# Nonlinear Process Monitoring Using Improved Kernel Principal Component Analysis

Chihang Wei[1], Junghui Chen[2], Zhihuan Song[1]

1. State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027
E-mail: chhwei@zju.edu.cn; songzhihuan@zju.edu.cn

2. Department of Chemical Engineering, Chung-Yuan Christian University, Taoyuan 32023
E-mail: jason@wavenet.cycu.edu.tw

**Abstract:** Kernel principal component analysis (KPCA) has become a popular technique for process monitoring in recent years. However, the performance largely depends on kernel function, yet methods to choose an appropriate kernel function among infinite ones have only been sporadically touched in the research literatures. In this paper, a novel methodology to learn a data-dependent kernel function automatically from specific input data is proposed and the improved kernel principal component analysis is obtained through using the data-dependent kernel function in traditional KPCA. The learning procedure includes two parts: learning a kernel matrix and approximating a kernel function. The kernel matrix is learned via a manifold learning method named maximum variance unfolding (MVU) which considers underlying manifold structure to ensure that principal components are linear in kernel space. Then, a kernel function is approximated via generalized Nyström formula. The effectiveness of the improved KPCA model is confirmed by a numerical simulation and the Tennessee Eastman (TE) process benchmark.

**Key Words:** Nonlinear Process Monitoring, Fault Detection, Manifold Learning, Kernel Function Approximation

## 1 INTRODUCTION

In manufacturing industry, efficient process monitoring is essentially significant due to the increasing demands on economy, safety and environmental protection. However, due to the underlying mass, energy balances and other operational restrictions, high dimensionality process data actually lies in a low dimensionality manifold embedded in the input space. Thus, multivariate statistical process monitoring (MSPM) has been widely used which firstly performs dimensionality reductions and then monitors in the low dimensional structure. [1]

Traditionally, MSPM employs principal component analysis (PCA) for discovering the relationship between process variables [1]. However, for nonlinear cases, performance of PCA-based process monitoring degenerates due to its assumption that the process data is linear [2]. Kernel PCA (KPCA) was proposed to generalize PCA to nonlinear cases [2][3]. However, the performance largely depends on the kernel function, yet methods to choose an appropriate kernel function have only been sporadically touched upon in the research literatures [4].

Actually, the assumption that guarantees the effectiveness of KPCA is that the underlying manifold structure becomes linear in the kernel space which is defined by the kernel function. However, standard kernel functions such as polynomial and Gaussian, do not consider the underlying manifold structure, which cannot accurately support the assumption. Based on this thoughtfulness, a novel methodology to learn a data-dependent kernel function from specific input data is proposed in this paper. The learning procedure includes two parts: learning a kernel matrix and approximating a kernel function. The kernel matrix is learned via a manifold learning method named maximum variance unfolding (MVU) [5] which considers underlying manifold structure to ensure that data is linear in kernel space. However, MVU can only conduct dimensionality reduction on training samples. Recent years, some papers focus on this problem. In [6], Shao attempted to introduce a multivariate linear regression model between input and output space of MVU, but this is problematic since the true regression model is usually nonlinear. In [7], Liu developed a Gaussian process model between input and output space. The performance of monitoring is iffy since this is a probability model which $T^2$ and squared prediction error (SPE) statistics aren't suitable for. In this paper, an effective and flexible methodology is introduced to approximate a kernel function via generalized Nyström formula. Meanwhile, algorithm to determinate parameters of the learned kernel function is creatively proposed.

The organization of this paper is as follows. In Section 2, KPCA-based process monitoring method is briefly outlined and the limitation KPCA is illustrated by a numerical simulation. The learning algorithm of kernel matrix is presented in Section 3. Approximation of a data-dependent kernel function is expounded in Section 4. In Section 5, simulations on a simple nonlinear system and the Tennessee Eastman (TE) process benchmark are performed to demonstrate the performance of improved KPCA-based monitoring. Finally, a conclusion is drawn in Section 6.

This work is supported by National Nature Science Foundation of China under Grant 61573308 and Research Fund for the Doctoral Program of Higher Education under Grant 20130101110138.

## 2 KPCA-BASED PROCESS MONITORING

### 2.1 KPCA

Kernel PCA (KPCA) was proposed to generalize PCA to nonlinear cases by using a kernel function to nonlinearly map original data to a higher or even infinite dimensional kernel space $\Psi^Q$ and performing PCA there.

Specifically, suppose $\mathbf{x}_1,\ldots,\mathbf{x}_N \in \mathbb{R}^M$ are training samples, then they are mapped to $\mathbf{\Phi}(\mathbf{x}_1),\ldots,\mathbf{\Phi}(\mathbf{x}_N) \in \Psi^Q$ by a nonlinear mapping $\mathbf{\Phi}(\cdot): \mathbb{R}^M \to \Psi^Q$, where $N, M, Q$ is quantity of sample, quantity of process variables and dimension of kernel space, respectively. Then, PCA is performed in $\Psi^Q$ by an eigenvalue problem below:

$$\lambda \mathbf{v} = \mathbf{C}_\Psi \mathbf{v} \tag{1}$$

where $\mathbf{C}_\Psi = \frac{1}{N}\sum_{n=1}^{N}[\mathbf{\Phi}(\mathbf{x}_n)][\mathbf{\Phi}(\mathbf{x}_n)]^T$ denotes the covariance matrix, and $\lambda, \mathbf{v}$ denote eigenvalue and eigenvector of $\mathbf{C}_\Psi$. Subsequently, the following eigenvalue problem is obtained:

$$\lambda[\mathbf{\Phi}(\mathbf{x}_i)\mathbf{v}] = \frac{1}{N}\sum_{n=1}^{N}[\mathbf{\Phi}(\mathbf{x}_n)][\mathbf{\Phi}(\mathbf{x}_n)]^T\mathbf{\Phi}(\mathbf{x}_i)\mathbf{v} \tag{2}$$

Equation (2) can be formulated to Equation (3) by introducing a kernel matrix $\mathbf{K} \in \mathbb{R}^{N*N}$ with elements $\mathbf{K}_{ij} = \mathbf{\Phi}(\mathbf{x}_i)\cdot\mathbf{\Phi}(\mathbf{x}_j) = \kappa(\mathbf{x}_i,\mathbf{x}_j)$ and $\kappa(\mathbf{x}_i,\mathbf{x}_j)$ is usually called kernel function:

$$\lambda\boldsymbol{\alpha} = \frac{1}{N}\mathbf{K}\boldsymbol{\alpha} \tag{3}$$

Letting $\lambda_1 \geq \ldots \geq \lambda_N \geq 0$ denote the ordered eigenvalues of $\mathbf{K}$ with corresponding eigenvectors $\alpha_1,\ldots,\alpha_N$, the $l-th$ principal component of an input sample $\mathbf{x}_w$ is given by:

$$\begin{aligned} t_{new,l} &= \langle \mathbf{v}_l, \mathbf{\Phi}(\mathbf{x}_w)\rangle = \frac{1}{\sqrt{\lambda_l}}\sum_{n=1}^{N}\alpha_i^l\kappa(\mathbf{x}_n,\mathbf{x}_w) \\ &= \frac{1}{\sqrt{\lambda_l}}\sum_{n=1}^{N}\alpha_i^l[\mathbf{k}_w]_n \end{aligned} \tag{4}$$

where $\mathbf{k}_w$ is called kernel vector of $\mathbf{x}_w$ with elements $[\mathbf{k}_w]_n = \kappa(\mathbf{x}_w,\mathbf{x}_n)$ and $\mathbf{x}_n$ is training sample. Then the reduce-dimension result of $\mathbf{x}_w \in \mathbb{R}^M$ is denoted as $\mathbf{y}_w = [t_{w,1}, t_{w,2},\ldots,t_{w,\mathrm{D}}] \in \mathbb{R}^D$, where $D < M$.

### 2.2 Monitoring Method

In this paper, classical $T^2$ and $SPE$ statistics are employed to measure the variation of data in reduce-dimension space and residual space. The monitoring procedure includes off-line phase and on-line phase. [2]

Off-line modeling phase:

1) Collect data samples $\mathbf{x}_1,\ldots,\mathbf{x}_N$ under normal operating conditions and normalize them.

2) Calculate kernel matrix $\mathbf{K}$ using kernel function.

3) Solve the eigenvalue problem in (3).

4) Extract $D$ nonlinear principal components of $\mathbf{x}_1,\ldots,\mathbf{x}_N \in \mathbb{R}^M$ to obtain $\mathbf{y}_1,\mathbf{y}_2,\ldots,\mathbf{y}_N \in \mathbb{R}^D$.

5) Compute monitoring statistics ($T^2$ and $SPE$) of the normal operating data, and compute monitoring statistics control limits ($T^2_{\lim}$ and $SPE_{\lim}$) as following:

$$T^2 = \mathbf{y}^{\mathrm{T}}\Pi^{-1}\mathbf{y} \tag{5}$$

$$\mathrm{SPE} = \| \mathbf{\Phi}(\mathbf{x}) - \tilde{\mathbf{\Phi}}(\mathbf{x}) \|^2 = \kappa(\mathbf{x},\mathbf{x}) - \mathbf{y}\cdot\mathbf{y} \tag{6}$$

$$T^2_{\lim} = \frac{D(N-1)}{N-D}F(D,N-D,\eta) \tag{7}$$

$$SPE_{\lim} = g\cdot\chi^2_h \tag{8}$$

where $\Pi$ is covariance matrix of $\mathbf{y}$, $\tilde{\mathbf{\Phi}}(\mathbf{x}) = \sum_{d=1}^{D}\mathbf{y}_d\mathbf{v}_d$ is the reconstruction of $\mathbf{\Phi}(\mathbf{x})$, $F(D,N-D,\eta)$ is distribution with parameters $(D,N-D,\eta)$, and $\chi^2_h$ is chi square distribution satisfying $g\cdot h = mean(\mathrm{SPE})$, $2g^2\cdot h = \mathrm{var}(\mathrm{SPE})$.

Online monitoring phase:

1) Normalize a new process data $\mathbf{x}_w$.

2) Calculate the kernel vector $\mathbf{k}_w$ by $[\mathbf{k}_w]_n = \kappa(\mathbf{x}_w,\mathbf{x}_n)$.

3) Extract $D$ nonlinear principal components of $\mathbf{x}_w$ by the KPCA model developed in off-line modeling phase.

4) Compute $T^2$ and $SPE$ statistics respect to $\mathbf{x}_w$ and monitor whether they exceed corresponding control limits or not.

### 2.3 Limitation

There exists a number of standard kernel functions and the representative ones are linear kernel $\kappa(\mathbf{x}_i,\mathbf{x}_j) = \mathbf{x}_i\cdot\mathbf{x}_j$, polynomial kernel $\kappa(\mathbf{x}_i,\mathbf{x}_j) = (1+\mathbf{x}_i\cdot\mathbf{x}_j)^\beta$, $\beta \in \mathbb{N}^+$ and Gaussian RBF kernel $\kappa(\mathbf{x}_i,\mathbf{x}_j) = \exp(-\|\mathbf{x}_i-\mathbf{x}_j\|^2/\tau)$, $\tau \in \mathbb{R}^+$. As mentioned in Section 1.1, in KPCA, original data is nonlinearly mapped a higher or even infinite dimensional kernel space $\Psi^Q$ to perform PCA there. The specific choice of a kernel function implicitly determines the mapping $\mathbf{\Phi}(\cdot)$ and the kernel space $\Psi^Q$. Thus, the performance of KPCA largely depends on the choice of kernel function.

A simple nonlinear simulation is considered to vividly illustrate this problem and comparisons of the performances of KPCA based process monitoring with different kernel functions are made. The simulation with three variables and one factor [2] is

$$\begin{aligned} x_1 &= t + e_1 \\ x_2 &= t^2 - 3t + e_2 \\ x_3 &= -t^3 + 3t^2 + e_3 \end{aligned} \tag{9}$$

where $e_1, e_2, e_3$ are independent noise variables which follow Gaussian distribution $N(0, 0.01)$ and $t$ follows uniform distribution $E(0.01, 2)$. 500 training samples are generated and two testing data sets each comprising 300 samples are also generated with one of the following two fault models:

Fault 1: A step change of $x_2$ by -0.4 is introduced starting from sample 101.

Fault 2: $x_3$ is linearly increased from sample 101 by adding $0.01(j-101)$ to the $x_3$ value of each sample in its range, where $j$ is sample number.

Three previous listed common used kernel functions are used. $T^2$ and *SPE* statistics, both with 99% confidence limit, will be given to evaluate monitoring performance. The reduced dimensionality $D$ is set as there's a large gap between the $Dth$ and $(D+1)th$ eigenvalues of kernel matrix $\mathbf{K}$ and if there isn't a large gap, it is set as the sum of the biggest $D$ eigenvalue is bigger than the product of 0.99 and total sum of eigenvalues.

Table 1 and Table 2 present detection rates and respective number of kernel principal components (KPCs) of KPCA using different kernel functions. The result of improved KPCA is also presented and this will be discussed in Section 5. For polynomial and RBF kernel function, several parameters are tested respectively and $\tau = 15$ in RBF kernel function is suggested by [2]. It should be mentioned that the low detection rates of $T^2$ statistics are caused by nongaussianity of data.

Table1. Detection Rates of the Nonlinear Simulation, Part 1

| Type | | Linear | Polynomial | | | |
|---|---|---|---|---|---|---|
| Parameter | | Null | 1 | 2 | 3 | 5 |
| Normal | T2 | 0.000 | 0.000 | 0.010 | 0.040 | 0.090 |
| | SPE | 0.020 | 0.000 | 0.010 | 0.020 | 0.035 |
| Fault 1 | T2 | 0.000 | 0.000 | 0.000 | 0.035 | 0.065 |
| | SPE | 0.020 | 0.330 | 0.500 | 0.550 | 0.350 |
| Fault 2 | T2 | 0.055 | 0.000 | 0.085 | 0.110 | 0.180 |
| | SPE | 0.815 | 0.315 | 0.440 | 0.390 | 0.290 |
| KPCs | | 2 | 1 | 2 | 2 | 2 |

Table2. Detection Rates of the Nonlinear Simulation, Part 2

| Type | | RBF | | | | IKPCA |
|---|---|---|---|---|---|---|
| Parameter | | 5 | 10 | 15 | 18 | Null |
| Normal | T2 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| | SPE | 0.020 | 0.001 | 0.025 | 0.030 | 0.040 |
| Fault 1 | T2 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| | SPE | 0.170 | 0.390 | 0.590 | 0.505 | 1.000 |
| Fault 2 | T2 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| | SPE | 0.415 | 0.535 | 0.600 | 0.585 | 0.830 |
| KPCs | | 2 | 2 | 2 | 1 | 1 |

It's evident that KPCA with linear kernel doesn't perform well because it's actually a linear PCA. For KPCA with polynomial kernel, the best monitoring performance of Fault 1 and Fault 2 occurs when $\beta = 3$, but both detection rates are not satisfactory. KPCA using RBF kernel outperforms KPCA using other common used kernels, and the best monitoring performance occurs under the suggested $\tau$. It's worth noting that the number of kernel principal components should be same as number of system factor, however, only a small fraction of KPCA models successfully obtains the right number: 1.

From the above description, it can be seen that the performance of KPCA strongly depends on the form of kernel function and its parameter. An appropriate choice of kernel function is indispensable for KPCA to give reasonable low-dimensional representation of data and process monitoring performance. In fact, the effectiveness of KPCA is based on the assumption that the underlying manifold structure of data in the input space becomes linear after being mapped into the kernel space which defined by the kernel function (hence PCA can be effectively performed there). Using standard kernels such as Gaussians or polynomial, which do not consider the underlying manifold structure in input data, cannot effectively support the above assumption. Apparently, there is no general kernel function suitable to all data sets. It is desirable to "learn" a data-dependent kernel function that adapts well to specific input data.

## 3 LEARNING KERNEL MATRIX

This paper proposed a method to "learn" a data-dependent kernel function from specific input data which effectively supports the assumption that the underlying manifold structure of data in the input space becomes linear after being mapped into the kernel space. Unlike traditional ways to select a kernel function from provided ones to support the above assumption, this method includes an optimization procedure that directly aims at improving and guaranteeing linearity of data in kernel space. Without any prior knowledge and experience, this method is more flexible and thus can accurately support the assumption of KPCA. The improved KPCA is obtained through substituting traditional kernel functions for the learned kernel function. Obviously, the improved KPCA should have a better monitoring performance.

Here a methodology casted on "Semi Definite Programming" (SDP) [5] is used which is an optimization algorithm, directly aiming at finding out the "best" kernel space, whose result is kernel matrix $\mathbf{K}$. This is a manifold learning algorithm, and in perspective of manifold learning, the data manifold in input space is unfolded in the kernel space $\Psi^Q$ which is implicitly defined by $\mathbf{K}$, then the manifold is unfolded in the reduced space, too. Both $\mathbf{K}$ and kernel space $\Psi^Q$ are learnt from specific input data, which are more flexible and suitable. Specifically, a kernel matrix is a matrix whose elements are dot products of pairwise data in kernel space: $\mathbf{K}_{ij} = \mathbf{\Phi}(\mathbf{x}_i) \cdot \mathbf{\Phi}(\mathbf{x}_j)$. The following subsections will review the semi-definite programming,

including one objective function and three constraints, which are all operates on $\mathbf{K}$ and its elements.

### 3.1 Objective Function

The objective function is constructed based on a simple intuition: any "fold" between two samples on a manifold serves to decrease the Euclidean distance between them [5]. Thus, to unfold the manifold, or to say, to improve linearity in kernel space $\Psi^{Q}$, it just needs to maximize the sum of pairwise squared distances between samples in kernel space and the objective function is obtained:

$$\begin{aligned}\Gamma &= \frac{1}{2}\sum_{n_1=1}^{N}\sum_{n_2=1}^{N}\| \mathbf{\Phi}(\mathbf{x}_{n_1}) - \mathbf{\Phi}(\mathbf{x}_{n_2}) \|^2 \\ &= \frac{1}{2}\sum_{n_1=1}^{N}\sum_{n_2=1}^{N}(\mathbf{K}_{n_1n_1} + \mathbf{K}_{n_2n_2} - \mathbf{K}_{n_1n_2} - \mathbf{K}_{n_2n_1})\end{aligned} \tag{10}$$

### 3.2 Constraints

1) Isometry

However, maximizing sum of pairwise squared distances will pull the mapped samples in $\Psi^{Q}$ far apart from each other, or in other words, destroy the local structure of training samples. Thus, the local geometric structure of input data should be preserved because nearby data samples in the input space should also stay close in kernel space so that the variation in input data can be represented in the final reduced space [5]. Then the constraint can be formally stated as following:

Let a $N*N$ binary adjacency matrix $\mathbf{S}$ indicate the neighborhood relationship of training samples by setting $\mathbf{S}_{ij}=1$ if $\mathbf{x}_j$ is a $k$ nearest neighbor of $\mathbf{x}_i$, otherwise $\mathbf{S}_{ij}=0$ .Whenever $\mathbf{x}_i$ and $\mathbf{x}_j$ are $k$ -nearest neighbors ( $\mathbf{S}_{ij}=1$ ) or are common neighbor samples of another sample ( $[\mathbf{S}^T\mathbf{S}]_{ij}>0$ ), the following equation should be hold:

$$\begin{aligned}&\| \mathbf{\Phi}(\mathbf{x}_i) - \mathbf{\Phi}(\mathbf{x}_i) \|^2 = \| \mathbf{x}_i - \mathbf{x}_j \|^2 \Leftrightarrow \\ &\mathbf{K}_{ii} + \mathbf{K}_{jj} - \mathbf{K}_{ij} - \mathbf{K}_{ji} = \| \mathbf{x}_i - \mathbf{x}_j \|^2\end{aligned} \tag{11}$$

In real applications, the number of nearest neighbors $k$ is usually set be the smallest integer which is large enough to ensure the $k$ nearest neighbor graph is connected.

2) Positive semi-definiteness

As elements of kernel matrix $\mathbf{K}$ can be interpreted as dot products of pairwise data in kernel space, $\mathbf{K}$ should be positive semi-definite:

$$\mathbf{K} \geq 0 \tag{12}$$

3) Centering

For calculation convenience, mapped training samples in $\Psi^{Q}$ are mean-centered, thus $\mathbf{K}$ should also be mean-centered:

$$\sum_{n=1}^{N}\mathbf{\Phi}(\mathbf{x}_n) = 0 \Leftrightarrow \| \sum_{n=1}^{N}\mathbf{\Phi}(\mathbf{x}_n) \|^2 = \sum_{n_1}^{N}\sum_{n_2}^{N}\mathbf{K}_{n_1n_2} = 0 \tag{13}$$

Under this constraint, Equation (10) can be abbreviated as:

$$\begin{aligned}\Gamma &= \frac{1}{2}\sum_{n_1=1}^{N}\sum_{n_2=1}^{N}\| \mathbf{\Phi}(\mathbf{x}_{n_1}) - \mathbf{\Phi}(\mathbf{x}_{n_2}) \|^2 \\ &= \frac{1}{2}\sum_{n_1=1}^{N}\sum_{n_2=1}^{N}(\mathbf{K}_{n_1n_1} + \mathbf{K}_{n_2n_2} - \mathbf{K}_{n_1n_2} - \mathbf{K}_{n_2n_1}) \\ &= \frac{1}{2}\sum_{n_1=1}^{N}\sum_{n_2=1}^{N}(\mathbf{K}_{n_1n_1} + \mathbf{K}_{n_2n_2}) = Trace(\mathbf{K})\end{aligned} \tag{14}$$

This constraint synchronously ensures the eigenvalues of $\mathbf{K}$ can be interpreted as measures of variance along principal components in $\Psi^{Q}$.

### 3.3 Optimization

Combine the objective function and three constraints to define an instance of semi definite programming. Notice that all three constraints and the objective function are convex, thus the optimization result is global optimal. There are several ready-made efficient toolboxes for solving this semi-definite programming, like CSDP toolbox and SeDuMi toolbox in Matlab programming development environment. In this paper, the former one is used.

## 4 APPROXIMATING KERNEL FUNCTION

### 4.1 Limitation of MVU

In the previous section, a kernel matrix $\mathbf{K}$ is automatically learned from training samples and the $D$ -dimensional embedding (dimension reduction output) of training samples can be obtained through performing eigen-decomposition of $\mathbf{K}$ . The off-line modeling phase is conducted. However, as far as on-line phase to handle novel data, the problem of extrapolating the embedding of a manifold learned from finite samples to novel data is remained unsolved. In this paper, a methodology via generalized Nyström formula to extrapolate the embedding is used, which is firstly proposed to generalize Gaussian process covariance matrix learned through Expectation-Maximization to novel inputs [8][9].

#### 4.2 Kernel Function Approximation

According to [9], firstly, the kernel eigenfunction $g_{p,\mathrm{N}}(\mathbf{x})$ of $\mathbf{K}$ is approximated with basic functions $z(\cdot,\cdot)$ . Using generalized Nyström formula, the approximated $qth$ scaled eigenfunction of $\mathbf{K}$ is

$$g_{p,\mathrm{N}}(\mathbf{x}) = \sum_{n=1}^{N} b_{qn} z(\mathbf{x}, \mathbf{x}_n) \tag{15}$$

with weights $\mathbf{b}_q = [b_{q1}, \ldots, b_{qN}]^T = (\mathbf{Z} + \iota\mathbf{I})^{-1}\mathbf{v}_q$ , where $\mathbf{v}_q$ denotes the eigenvector of $\mathbf{K}$ . Matrix $\mathbf{Z}$ denotes gram matrix with elements $\mathbf{Z}_{ij} = z(\mathbf{x}_i, \mathbf{x}_j)$ , and $\iota\mathbf{I}$ is the regularization term introduced to stabilize inverse. Then, the corresponding approximated data-dependent kernel function is calculated as:

$$\kappa(\mathbf{x}_i,\mathbf{x}_j)=\sum_q l_q g_{q,N}(\mathbf{x}_i)g_{q,N}(\mathbf{x}_i)$$
$$=\mathbf{z}(\mathbf{x}_i)^T(\mathbf{Z}+\iota\mathbf{I})^{-1}\mathbf{K}(\mathbf{Z}+\iota\mathbf{I})^{-1}\mathbf{z}(\mathbf{x}_j) \quad (16)$$

### 4.3 Parameters Optimization

So far, the approximated data-dependent kernel function in Equation (16) is not workable, as there are still two parameters left to determine: $\iota$ and $\rho$. To select the optimal values of $\iota$ and $\rho$ for a specific learned manifold, an algorithm is creatively proposed.

From the perceptual intuition, the following parameter optimization is designed, where the square sum of differences between elements of $\mathbf{K}$ and corresponding results of approximated data-dependent kernel function is minimized:

$$\{\rho^*,\iota^*\}=\arg\min_{\rho,\iota}\sum_{n_1=1}^{N}\sum_{n_2=1}^{N}(\mathbf{K}_{n_1n_2}-\kappa(\mathbf{x}_{n_1},\mathbf{x}_{n_2}))^2 \quad (17)$$

In this case, accuracy of approximation is uniquely considered under training samples. However, due to perturbations of data caused by noise in the data acquisition or pre-processing procedures, result of Equation (17) is much likely to bring in over-fitting.

Instead, an innovative parameter optimization method is proposed in this paper. Equation (17) guarantees the highest accuracy of approximation through minimizing the difference between learned kernel matrix and approximated kernel matrix under training samples. However, as there exists perturbations in data caused by noise in the data acquisition or pre-processing procedures, not only restrict training samples, but also slightly noised training samples generated from adding slight noise should be considered in parameters optimization.

Firstly, some slightly noised training samples should be obtained which is assumed to be written as $\mathbf{x}_i^{off}$ respect to training sample $\mathbf{x}_i$. The training samples $\mathbf{x}_i$ are slightly biased by artificially adding a small noise which obeys to a Gaussian distribution. Secondly, the following parameter optimization is deduced:

$$\{\rho^*,\iota^*\}=\arg\min_{\rho,\iota}[\sum_{n_1=1}^{N}\sum_{n_2=1}^{N}(\mathbf{K}_{n_1n_2}-\kappa(\mathbf{x}_{n_1},\mathbf{x}_{n_2}))^2+$$
$$\sum_{n_1=1}^{N}\sum_{n_2=1}^{N}(\mathbf{K}_{n_1n_2}-\kappa(\mathbf{x}_{n_1}^{off},\mathbf{x}_{n_2}^{off}))^2] \quad (18)$$

Equation (18) has two parts, while minimizing $\sum_{n_1=1}^{N}\sum_{n_2=1}^{N}(\mathbf{K}_{n_1n_2}-\kappa(\mathbf{x}_{n_1},\mathbf{x}_{n_2}))^2$ serves to guarantee approximation accuracy using training samples, and minimizing $\sum_{n_1=1}^{N}\sum_{n_2=1}^{N}(\mathbf{K}_{n_1n_2}-\kappa(\mathbf{x}_{n_1}^{off},\mathbf{x}_{n_2}^{off}))^2$ serves to guarantee that the embedding of slightly noised training samples are close to the corresponding elements of $\mathbf{K}$, restraining over-fitting and ensuring authenticability of data. Thus, this bi-objective optimization focus on optimizing parameters to obtain a robust extrapolation kernel function, which can deal with noisy data within close proximity of the learned manifold.

To solve Equation (18) in real application, $\iota$ is usually fixed to a small value like 0.001 and $\rho$ is optimized. A simple dichotomy optimization is resorted which is practically tractable, since there is only one parameter.

## 5 SIMULATION

In this section, process monitoring performances of proposed improved KPCA will be demonstrated by two case studies. One is the numerical simulation, which has been introduced in Section 2 and the other one is the Tennessee Eastman (TE) process benchmark, which has been widely used for evaluating various monitoring approaches [6]. The effectiveness of the proposed method will be verified through comparison with KPCA using traditional standard kernel functions.

The number of nearest neighbors $k$, reduce-dimension $D$ and $T^2, SPE$ statistics are all set or calculated based on the idea illuminated previously in this paper.

### 5.1 Numerical Example

This numerical example has been detailed recommended in Section 2 and all training samples and testing data sets are the same. For the whole procedure of process monitoring, a kernel matrix is automatically learned from input samples, and then reduce-dimension $D$, reduce-dimension output and statistics control limits are calculated. Next, a kernel function is approximated including automatically determining $\rho$ in learned kernel function. Finally, $T^2, SPE$ statistics of novel samples can be computed. According to the rule, in this case, $k=5$, $\rho=150$, $D=1$.

The eigenvalue spectrum (not provided in this paper) shows that the first one output capture more than 99% variation information in the output space. The detection rates are already presented in Table 2.

I can be concluded from monitoring result that the improved KPCA outperforms KPCA using traditional standard kernel functions. Such results are not surprised, since improved KPCA specifically considers local and global structure of data, and learns a kernel function especially suitable for specific data.

### 5.2 TE Process Benchmark

The TE benchmark process has been widely used as a benchmark simulation of control algorithms and monitoring approaches. In this section, the proposed improved KPCA and traditional KPCA using standard kernel function will be compared based on this process. This process has 12 manipulated variables and 41 measured variables. 33 variables are chosen for monitoring as other research papers [6]. A set of 21 programmed faults are introduced to the process. The training data set consists of 600 normal operating samples, and each testing data sets for one fault consists of 960 samples in which fault is introduced at sample 161.

In improved KPCA, according to the rule, $k=5$ , $\rho=1080$ , $D=13$ . In traditional KPCA, linear kernel function, polynomial kernel function and Gaussian RBF kernel function is considered. For polynomial kernel function, several parameters are tested respectively and for Gaussian

RBF kernel function, $\tau$ is selected to be 320 as suggested by [2]. The monitoring result shows that for fault 3,5,10,14,19,21, the performance of improved KPCA is better, thus Table 3 only list detection rates of these faults for simplicity, while for other faults, the performance of improved KPCA is basically equal to the best one among all the other models. Conclusively, the improved KPCA outperforms KPCA using traditional standard kernel functions. Such results are not surprised, too.

## 6 CONCLUSION

In this paper, a methodology called improved kernel principal component (improved KPCA) is proposed. Basically, it is traditional KPCA using a data-dependent kernel function which is automatically learned from specific data. The kernel function is obtained through two main procedures: learning a kernel matrix and approximating a kernel function. Compared with traditional KPCA using standard kernel functions, improved KPCA has the following several attractive advantages.

Firstly, the local and global structure (manifold structure) of specific data is considered and the underlying manifold structure is unfolded in kernel space such that mapped data in kernel space is more likely to be linear. Hence, the nonlinear principal components of improved KPCA will be linear principal components in the kernel space which can capture the variation of data more flexibly and effectively, leading to a better performance of process monitoring. Secondly, the intrinsic dimension of data can be effectively determined such that the number of kernel principal components can be suitably set and calculation may be reduced as the number of kernel principal components of traditional KPCA using standard kernel functions is usually larger. Thirdly, this methodology requires neither prior knowledge, experience nor faulty samples to train the model, which strictly meets the real applications. Additionally, the result of this methodology is global-optimal.

Simulations on a simple nonlinear system and the TE process benchmark are performed to demonstrate that improved KPCA leads to significant improvement of process monitoring performance over traditional KPCA using traditional standard kernel functions.

## REFERENCES


[1] S. J. Qin, Statistical process monitoring: basics and beyond, Journal of Chemometrics, Vol.17, 480-502, 2003.

[2] J. M. Lee, C. K. Yoo, S. W. Choi, P. A. Vanrolleghem, I. B. Lee, Nonlinear process monitoring using kernel principal component analysis, Chemical Engineering Science, Vol.59, 223-224, 2004.

[3] Y. W. Zhang, S. J. Qin, Fault detection of nonlinear processes using multiway kernel independent component analysis, Industrial & Engineering Chemistry Research, Vol.46, 7780-7787, 2007.

[4] J. D. Shao, G. Rong, J. M. Lee, Learning a data-dependent kernel function for KPCA-based nonlinear process monitoring, Chemical Engineering Research and Design, Vol.87, 1471-1480, 2009.

[5] K. Q. Weinberger, L. K. Saul, Unsupervised learning of image manifolds by semidefinite programming, International Journal of Computer Vision, Vol.70, 77-90, 2006.

[6] J. D. Shao, G. Rong, Nonlinear process monitoring based on maximum variance unfolding projections, Expert Systems with Applications, Vol.36, 11332-11340, 2009.

[7] Y. J. Liu, T. Chen, Y. Yao, Nonlinear process monitoring and fault isolation using extended maximum variance unfolding, Journal of Process Control, Vol.24, 880-891, 2014.

[8] A. Schwaighofer, V. Tresp, K. Yu, Learning Gaussian process kernels via hierarchical Bayes, Advances in Neural Information Processing Systems, 2004.

[9] T. J. Chin, D. Suter, Out-of-sample extrapolation of learned manifolds, Pattern Analysis and Machine Intelligence, Vol.30, 1547-1556, 2008.


Table3. Detection Rates of TE Process

| Type | Linear | | Polynomial | | | | | | RBF | | Improved KPCA | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Parameter | Null | | 1 | | 2 | | 3 | | 320 | | Null | |
| Fault Number | T2 | SPE | T2 | SPE | T2 | SPE | T2 | SPE | T2 | SPE | T2 | SPE |
| 0 | 0.007 | 0.011 | 0.003 | 0.005 | 0.046 | 0.563 | 0.006 | 0.245 | 0.003 | 0.001 | 0.017 | 0.066 |
| 3 | 0.046 | 0.041 | 0.045 | 0.006 | 0.282 | 0.812 | 0.153 | 0.631 | 0.021 | 0.147 | 0.135 | 0.372 |
| 5 | 0.266 | 0.998 | 0.283 | 0.997 | 0.450 | 0.960 | 0.358 | 0.735 | 0.260 | 0.563 | 1.000 | 1.000 |
| 10 | 0.046 | 0.084 | 0.882 | 0.426 | 0.646 | 0.967 | 0.560 | 0.902 | 0.456 | 0.751 | 0.750 | 0.910 |
| 14 | 1.000 | 0.738 | 1.000 | 0.165 | 1.0000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 |
| 19 | 0.238 | 0.823 | 0.915 | 0.260 | 0.392 | 0.986 | 0.009 | 0.816 | 0.083 | 0.368 | 0.400 | 0.887 |
| 21 | 0.592 | 0.486 | 0.598 | 0.050 | 0.572 | 0.887 | 0.455 | 0.762 | 0.466 | 0.550 | 0.552 | 0.718 |