\mathbf{z} - \bar{\mathbf{z}}) \tag{1}$$

where the $\bar{\mathbf{z}}$ is the mean vector of the latent variable $\mathbf{z}$ and the $\mathbf{S_z}$ is the covariance matrix of $\mathbf{z}$.

Hotelling T^2 chart is an application of Mahalanobis Distance which is treated as an improved version of Euclidean distance and can solve the inconsistent and related problem in the dimensions of Euclidean distance [10]. The essence of the Mahalanobis distance can also be regarded as taking various relationships between characteristics into account, mapping the data into the $\mathcal{N}(0,1)$ interval, and finding their Euclidean distance [11]. The decision boundary of the T^2 chart can be seen as an ellipsoidal approximately. There has a hypothesis that the value of T^2 should follow the normal distribution [12]. If the condition is not satisfied, the calculation process cannot compress the data into a standard size thus the statistics index may incite a weak performance. The control limit of T^2 has followed the $\mathcal{F}$ distribution and been shown as the following Eq. (2).

$$CL_{T^2} = \frac{k\left(n^2-1\right)}{n\left(n-k\right)} \mathcal{F}_{(\alpha,k,n-k)} \tag{2}$$

where the k and n are the numbers of variables and data and α is the Type I error rate.

However, the situation is more complicated when the variables are correlated. The covariance matrix $\mathbf{S}$ is difficult to invert with these correlated data because the covariance matrix becomes almost a singular matrix, which shows poor fault detection performance [13]. To better deal with the high-correlated data, an important step is extracting underlying features encapsulated with the process data [14]. The latent variables extract important information by projecting the high-dimensional process data into a lower-dimensional space [15]. To effectively monitor process operating performance, one must utilize all the information contained in a large number of routine measurements on the process variables (X) as well as that in the infrequent final property measurements (Y) [16]. A PLS model-based process monitoring scheme is proposed when both predictor and response variables have to be accounted for simultaneously [17]. The extracted variables are linearly independent of each other, and these variables can represent most of the information in the original database. A partial least squares (PLS)-based control chart was developed to address this problem [18,19]. The elimination of PLS loading weights is achieved through the multivariable distribution Hotelling T^2, which utilizes the information from all used PLS components [20]. The PLS method is similar to the PCA method, which also has the extracted and residual variables. Different from the PCA algorithm that only uses the information in the independent variables, the principle of the PLS algorithm makes the latent variables not only well summarize the information of the independent variables but also have a strong ability to explain the dependent variables. We set the number of the extracted variables as m, the residual variables as n, and the total number as h.

To determine the Euclidean distance between the original data space and the space of extracted variables and emulate the loss caused by the residual variables [21], we introduce a control chart named SPE to measure the residual loss which has shown as the following Eq. (3).

$$SPE = \left(\mathbf{z}' - \mathbf{z}\right)\left(\mathbf{z}' - \mathbf{z}\right)^T \tag{3}$$

where $\mathbf{z}$ means the original vector and $\mathbf{z}'$ means the vector reconstructed by latent variable.

The following Eq. (4) computes the control limit of the SPE chart:

$$CL_{SPE} = \frac{\mathbf{b}}{2\mathbf{a}} \chi^2_{(2\mathbf{a}^2/\mathbf{b},\alpha)} \tag{4}$$

where $\mathbf{a}$ and $\mathbf{b}$ are the mean vector and covariance matrix of SPE and α is the Type I error rate.

Besides the SPE index, we also use the index T^2 to measure the Mahalanobis distance. The value of T^2 is computed by the hidden variables extracted by PLS. When we imply the process data $\mathbf{Z}$ into the PLS model, $\mathbf{Z}$ will be disintegrated into score matrix $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_h]$, loading matrix $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \cdots, \mathbf{d}_h]$ where h is the number of the extracted variables. The Eq. (5) is shown as follows:

$$\mathbf{Z} = \mathbf{P}_m \mathbf{D}_m^T + \mathbf{F} \tag{5}$$

where the matrix $\mathbf{F}$ is based on the residual space and can be used to weigh the disturbance of the process noise. The first m variables can be used to calculate the following PLS-based T^2 statistic which has shown as the following Eq. (6).

$$T^2_{PLS} = \sum_{i=1}^{m} \frac{\mathbf{p_i}^2}{\mathbf{s}^2_{\mathbf{p_i}}} \tag{6}$$

where s_{p_i} means the covariance of $\mathbf{u_i}$. And the T^2_{PLS} index is equal to the T^2 index while the m is equivalent to h. With the hypothesis that the input of T^2_{PLS} follows the Gaussian distribution, we can compute the control limit of T^2_{PLS} chart by the following Eq. (7).

$$CL_{T^2_{PLS}} = \frac{m\left(h^2-1\right)}{h\left(h-m\right)} \mathcal{F}_{(\alpha,m,h-m)} \tag{7}$$

where α is the Type I error rate.

For these two statistics indices, T^2_{PLS} is in charge of monitoring the process and detecting the faults triggered by extracted variables, while SPE_{PLS} is responsible for residual variables. When these indices are more significant than the limit computed by the above equation, the process is out of control, and some faults have been detected in PLS-based charts [22]. However, there are two shortcomings of PLS-based charts. First, the PLS method does not perform well when applied to the nonlinear process. Second, due to the assumption made in the derivation process, the T^2 index severely depends on whether the input data follow the Gaussian distribution.

To solve these sticky problems, we find autoencoder (AE) as an effective solution. As an essential sub-field in data analysis [23], AE acts as a bridge that can connect big machinery data and intelligent monitoring [24]. Its unsupervised learning property signifies it does not require label data [25] which is hard to obtain in modern industry processes [26]. AE is composed of the input layers called the encoder, the hidden layer, and the output layers called the decoder [27]. The encoder is an efficient dimensional reduction algorithm for high-dimensional data [28]. It decreases the dimension not in the way that reduces some dimensions by mathematical calculation. Instead, it mapped the data to a low dimension state by encoding and is able to decode these reduced data to the original dimension through the decoder [29]. As a result, AE has a significant function that its output is equivalent to its input [30]. The operating principle of its encoder can be regarded as a combination of several neural networks which can resolve the nonlinear problem because their

mapping structure and activation function is used to add nonlinear factors to argue the expression ability of these networks.

Another problem is that the input data of T^2 and SPE usually do not follow the Gaussian distribution. To tackle it, we bring the Variational AutoEncoder (VAE) into our sight. The VAE model as a generative model has been widely applied to many fields [31], and its mathematical basis is Bayesian inference [32]. Different from the hidden layer of AE, the VAE uses the neural networks of the encoder to generate two hyperparameters μ and σ which are extracted from input data [33]. These two parameters participate in building the latent distribution $\mathcal{N}(\mu,\sigma)$ thus the latent variables that are sampled from this distribution follow the Gaussian distribution. Then we input these latent variables instead of the input data into the control chart to solve the last difficulty. In summary, the application of the VAE model can solve both the nonlinear and normal distribution problems. In this way, many kinds of derivative versions of VAE have been proposed to fit different industrial demands. Hemmer et al. [34] use conditional VAE to monitor the health condition of low-speed axial rolling element bearing. Yan et al. [35] apply the deep regularized VAE to rotor-bearing system for intelligent fault diagnosis. Zhang et al. [36] propose a novel deep Semi-Supervised method of Multi-Layer VAE framework of planetary gearbox vibration-based fault diagnosis. Cheng et al. [37] dish variational recurrent autoencoder and compare to other neural network based methods on the benchmark simulated Tennessee Eastman process. Zhu et al. [38] draw a method named information concentrated variational autoencoder on Tennessee eastman process.

In this paper, we proposed a method that combined the PLS and VAE. The combination algorithm will apply to the trimethylchlorosilane purification process to verify its superiority over other traditional algorithms. The characteristic of the PLS algorithm represents that the monitoring statistics are determined by both the independent variable and dependent variable. It means that its latent variable contains the information of both dependent and independent variables, which is different from the latent variable extracted by the VAE algorithm that is only correlated with the independent variable. We use the PLS to extract the more informative latent variable and bring it as the input dataset of VAE. And the VAE algorithm maps the input dataset to latent distribution that follows the Gaussian distribution. In this way, we obtain the latent variable that is more informative and follows the hypothesis of the monitoring indices. The flow chart of our proposed algorithm is shown in Fig. 1.

The contributions of this paper are displayed as follows:

(1) We integrate the PLS method with the VAE method. The main idea of this method is to use the PLS method to transfer the data from the original space to a low-dimensional space that can save the correlation information between the independent and dependent variables. And applying the VAE method, which can address the difficulty that the extracted latent variables do not follow the Gaussian distribution and nonlinear process. To integrate the PLS and VAE method, we first use the PLS method to decrease the data dimensions, then apply these processed data to VAE to extract their latent variables and calculate the statistic index T^2 and SPE to actualize the process monitoring.
(2) We make a comparison in both simulation and real industrial process with some other methods, which are also on the basis of the latent variables. To verify the superiority of the proposed method, we use the false alarm rate, which statistics indices are more significant than the calculated control limit, to measure the accuracy of these methods, and their result is displayed in the form of the picture.

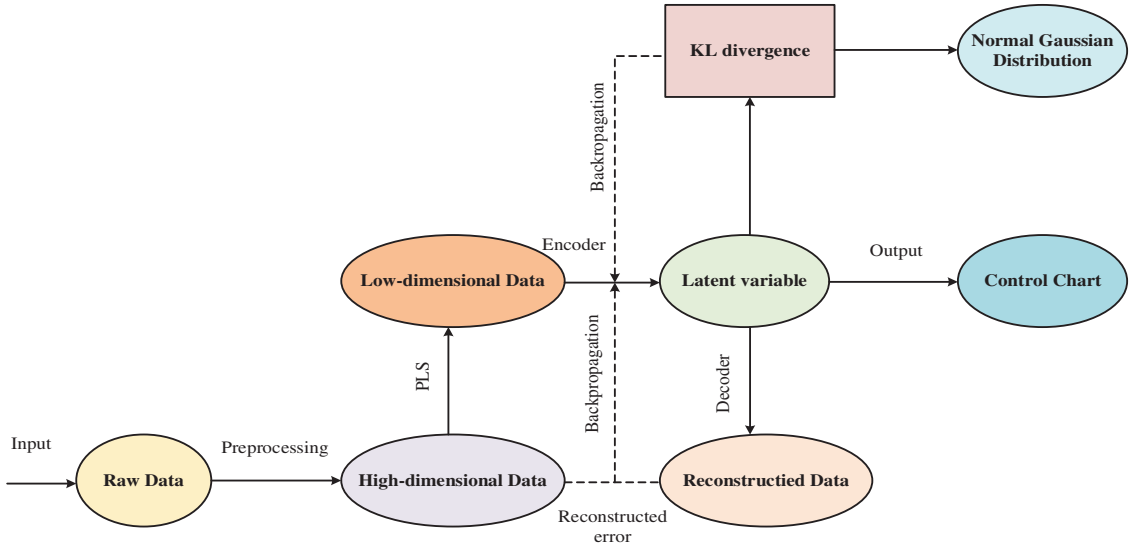


Figure 1: The flow chart of proposed algorithm

The rest of this paper is organized as follows. In Section 2, we will introduce the principle and structure of the PLS and VAE model. Then we will represent the PLS algorithm in Algorithm 1, the VAE algorithm in Algorithm 2 and demonstrate how to integrate these two algorithms with T^2 and SPE. In Section 3, we generate some sampling data points from several distributions and use them to make a simulation to test the performance of the PLS-VAE method and three other methods to make a comparison. In Section 4, we use real industrial process data of Trimethylchlorosilane purification to check the superiority of our proposed methods. And the conclusion is demonstrated in Section 5.

Algorithm 1: The step of PLS algorithm

Data: $\mathbf{X}; \mathbf{Y}$

Results: $\mathbf{D} = \{\mathbf{d}_1, \mathbf{d}_2, \cdots \mathbf{d}_k\}$, $\mathbf{M} = \{\mathbf{m}_1, \mathbf{m}_2, \cdots \mathbf{m}_k\}$, $\mathbf{R} = \{\mathbf{r}_1, \mathbf{r}_2, \cdots \mathbf{r}_k\}$, $\mathbf{H} = \{\mathbf{h}_1, \mathbf{h}_2, \cdots \mathbf{h}_k\}$

Main steps:

Compute $\mathbf{w}_i$ as the maximize the eigenvectors of the matrix $\mathbf{X^T Y Y^T X}$

Compute the number k though the cross-validation test

$\mathbf{X_1} = \mathbf{X}, \mathbf{r}_1 = \mathbf{w}_1;$

$\mathbf{m}_i = \mathbf{X}_i \mathbf{w}_i;$

$\mathbf{X}_{i+1} = \mathbf{X}_i - \mathbf{t}_i \mathbf{p}_i^T;$

$\mathbf{d}_i = \frac{\mathbf{X}_i^T \mathbf{m}_i}{\mathbf{m}_i^T \mathbf{m}_i};$

$\mathbf{r}_i = \prod_{j=1}^{i-1} \left(\mathbf{I}_{n \times n} - \mathbf{w}_j \mathbf{p}_j^T \right) \mathbf{w}_i, i > 1;$

$\mathbf{h}_i = \frac{\left(\mathbf{r}_i^T \mathbf{X}_i \right)^T}{\mathbf{t}_i^T \mathbf{t}_i};$

Algorithm 2: The main step of VAE algorithm

Data: Training data set $\mathbf{x}_{train}$; testing data set $\mathbf{x}_{test}$; Type I error rate α
Result: Generated data set $\mathbf{X}'$; latent variables $\mathbf{z}$
Purpose: Initializing the parameters θ and ϕ of the encoder $\mathbf{Q}_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \mu_\mathbf{z}, \sigma_\mathbf{z})$ and the decoder $\mathbf{P}_\theta(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}; \mu_\mathbf{x}, \sigma_\mathbf{x})$ by training the training data set $\mathbf{x}_{train}$
Main idea: Update the parameters θ and ϕ by minimizing the index $\mathbb{L}$ until the parameters converge; $\mathbb{L} = KL\left(Q_\phi(\mathbf{z}|\mathbf{x_{train}}) || P(\mathbf{z})\right) - \mathbb{E}_{\mathbf{z}\sim Q_\phi(\mathbf{z}|\mathbf{x})}[\log P_\theta(\mathbf{x}|\mathbf{z})]$
Specific steps:
for $i = 1, 2, \cdots, n$ **do:**
Extract features: Extract the Gaussian feature $\mu_\mathbf{z}$ and $\sigma_\mathbf{z}$ from training data set $\mathbf{x}_{train}$;
Reparameterization: $\mathbf{z} = \mu_\mathbf{z} + \varepsilon \Sigma_\mathbf{z}^{1/2}$;
Generate: New data set $\mathbf{x}'$ is generated by decoding the latent variables $\mathbf{z}$;
Update: Calculate the value of $\mathbb{L}$ and try to decrease it by iterating n times;
end for
Monitoring Index $\mathbf{T^2}$ and SPE:
for $i = 1, 2, \cdots, n$ **do:**
$T^{2^i} = \left(\mathbf{z}^i - \overline{\mathbf{z}}\right)^T \mathbf{S_z}^{-1} \left(\mathbf{z}^i - \overline{\mathbf{z}}\right)$,
$SPE^i = \left(\mathbf{x}_{test}^i - \mu_x^i\right)\left(\mathbf{x}_{test}^i - \mu_x^i\right)^T$,
end for
The control limit of $\mathbf{T^2}$ chart: $CL_{T^2} = \frac{k(n^2-1)}{n(n-k)} \mathcal{F}_{(\alpha,k,n-k)}$
The control limit of SPE chart: $CL_{SPE} = \frac{b}{2a} \chi^2_{(2a^2/b,\alpha)}$
Let latent variables $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_k\}$, where k means the number of the dimensions of latent variable, $\overline{\mathbf{z}} = \mathbb{E}[\mathbf{Z}]$ means the expectation vector of variables $\mathbf{z}$, $\mathbf{S_z} = \frac{1}{m-1}\sum_{i=1}^{m}(\mathbf{z}_i - \overline{\mathbf{z}})(\mathbf{z}_i - \overline{\mathbf{z}})^T$ is the samples covariance matrix and m is the number of samples.
The latent variables $\mathbf{H} = \{SPE_1, SPE_2, \cdots, SPE_m\}$, where m means the number of the dimensions of original data set. $\mathbf{a} = \mathbb{E}[\mathbf{H}]$ means the expectation vector of variables $\mathbf{H}.\mathbf{b} = \mathbb{E}\left[(\mathbf{H} - \mathbf{a})(\mathbf{H} - \mathbf{a})^T\right]$ represents the covariance matrix of variables $\mathbf{H}$.
Criterion:
for $i = 1, 2, \cdots, n$ **do:**
While $T^{2^i} \leq CL_{T^2}$ or $SPE^i \leq CL_{SPE}$:
the process is in-control in the time point n
While $T^{2^i} > CL_{T^2}$ and $SPE^i > CL_{SPE}$,:
the process is out-control in the time point n
end for

2 Preliminaries and Algorithm

2.1 Partial Least Squares (PLS)

The partial least squares (PLS) algorithm has some properties similar to principal component analysis (PCA), which is used to decrease the dimensions of the original data. The PLS algorithm can be used to make regression modeling analysis, reduce the data dimension, and correlation analysis which means that PLS integrates the using fields and advantages of the multiple linear regression algorithms, principal component analysis algorithm, canonical correlation analysis algorithm.

The PLS is widely used when the input vector $\mathbf{X}$ has a high correlation with the output vector $\mathbf{Y}$. The essential function of the PLS is transforming the inter-related and col-linear data to several uncorrelated and uni-variate data vectors. To achieve this target, the PLS projects the high-dimensional data space of the independent variable and the dependent variable to the corresponding low-dimensional space, extracts the principal components $\mathbf{t_n}$ and $\mathbf{u_n}$ which represent the input and output vectors, respectively, and then establishes a regression relationship between them. The $\mathbf{t_n}$ and $\mathbf{u_n}$ should follow the claims that they carry the original information in their respective data vectors as much as possible. The degree of correlation between $\mathbf{t_n}$ and $\mathbf{u_n}$ should be maximized. In this way, $\mathbf{t_n}$ and $\mathbf{u_n}$ can represent the variables $\mathbf{X}$ and $\mathbf{Y}$ mostly, and component $\mathbf{t_n}$ of the independent variable has the strongest explanatory power for component $\mathbf{u_n}$ of the dependent variable.

The independent variable $\mathbf{X}$ and the dependent variable $\mathbf{Y}$ can be written as the following Eqs. (8) and (9).

$$\mathbf{X} = \mathbf{m_1}\mathbf{d_1}^T + \cdots + \mathbf{m_n}\mathbf{d_n}^T + \mathbf{C_{n+1}} = \mathbf{M}\mathbf{D}^T + \mathbf{C} \tag{8}$$

$$\mathbf{Y} = \mathbf{w_1}\mathbf{h_1}^T + \cdots + \mathbf{w_n}\mathbf{h_n}^T + \mathbf{B_{n+1}} = \mathbf{W}\mathbf{H}^T + \mathbf{B} \tag{9}$$

where n is the number of components extracted from variables and its value is determined by a kind of statistical techniques such as cross-validation. $\mathbf{P}^T$ and $\mathbf{Q}^T$ represent the loading matrices while $\mathbf{T}$ and $\mathbf{U}$ are the score matrices. The $\mathbf{t_1}$, $\mathbf{p_1}^T$, $\mathbf{u_1}$, $\mathbf{q_1}^T$ are the i^{th} vector of these matrices respectively while the $\mathbf{C}$ and $\mathbf{B}$ mean the residues matrices of $\mathbf{X}$ and $\mathbf{Y}$.

2.2 Variational AutoEncoder (VAE)

Variational AutoEncoder (VAE) has been widely used in data analysis which is essential in data-based modeling. The VAE can be used as a generative model whose purpose is to produce a database $\mathbf{x}'$ similar to the original database $\mathbf{x}$.

The generative procedure can be described in two steps. First, a distribution $p_\theta(\mathbf{z})$ will be gained, which is the distribution of the latent variables $\mathbf{z}$. The latent variables are unobserved and represent some of the hidden features of the original database $\mathbf{x}$. Second, the target database $\mathbf{x}'$ will be generated from these latent variables $\mathbf{z}$ in the inverse way that the $\mathbf{z}$ is generated. And the first step is named as encoder, while the second step is often called decoder. The reason for these name methods is that the main action of the first step is the same as coding $\mathbf{x}$ to $\mathbf{z}$ while the second step is similar to decoding $\mathbf{x}'$ from $\mathbf{z}$.

The distribution $P(\mathbf{x})$ which is mentioned above is shown as the following Eq. (10).

$$\log P_\theta(\mathbf{x}) = D_{KL}\left[Q_\phi(\mathbf{z}|\mathbf{x}) \,||\, P_\theta(\mathbf{z}|\mathbf{x})\right] + \mathbb{E}_{\mathbf{z}\sim Q_\phi(\mathbf{z}|\mathbf{x})}\left[\log P_\theta(\mathbf{x}|\mathbf{z})\right] - D_{KL}\left[Q_\phi(\mathbf{z}|\mathbf{x}) \,||\, P_\theta(\mathbf{z})\right] \tag{10}$$

where θ and ϕ represent the parameters of the network layers of the encoder and decoder. $P_\theta(\mathbf{z})$ is the prior distribution, $P_\theta(\mathbf{x})$ is the likelihood that will be maximized in the following steps, $P_\theta(\mathbf{x}|\mathbf{z})$ is a conditional distribution, and $P_\theta(\mathbf{z}|\mathbf{x})$ is the posterior distribution. The $\mathbb{E}$ means the mathematical expectation of the distribution, while the D_{KL} will be introduced in the Eq. (11).

Now, we should find the variational lower bound of $\log P_\theta(\mathbf{x})$ and we find that when we want to maximize an objective function, it can be achieved by maximizing its lower bound. So our purpose is transferring to maximize the $\log P_\theta(\mathbf{x})$. When the neural network is well trained, the first term can be approximate zero. The second term encourages the network to decrease the reconstruction loss in order to get a database that is more similar to the original one. The last term forces the distribution $Q_\phi(\mathbf{z}|\mathbf{x})$ to get close to $P_\theta(\mathbf{z})$, which is a normal Gaussian distribution. As a result, the latent variables $\mathbf{z}$ will follow the Gaussian distribution.

And that means the features of the original database that follow Gaussian distribution can be extracted by doing these, which is beneficial in applying the common statistics index like T^2 and SPE on the process.

In our study, a simple distribution is often used to approximate a complex distribution. At this time, we need a quantity to measure how much information the approximate distribution we choose has lost compared to the original distribution, which is how the KL divergence works. By constantly changing the parameters of the approximate distribution, we can get different values of KL divergence. In a specific range of variation, when the KL divergence reaches the minimum value, the corresponding parameter is the optimal parameter we want. When the P and Q are both continuous distribution, the $p(x)$ and $q(x)$ are their probability density function respectively. The KL divergence can be represented as follows:

$$D_{KL}(P||Q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx \tag{11}$$

As is shown in Fig. 2, the encoder $Q_\phi(\mathbf{z}|\mathbf{x})$ is consists of several neural network layers, and every layer has activation functions to make input nonlinear. By doing this, the input data $\mathbf{x}$ can be mapped to latent variable. And one of the most important steps of VAE is extracting the mean $\mu_\mathbf{z}$ and the covariance $\Sigma_\mathbf{z}$ from these data. After getting the mean and covariance, the latent variable $\mathbf{z}$ can be sampled from the Gaussian distribution consisting of the mean and covariance as the following Eq. (12).

$$\mathbf{z} \sim \mathcal{N}(\mu_\mathbf{z}, \Sigma_\mathbf{z}) \tag{12}$$

We denote the weight matrix as $\mathbf{W}_\mathbf{z}^i$ and the bias vector as $\mathbf{b}_h^i$, which are used to connect the node in the i^{th} hidden layer to the node h in the $(i+1)^{th}$ hidden layer in the encoder, for $i = 1, 2, \cdots, k$. At this case, we set the L^{th} hidden layer as the last layer for the encoder and the output of the encoder is shown as the following Eqs. (13) and (14).

$$\mu_{\mathbf{z}_i} = \mathbf{W}_{\mu_{\mathbf{z}_i}}^L \mathbf{h}^L + \mathbf{b}_{\mu_{\mathbf{z}_i}}^L \tag{13}$$

$$\sigma_{\mathbf{z}_i} = \mathbf{W}_{\sigma_{\mathbf{z}_i}}^L \mathbf{h}^L + \mathbf{b}_{\sigma_{\mathbf{z}_i}}^L \tag{14}$$

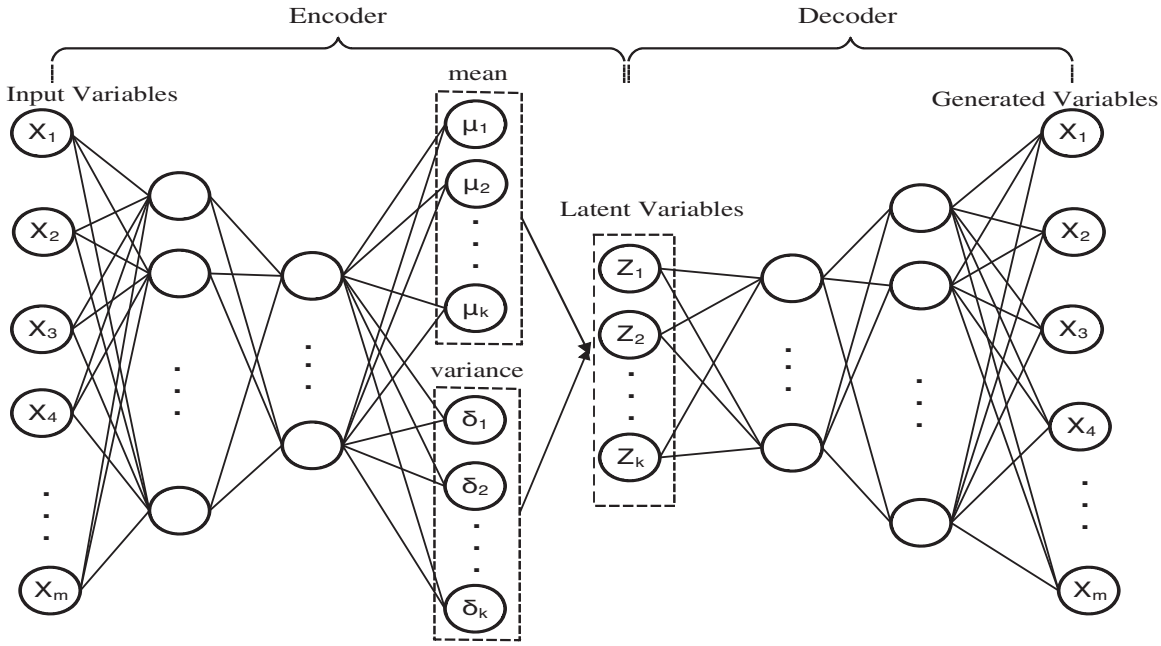


Figure 2: The architecture of variational AutoEncoder

And the nodes in the l^{th} hidden layer can be calculated by their last layer as the following Eq. (15).

$$h_i^l = \mathbf{S}\left(\mathbf{W}_i^{l-1}\mathbf{h}^{l-1} + \mathbf{b}_i^{l-1}\right) \tag{15}$$

where the $\mathbf{S}$ is an activation function that introduces nonlinear characteristics into our network. The typical activation function includes sigmoid function, tanh function, Rectified Linear Unit (ReLU) function, etc.

During the training process, the model needs to back-propagate the error through all the layers in order to decrease the error. However, the layer that sampling $\mathbf{z}$ is discontinuous and thus does not have a gradient, which will hinder the back-propagate. To solve this problem, a trick called the reparameterization trick was used. This trick makes $\mathbf{z}$ a definite node by introducing another node ε that follows the Gaussian distribution $\mathcal{N}(0, \mathbf{I})$, where $\mathbf{I}$ is the identity matrix with approximate dimensions. The sampling process now takes place in node ε rather than node $\mathbf{z}$. The back-propagate can be executed smoothly. The reparameterization trick is shown as the following Eq. (16).

$$\mathbf{z} = \mu_{\mathbf{z}} + \varepsilon \Sigma_{\mathbf{z}}^{1/2} \tag{16}$$

where $\left(\Sigma_{\mathbf{z}}^{1/2}\right)\left(\Sigma_{\mathbf{z}}^{1/2}\right)^T = \Sigma_{\mathbf{z}}$.

After gaining the Gaussian feature μ, Σ and then generating the latent variables $\mathbf{z}$, we can use $\mathbf{z}$ to construct some process monitoring control charts, such as T^2 chart. The equation of T^2 and its control limit is represented as the following Eqs. (17) and (18).

$$T^2 = (\mathbf{z} - \overline{\mathbf{z}})^T \mathbf{S}_{\mathbf{z}}{}^{-1} (\mathbf{z} - \overline{\mathbf{z}}) \tag{17}$$

$$CL_{T^2} = \frac{k\left(n^2-1\right)}{n\left(n-k\right)}\mathcal{F}_{(\alpha,k,n-k)} \tag{18}$$

where $\bar{\mathbf{z}}$ and $\mathbf{S_z}$ are the mean vector and the covariance matrix of the latent variables $\mathbf{z}$, n is the number of the observed data, k is the dimension of the latent variables $\mathbf{z}$ and α is the Type I error rate.

The decoder is the opposite of the encoder both the structure and the transferred information. The input of the decoder is the latent variables $\mathbf{z}$ sampling from $\mathcal{N}(\mathbf{z}|\mu_{\mathbf{z}}, \sigma_{\mathbf{z}})$, while the output is the reconstructed data which is similar to the original database. In this case, we set the L^{th} hidden layer as the last layer for the decoder and m as the node in the L^{th} hidden layer. The structure of the network of the decoder can be shown as the following Eq. (19).

$$\mathbf{x}_i' = \mathbf{S}'\left(\mathbf{W}_i'^{L}\mathbf{m}^{L} + \mathbf{b}_i^{L}\right) \tag{19}$$

Because the generated data set is printed by the decoder $\mathbf{P}_\theta(\mathbf{z}|\mathbf{x})$ from $\mathbf{z}$, it can be treated as a probabilistic decoder. That means the data set $\mathbf{x}'$ can be regarded as $\mathcal{N}\left(\mathbf{x}'|\mu_{\mathbf{x}'}, \sigma_{\mathbf{x}'}\right)$. The mean vector $\mu_{\mathbf{x}'}$ and the variance matrix $\sigma_{\mathbf{x}'}$ is shown as the following Eqs. (20) and (21).

$$\mu_{\mathbf{x}_i'} = \mathbf{W}_{\mu_{\mathbf{x}_i'}}^{L'}\mathbf{h}^{L'} + \mathbf{b}_{\mu_{\mathbf{x}_i'}}^{L'} \tag{20}$$

$$\sigma_{\mathbf{x}_i'} = \mathbf{W}_{\sigma_{\mathbf{x}_i'}}^{L}\mathbf{h}^{L} + \mathbf{b}_{\sigma_{\mathbf{x}_i'}}^{L'} \tag{21}$$

where the L' is the layer in front of the layer which generate the mean vector and the variance matrix.

These two parameters can be used to build the SPE control chart, the equation of SPE and the control limit are represented as the following Eqs. (22) and (23).

$$SPE = \left(\mathbf{x}' - \mu_{\mathbf{x}'}\right)\left(\mathbf{x}' - \mu_{\mathbf{x}'}\right)^T \tag{22}$$

$$CL_{SPE} = \frac{\mathbf{v}}{2\mathbf{m}}\chi^2_{(2\mathbf{m}^2/\mathbf{v},\alpha)} \tag{23}$$

where $\mathbf{m}$ and $\mathbf{v}$ are the mean vector and covariance matrix of SPE chart.

Different from the traditional VAE-based algorithm and VAE based algorithm that merely reduce the dimensions of all input variables, the PLS-VAE based algorithm utilizes the characteristic of the PLS model to produce latent variables which contain the information between independent and dependent variables. Then we input these latent variables into the VAE model, acquire some new latent variables by the encoder, and use them to calculate the T^2 and SPE. In this way, these indices that contain more information perform more suitable on industrial process monitoring, and the case study in Section 4 will verify the superiority of this algorithm.

The main step of the calculation of T^2 and SPE is represented in Algorithm 2.

3 Simulation

3.1 Simulation Preparations

This simulation is built to examine the superior of the PLS-VAE algorithm by comparing it to other methods based on latent variables. In this case, we choose the VAE, PLS, AE methods and test their performance under different conditions. We use several kinds of data distributions

as inputs included the normal and non-normal database. The non-normal database included the Chi^2 distribution representing the skewed situation, the t-distribution that denotes the symmetric process and the gamma distributions which stands for the kurtosis industrial process. We use the ratio of the actual Type I error rate and the expected Type I error rate to be a criterion to assess the performance of these monitoring methods. In these four distributions, we generated 1000 data which contains 900 in-control data and 100 out-control data, and every data is composed of 100 and 300 variables, respectively. To be closer to the actual industrial process, we generate a column that is a linear combination of all variables. This column is used to simulate the output variable which is affected by input variables.

We build a VAE model which is consists of seven fully connected hidden layers. The architecture of the VAE model included three hidden layers both in the encoder and decoder and one hidden layer in the middle, which is built for the latent variables.

The complete architecture is shown in Fig. 2. We use the tanh function as the activation function. To make a comparison, we build four models which are PLS based model, AE based model, VAE based model and PLS-VAE based model. To be equal, we use the same hyperparameters of the VAE model including the number of layers, the quantity of nodes and the activation function. We choose the PCA method to ensure the amount of the extracted variables in these four models can bear the weight of the vast majority of dimensions of the original data set. And the reason we choose the PCA is that the algorithm can easily obtain the ratio of each extracted component in the whole variance. As is shown in Table 1, the first 25 principal components account for nearly 90 percent of the entire data in normal distribution, which means that we can use 25 principal components to substitute all the data. Table 1 shows the ratio of the first 25 components in each distribution.

Table 1: Numbers of the hidden feature and its ratio to the whole data set which is obtained by PCA

Input distribution	Numbers of hidden variables	
	p_{100}	p_{300}
Normal distribution	25(91.3%)	25(92.6%)
Chi^2 distribution	25(88.6%)	25(87.5%)
t-distribution	25(90.2%)	25(91.7%)
Gamma distribution	25(92.6%)	25(90.3%)

As for the hyperparameters in VAE, we use 0.001 as the learning rate, which is commonly used in the deep learning field. The loss function is optimized by Adam optimizer with 0.001 learning rate. The mean squared error is assigned as the standard to weigh up the loss function to get minimized by training.

The VAE model has a function that can transform the non-normal distribution to normal distribution, which can improve the performance of the T^2 and SPE control chart efficiently. To verify this function, we use Royston's multivariate normality test to calculate a parameter named p-value of these distributions. When the parameter p-value is larger than 0.05 which is often adopted in daily use, the data can be regarded as following the normal distribution. In the original database we created, taking Gaussian distribution as an example, the equation used for producing

the data is shown as the following Eqs. (24) and (25).

$$\begin{cases} \mathbf{x_1} = \mathbf{s_1} + \mathbf{u_1} \\ \mathbf{x_2} = \mathbf{w_1 s_1} + \mathbf{u_2} \\ \mathbf{x_3} = \mathbf{w_2 s_2} + \mathbf{u_3} \\ \vdots \\ \mathbf{x_n} = \mathbf{w_{n-1} s_{n-1}} + \mathbf{u_n} \end{cases} \tag{24}$$

$$\mathbf{y} = \sum_{i=1}^{n} \mathbf{v_i x_i} \tag{25}$$

where $\mathbf{s_i}$ means basic variable and follows $\mathcal{N}(0,1)$ distribution, $\mathbf{x_i}$ represents the process variable, $\mathbf{u_i}$ denotes the system noise, $\mathbf{w_i}$ is an integer matrix randomly sampled from 0 to 10, n means the number of process variable, $\mathbf{y}$ represents the output variable, $\mathbf{v_i}$ is the weighted matrix of $\mathbf{x_i}$ which is generated randomly and has been normalized so $\sum_{i=1}^{n} \mathbf{v_i} = 1$.

The specific production process is as follows:

(1) We generate 900 samples of 100 process variables $(\mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_{100}})$ to simulate the normal circumstance.
(2) We introduce anomalies at the 901st sample point by making $\mathbf{s}_m$ follows $\mathcal{N}(5,1)$ instead of $\mathcal{N}(0,1)$ to simulate the fault state. The number m can be chosen freely in 100. Its purpose is to make $\mathbf{x_m}$ has a significant change and can be detected by the monitoring algorithm.
(3) The first 900 samples will be combined with these 100 samples as a complete dataset.

We input the original database to PLS, AE, VAE, PLS-VAE methods and test the extracted feature to verify whether it has this function. The result is shown in Table 2. According to this table, only when the input data follows the normal distribution, the extracted features also follow the normal distribution in all methods. On the contrary, the latent features extracted in the VAE architecture follow the normal distribution.

Table 2: p-values of Royston's multivariate normality test (Type I error rate $\alpha = 0.05$)

Input distribution	Data base	Methods			
		PLS	AE	VAE	PLS-VAE
Normal distribution	p_{100}	0.58	0.81	0.74	0.85
Normal distribution	p_{300}	0.19	0.56	0.81	0.83
Chi^2 distribution	p_{100}	0	0	0.80	0.57
Chi^2 distribution	p_{300}	0	0	0.74	0.85
t-distribution	p_{100}	0	0	0.30	0.46
t-distribution	p_{300}	0	0	0.27	0.72
Gamma distribution	p_{100}	0	0	0.45	0.46
Gamma distribution	p_{300}	0	0	0.82	0.43

3.2 Simulation Analysis

We compare the performance of these four different methods by the ratio of the actual Type I error rate and the expected Type I error rate. We use different values of Type I error rate

from the range 0.02 to 0.2. The control chart result is shown in Fig. 3. In these pictures, the line generated by these ratio points is treated as better when they get closer to the 45 degree line because this line means the actual model obtain the same value as the expected one. These pictures confirm that the control charts which have VAE structure get better performance than other control charts because the expected rate it generates is more similar to their corresponding actual rate, which means its curved line is closer to the 45 degree line. Some lines are overlapped because some different methods get the same value. In the non-normal case, we find that the T^2 index of gamma distribution has a lower rate than others because that the data represents in a symmetric way is more suitable for the confidence interval of T^2 statistics.

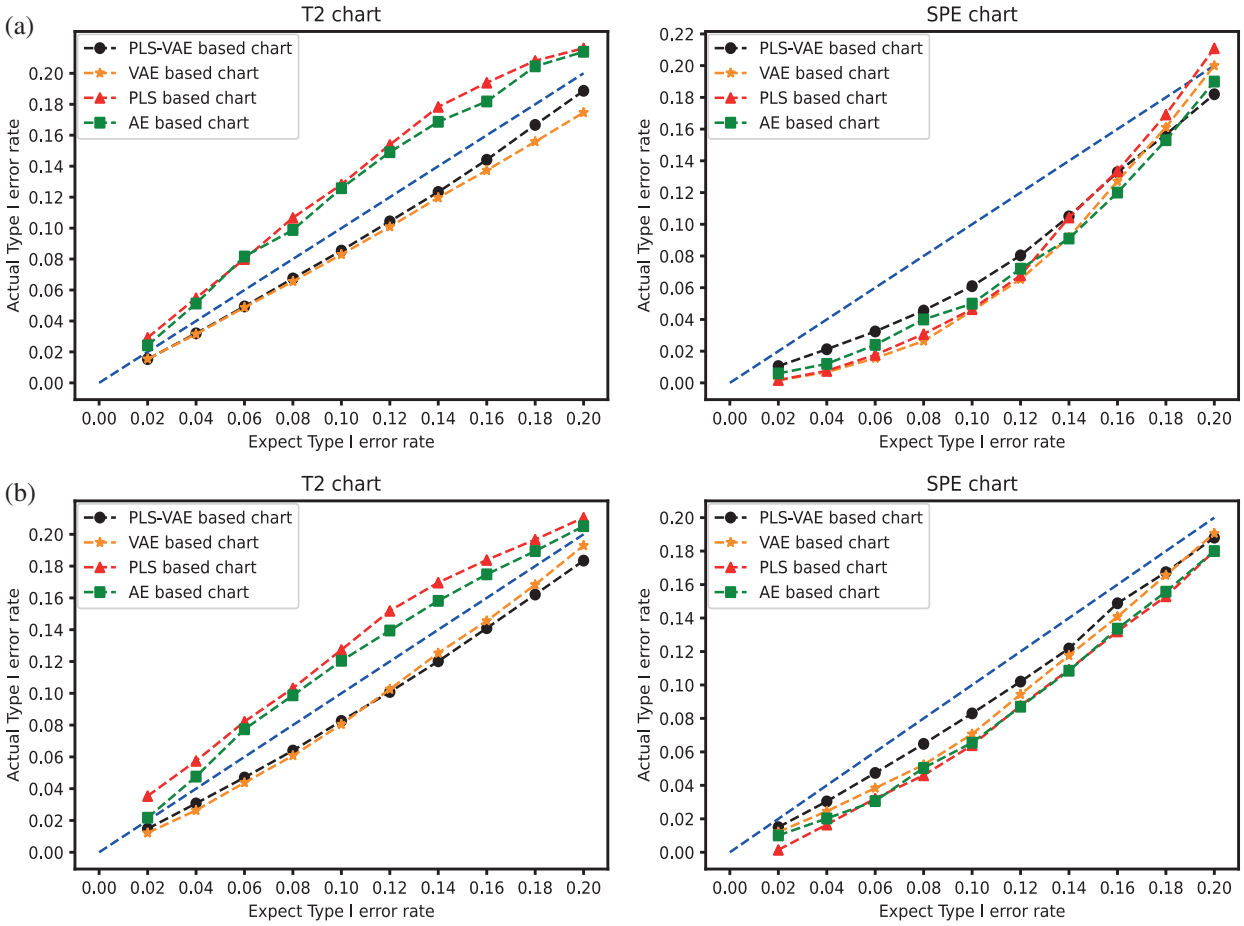


Figure 3: (Continued)

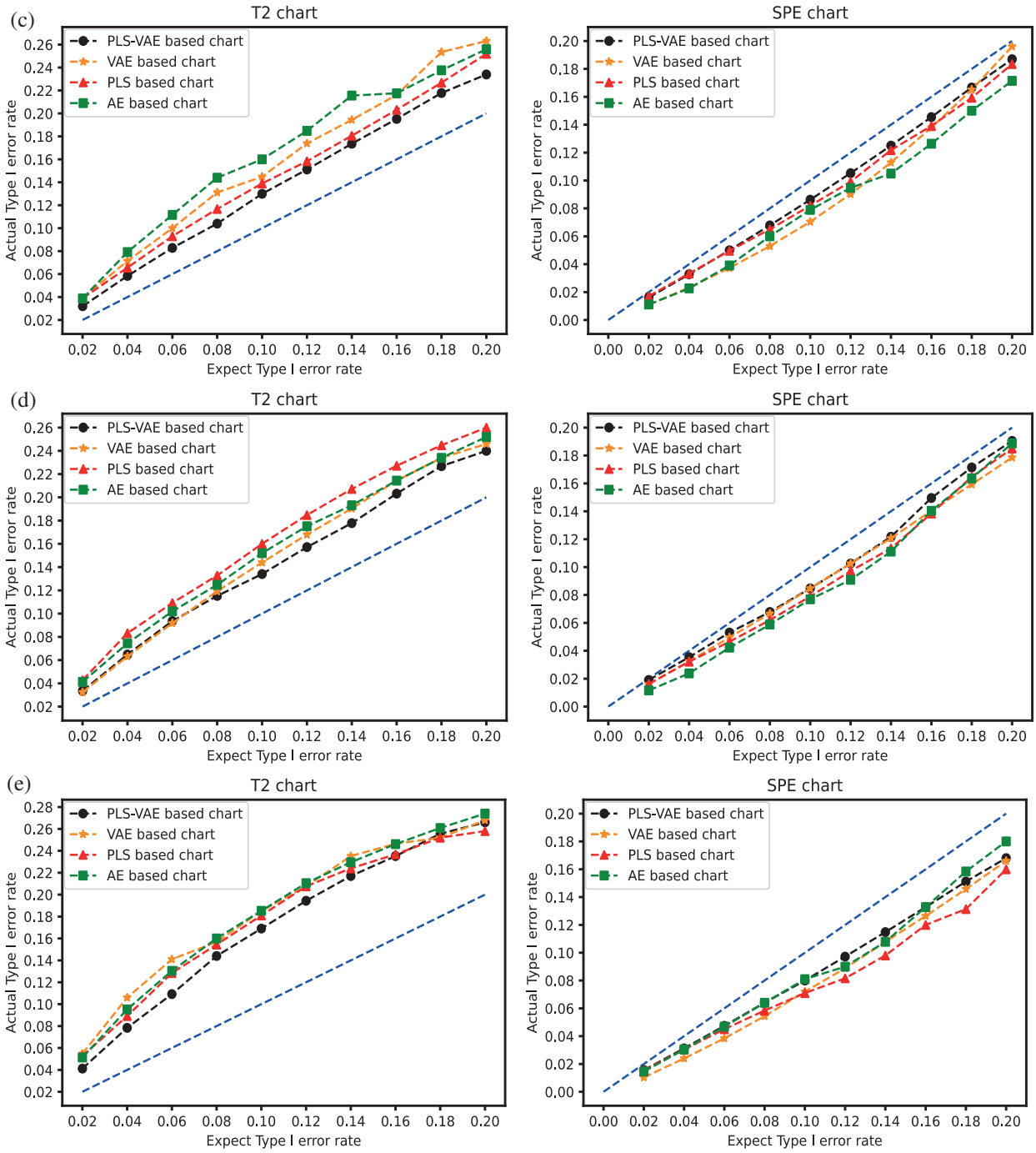


Figure 3: (Continued)

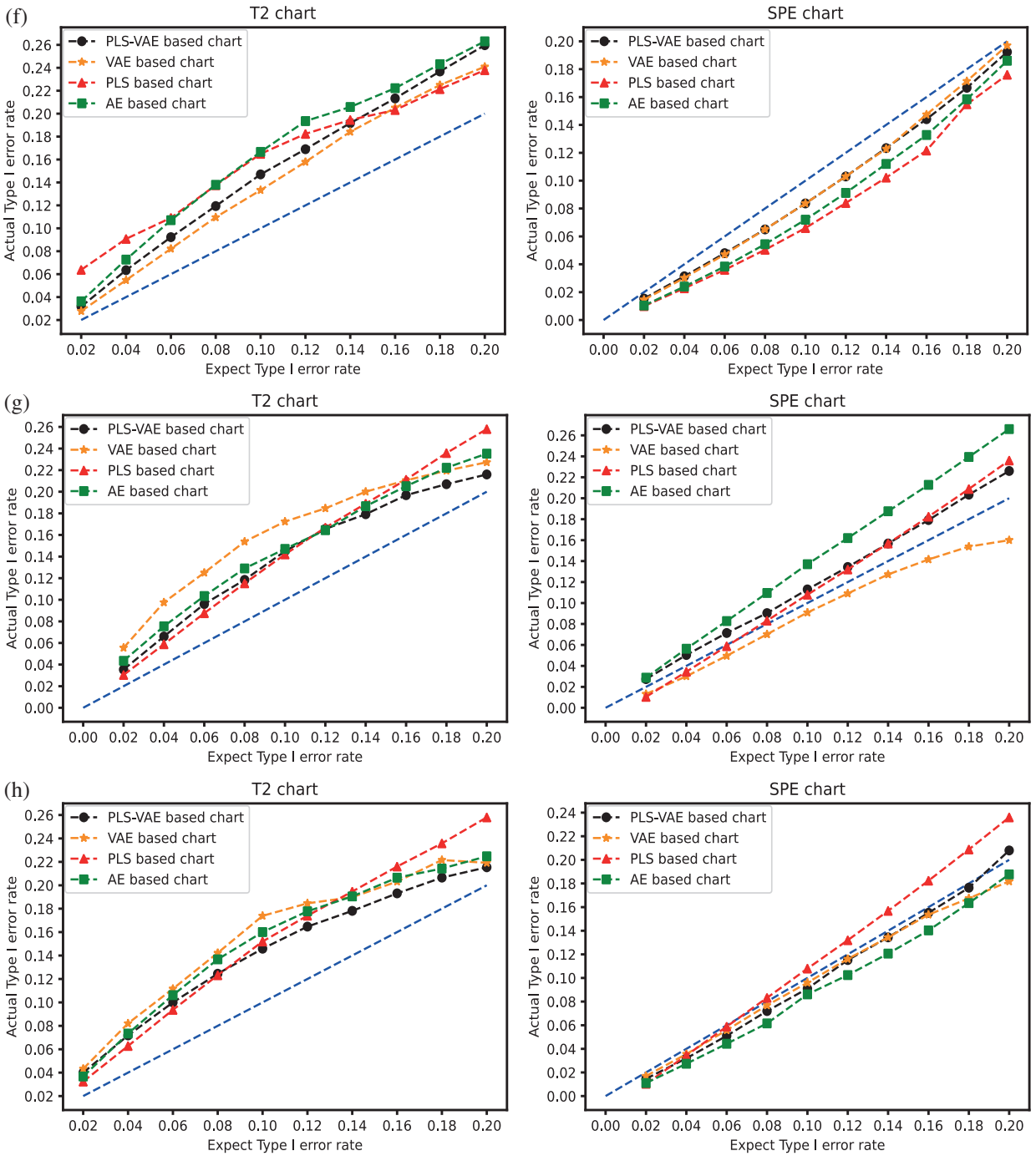


Figure 3: The T^2 and SPE control charts of different input distribution: (a) N_{100} (b) N_{300} (c) C_{100} (d) C_{300} (e) G_{100} (f) G_{300} (g) T_{100} (h) T_{300}

4 Industrial Case Analysis

4.1 Description of Trimethylchlorosilane Purification Process

The separation and purification process and system of Trimethylchlorosilane mixed monomer is composed of several processes included distillation separation, extraction, cyclic reaction. At present, the organic silicon monomer separation process which is commonly used is the direct sequence. In this sequence, the crude monomer sends through the high-removal tower and low-removal tower firstly, then gets into the binary tower to detach monomethyltrichlorosilane and dimethyldichlorosilane. The product collected on the top of the low-removal tower has been extracted from the low boiler component in the light detachment tower and the trimethylchlorosilane component in the trimethylchlorosilane tower. The high boiler component of the material in the tower kettle of the De-high tower is separated through the high boiler tower. In this process, the azeotrope component collected from the azeotrope tower is not easy to solve. The material in both the top of the high boiler tower and the tower kettle of trimethylchlorosilane tower need to get back in the former system and separate monomethyltrichlorosilane and dimethyldichlorosilane product from it. Because of the high standard to the purity of product, any tiny fluctuations of the process may make the products disqualified.

Our study focuses on the process links which monitoring object is the secondary cooling material liquid phase outlet. This link plays an integral part in the Trimethylchlorosilane purification process because its production determines the degree of the purity of the final production. The process which is directly relevant to this link is demonstrated in Fig. 4. The facility of this process link is composed of a reboiler which is used to supply the energy to the distillation tower, a distillation tower which is used to distil some kinds of products that we need from the top of the tower, a condenser which is using for condensing the input material and a reflux accumulator which refluxes to the distillation tower by a bump.

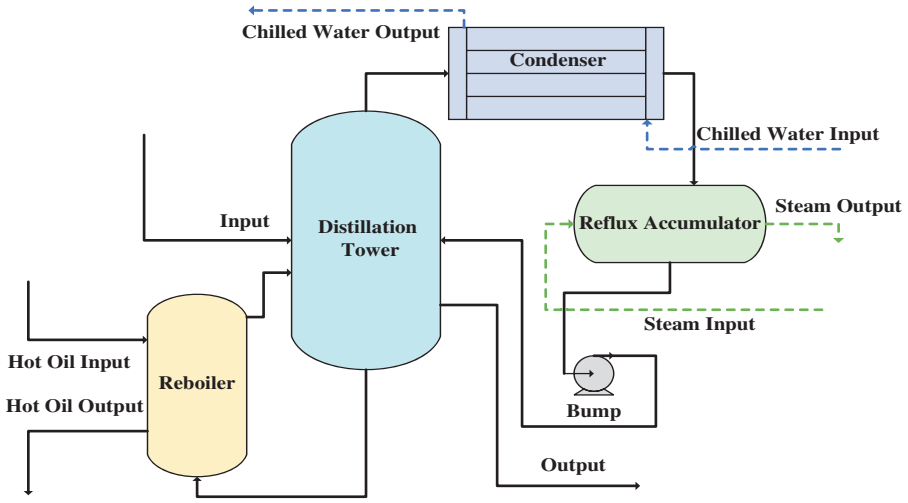


Figure 4: The process flow of the Trimethylchlorosilane purification

4.2 Analysis of the Industrial Process

In this case, we obtain 1000 observed data composed of 18 variables from the Trimethylchlorosilane purification process. The physical meaning of these 18 variables is shown in

Table 3. The first seventeen variables are treated as the independent variables, while the last one is the dependent variable of the Trimethylchlorosilane purification process. In this process, the input of the proposed method is these 1000 observed data composed of 18 variables, and the output is the control charts shown in Fig. 5.

Table 3: The physical meaning of these 18 variables

Serial number	Physical meaning	Unit
1	Second cold quench water inlet	T/H
2	Ejection density	kg/m^3
3	Hot oil inlet	T/H
4	First cold circulating water inlet flow	T/H
5	Second cold quench water inlet	T/H
6	Hot oil supplement	T/H
7	Side outflow	T/H
8	Eject outflow	kg/h
9	M3 purification tower upper part	MPA
10	Three Pagoda split adjustment	MPA
11	Hot oil temperature difference	°C
12	Bottom PCT	°C
13	Upper PCT	°C
14	Hot oil import	°C
15	Tower kettle liquid level	%
16	Reflux tank level	%
17	Product cooler temperature	°C
18	First cold material liquid phase outlet	°C

To determine the ultimate model establish methods, we compare the monitoring control chart of PLS-, AE-, VAE- and PLS-VAE-based charts and the Type I error rate $\alpha = 0.05$. The multivariable control chart based on these methods is shown in Figs. 5a–5d, respectively. In this study, we choose the T^2 and SPE control charts to describe the industrial process. The monitoring thresholds in Figs. 5a–5d are calculated by Eqs. (2) and (4). The parameters k and n in Eq. (2) are the number of latent variables and input data, respectively. The number of latent variables is determined by the PCA algorithm, and these variables contain 99.3 percent of the original information. It can be explicitly demonstrated that the methods which have the VAE structure have a lower false alarm rate, which means it may be more suitable for this kind of industrial process monitoring. Compared to the VAE control chart, the PLS-VAE method performs better due to the relationship between the dependent and independent variables.

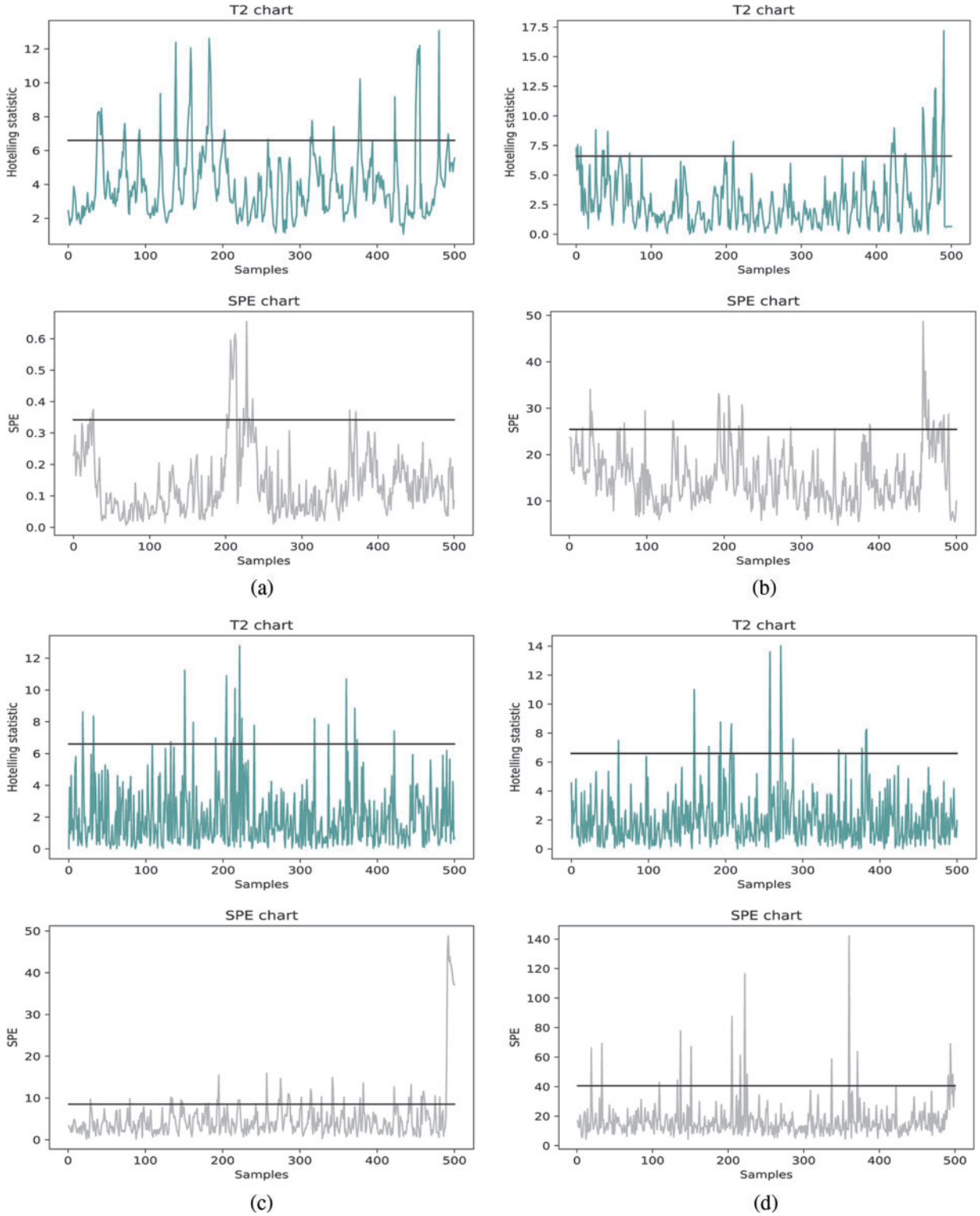
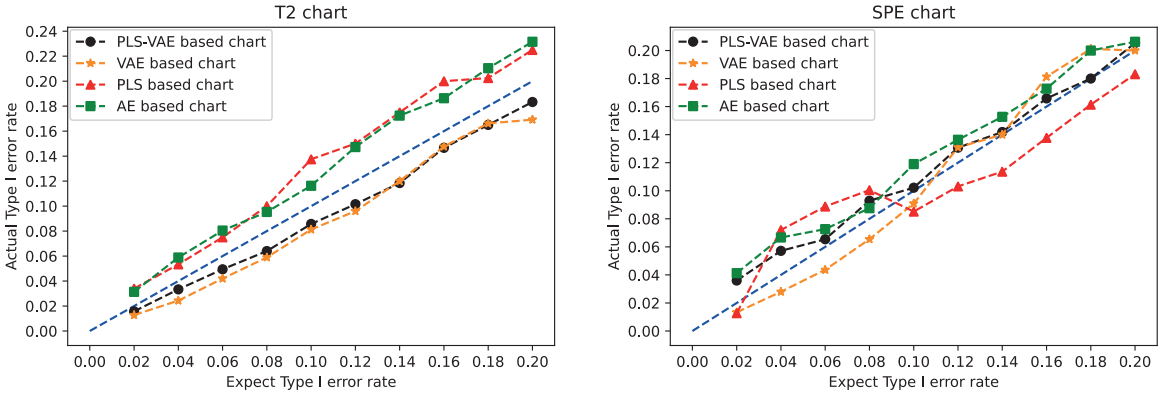


Figure 5: The control charts of different monitoring methods: (a) PLS based chart (b) AE based chart (c) VAE based chart (d) PLS-VAE based chart

To test the accuracy of the model we established, we input the test data set into the model training by the training data set and record the process quality indexes. Then, we input the test data set into the model training by itself and record the quality indexes. The comparative standard of the T^2 and SPE control chart is the actual false alarm rates and the expected false rate of the PLS-, AE-, VAE- and PLS-VAE-based charts. The actual rate is calculated from the former model, while the expected rate is obtained from the latter one. The ratio between these two rates is shown in Fig. 6a, which is represented the performance of each method. The 45 degree line means the actual rate is equal to the expected rate, which is the best circumstance, and the line which is closest to the 45 degree line performs better than others, especially when the value of the range is less than 0.15. And the value we usually choose is 0.05, thus we can determine that the PLS-VAE is better. Fig. 6b shows the ratio between the actual false rate and expected false rate of the SPE statistics. The criterion standard is also the distance to 45 degree line. And the resulting figure shows the proposed method is closer to the standard line, which means it performs better than other methods.

The model fit in Fig. 6 does not look particularly clear. In order to make it more clarity, the RMSE and R^2 of Fig. 6 are shown in Tables 4 and 5. In this tables, we can find that the RMSE and R^2 of PLS-VAE perform better than others.



(a) (b)

Figure 6: Comparison between these methods by the ratio of the actual Type I error rate and the expected Type I error rate: (a) T^2 control chart (b) SPE control chart

Table 4: The RMSE and R^2 of Fig. 6a

	PLS	AE	VAE	PLS-VAE
RMSE	0.0085	0.0076	0.0061	0.0046
R^2	0.7799	0.8239	0.8884	0.9357

Table 5: The RMSE and R^2 of Fig. 6b

	PLS	AE	VAE	PLS-VAE
RMSE	0.0053	0.0068	0.0042	0.0031
R^2	0.9159	0.8589	0.9463	0.9718

5 Conclusion

In this paper, we propose applying the PLS-VAE based control chart to monitor the multi-variables process when the input data is not following the Gaussian distribution and has a relationship between the dependent variables and independent variables. Compared to the AE model, VAE can extract the latent variable which is forced to follow the Gaussian distribution by decreasing the KL divergence between the latent distribution and the normal distribution $\mathcal{N}(0;1)$. And the PLS methods can utilize the information in the relationship between the input and output data. Then we use the T^2 and SPE statistics indices to monitor the process and the control chart based on the PLS, SAE, VAE, PLS-VAE to make a test on the simulation and the real industrial data, which is obtained from the Trimethylchlorosilane purification process. The result of the simulation and case study improve the superiority of performance of the PLS-VAE based control chart.

Our next workshop will focus on the integration between the fields of deep learning, such as VAE and some other traditional control methods. The target fields may be the multiple model process and some other process that is suitable for data-based control methods. We will also try to bring its generative ability into process monitoring and fault detection especially when the number of data is obtained scarcely and costly.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Ge, Z., Chen, J. (2015). Plant-wide industrial process monitoring: A distributed modeling framework. *IEEE Transactions on Industrial Informatics, 12(1),* 310–321. DOI 10.1109/TII.2015.2509247.
2. Zhou, K., Liu, T., Zhou, L. (2015). Industry 4.0: Towards future industrial opportunities and challenges. *2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, pp. 2147–2152. Zhangjiajie, China.
3. Tidriri, K., Chatti, N., Verron, S. (2016). Bridging data-driven and model-based approaches for process fault diagnosis and health monitoring: A review of researches and future challenges. *Annual Reviews in Control, 42(8),* 63–81. DOI 10.1016/j.arcontrol.2016.09.008.
4. Ge, Z., Liu, Y. (2018). Analytic hierarchy process based fuzzy decision fusion system for model prioritization and process monitoring application. *IEEE Transactions on Industrial Informatics, 15(1),* 357–365. DOI 10.1109/TII.2018.2836153.
5. Jiang, Q., Yan, X., Huang, B. (2019). Review and perspectives of data-driven distributed monitoring for industrial plant-wide processes. *Industrial & Engineering Chemistry Research, 58(29),* 12899–12912. DOI 10.1021/acs.iecr.9b02391.

6. Gupta, M., Kaplan, H. C. (2017). Using statistical process control to drive improvement in neonatal care: A practical introduction to control charts. *Clinics in Perinatology, 44(3),* 627–644. DOI 10.1016/j.clp.2017.05.011.
7. Ji, H., He, X., Shang, J. (2017). Incipient fault detection with smoothing techniques in statistical process monitoring. *Control Engineering Practice, 62(2),* 11–21. DOI 10.1016/j.conengprac.2017.03.001.
8. Wang, Y., Si, Y., Huang, B. (2018). Survey on the theoretical research and engineering applications of multivariate statistics process monitoring algorithms: 2008–2017. *The Canadian Journal of Chemical Engineering, 96(10),* 2073–2085. DOI 10.1002/cjce.23249.
9. He, Q. P., Wang, J. (2018). Statistical process monitoring as a big data analytic tool for smart manufacturing. *Journal of Process Control, 67,* 35–43. DOI 10.1016/j.jprocont.2017.06.012.
10. Ahsan, M., Mashuri, M., Lee, M. H. (2020). Robust adaptive multivariate Hotelling's T^2 control chart based on kernel density estimation for intrusion detection system. *Expert Systems with Applications, 145(15),* 113105. DOI 10.1016/j.eswa.2019.113105.
11. Draisma, J., Horobeţ, E., Ottaviani, G. (2016). The Euclidean distance degree of an algebraic variety. *Foundations of Computational Mathematics, 16(1),* 99–149. DOI 10.1007/s10208-014-9240-x.
12. Mostajeran, A., Iranpanah, N., Noorossana, R. (2018). An explanatory study on the non-parametric multivariate T^2 control chart. *Journal of Modern Applied Statistical Methods, 17(1),* 12. DOI 10.22237/jmasm/1529418622.
13. Dong, Y., Qin, S. J. (2018). A novel dynamic algorithm for dynamic data modeling and process monitoring. *Journal of Process Control, 67(1),* 1–11. DOI 10.1016/j.jprocont.2017.05.002.
14. Wen, L., Li, X., Gao, L. (2017). A new convolutional neural network-based data-driven fault diagnosis method. *IEEE Transactions on Industrial Electronics, 65(7),* 5990–5998. DOI 10.1109/TIE.2017.2774777.
15. Ge, Z. Q. (2017). Review on data-driven modeling and monitoring for plant-wide industrial processes. *Chemometrics and Intelligent Laboratory Systems, 171(2),* 16–25. DOI 10.1016/j.chemolab.2017.09.021.
16. Dijkstra, T. K., Henseler, J. (2015). Consistent partial least squares path modeling. *MIS Quarterly, 39(2),* 297–316. DOI 10.25300/MISQ/2015/39.2.02.
17. Tong, C., Lan, T., Yu, H. (2019). Distributed partial least squares based residual generation for statistical process monitoring. *Journal of Process Control, 75(2017),* 77–85. DOI 10.1016/j.jprocont.2019.01.005.
18. Harrou, F., Nounou, M. N., Nounou, H. N. (2015). PLS-based EWM fault detection strategy for process monitoring. *Journal of Loss Prevention in the Process Industries, 36(1),* 108–119. DOI 10.1016/j.jlp.2015.05.017.
19. Muradore, R., Fiorini, P. (2012). A PLS-based statistical approach for fault detection and isolation of robotic manipulators. *IEEE Transactions on Industrial Electronics, 59(8),* 3167–3175. DOI 10.1109/TIE.2011.2167110.
20. Mehmood, T. (2016). Hotelling T^2 based variable selection in partial least squares regression. *Chemometrics and Intelligent Laboratory Systems, 154(3),* 23–28. DOI 10.1016/j.chemolab.2016.03.001.
21. Tôrres, A. R., Junior, S. G. (2015). Multivariate control charts for monitoring captopril stability. *Microchemical Journal, 118,* 259–265. DOI 10.1016/j.microc.2014.07.017.
22. Chen, J., Liu, K. C. (2002). On-line batch process monitoring using dynamic PCA and dynamic PLS models. *Chemical Engineering Science, 57(1),* 63–75. DOI 10.1016/S0009-2509(01)00366-9.
23. Yin, S., Li, X., Gao, H. (2015). Data-based techniques focused on modern industry: An overview. *IEEE Transactions on Industrial Electronics, 62(1),* 657–667. DOI 10.1109/TIE.2014.2308133.
24. Sisinni, E., Saifullah, A., Han, S. (2018). Industrial Internet of Things: Challenges, opportunities, and directions. *IEEE Transactions on Industrial Informatics, 14(11),* 4724–4734. DOI 10.1109/TII.2018.2852491.
25. Lund, D., MacGillivray, C., Turner, V. (2014). Worldwide and regional Internet of Things (IOT) 2014–2020 forecast: A virtuous circle of proven value and demand. *Technical Report Series, 1(1*), 9. International Data Corporation (IDC).
26. Zhao, R., Yan, R., Chen, Z., Mao, K., Wang, P. et al. (2019). Deep learning and its applications to machine health monitoring. *Mechanical Systems and Signal Processing, 115(1),* 213–237. DOI 10.1016/j.ymssp.2018.05.050.

27. Suh, S., Chae, D. H., Kang, H. G. (2016). Echo-state conditional variational autoencoder for anomaly detection. *International Joint Conference on Neural Networks (IJCNN)*, pp. 1015–1022. Vancouver, BC, Canada.
28. Lee, S., Kwak, M., Tsui, K. L. (2019). Process monitoring using variational autoencoder for high-dimensional nonlinear processes. *Engineering Applications of Artificial Intelligence, 83(7),* 13–27. DOI 10.1016/j.engappai.2019.04.013.
29. Meng, Q., Catchpoole, D., Skillicom, D. (2017). Relational autoencoder for feature extraction. *International Joint Conference on Neural Networks (IJCNN)*, pp. 364–371. Anchorage, AK, USA.
30. Liou, C. Y., Cheng, W. C., Liou, J. W. (2014). Autoencoder for words. *Neurocomputing, 139(5210),* 84–96. DOI 10.1016/j.neucom.2013.09.055.
31. Xu, W., Tan, Y. (2019). Semisupervised text classification by variational autoencoder. *IEEE Transactions on Neural Networks and Learning Systems, 31(1),* 295–308. DOI 10.1109/TNNLS.2019.2900734.
32. Zhang, Z., Jiang, T., Zhan, C. (2019). Gaussian feature learning based on variational autoencoder for improving nonlinear process monitoring. *Journal of Process Control, 75(2),* 136–155. DOI 10.1016/j.jprocont.2019.01.008.
33. Xie, R., Hao, K., Chen, L., Huang, B. (2020). Supervised variational autoencoders for soft sensor modeling with missing data. *IEEE Transactions on Industrial Informatics, 16(4),* 2820–2828. DOI 10.1109/TII.2019.2951622.
34. Hemmer, M., Klausen, A., Khang, H. V. (2020). Health indicator for low-speed axial bearings using variational autoencoders. *IEEE Access, 8,* 35842–35852. DOI 10.1109/ACCESS.2020.2974942.
35. Yan, X., She, D., Xu, Y. (2021). Deep regularized variational autoencoder for intelligent fault diagnosis of rotor-bearing system within entire life-cycle process. *Knowledge-Based Systems, 226(6),* 107142. DOI 10.1016/j.knosys.2021.107142.
36. Zhang, K., Tang, B., Qin, Y. (2019). Fault diagnosis of planetary gearbox using a novel semi-supervised method of multiple association layers networks. *Mechanical Systems and Signal Processing, 131(5786),* 243–260. DOI 10.1016/j.ymssp.2019.05.049.
37. Cheng, F., He, Q. P., Zhao, J. (2019). A novel process monitoring approach based on variational recurrent autoencoder. *Computers & Chemical Engineering, 129(17),* 106515. DOI 10.1016/j.compchemeng.2019.106515.
38. Zhu, J., Shi, H., Song, B. (2020). Information concentrated variational autoencoder for quality-related nonlinear process monitoring. *Journal of Process Control, 94,* 12–25. DOI 10.1016/j.jprocont.2020.08.002.