Modeling Heterogeneous Time Series Dynamics to Profile Big Sensor Data in Complex Physical Systems

Bin Liu
Rutgers University, NJ, USA
binben.liu@rutgers.edu

Haifeng Chen, Abhishek Sharma, Guofei Jiang
NEC Laboratories America, Princeton, NJ, USA
{*haifeng, absharma, gfj*}*@nec-labs.com*

Hui Xiong
Rutgers University, NJ, USA
hxiong@rutgers.edu

***Abstract*—While a massive amount of time series can now be collected in many physical systems, it is a challenge to build an analytic model that can correctly profile the data because those time series usually exhibit various behaviors. In this paper we propose an integrated method to address the heterogeneity issue in modeling big time series data. We first extracts relevant features to summarize the underlying dynamics of those series. We present both linear and nonlinear feature extraction techniques, as well as a procedure to determine the right extraction method for individual time series. Given extracted features, our method further models the trajectory pattern of time series in the feature space. Both a regression based and a density based method are presented to profile different types of feature trajectories. Experimental results in a real power plant illustrate that our feature extraction and trajectory model are effective to profile various time series. Our method has been used to successfully detect anomalies in the system.**

***Keywords*-Time series, Trajectory model, Anomaly detection**

I. Introduction

With the decreased hardware cost and increased sensor capability, traditional physical systems undergo revolutionary changes recently in their computing, communication and storage capabilities. They are now equipped with a large network of sensors distributed across different parts of the system, which leads to a tremendous amount of time series data available to system operators. It is important to build effective models to profile those time series so that we can better understand the underlying dynamics of system operation and hence monitor its status to detect anomalies.

However, different time series in the collected data usually demonstrate totally different behaviors due to the diversities in system components and their functionalities. Figure 1 presents some examples of time series collected from a real power plant system. As we can see, they are dramatically different in terms of shape, trend, seasonal variation and periodicity. Some series exhibit deterministic periodic behaviors, whereas others show irregular curves in their evolutions. While there already exist a number of time series modeling techniques in the literature, current methods usually rely on certain assumptions, and are limited for the category of time series that follow those assumptions. For example, the commonly used autoregressive(AR) model [1] assumes that current observation in time series linearly depends on its previous observations. While the support vector regression(SVR) [2] has been proposed to handle nonlinear behaviors in time series, it requires the knowledge of correct window size to determine the number of predictors in the regression. In addition, some time series may fit poorly in regression based models, and can only be profiled by nonparametric techniques such as density based methods [3]. Given a massive number of heterogeneous time series, it is necessary to have a more general method to treat different time series differently, so that we can identify the best profile for each time series based on its characteristics.

To this end, this paper proposes an integrated method to model the dynamics of time series so that the heterogeneous behaviors in big sensor data can be correctly profiled. Given a time series $\mathbf{x} = \{x_1, x_2, \ldots, x_t, \ldots\}$, our method first extracts related features that best summarize its dynamics, and then builds a model to capture the evolutionary trajectory of extracted feature vectors. In the feature extraction step, we apply a sliding window on $\mathbf{x}$ to derive a sequence of state vectors $\mathbf{z}_t = [x_{t-d+1}, \cdots, x_{t-1}, x_t]$, which leads to a state matrix $Z = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n]^T$. Compact features are then extracted by applying subspace decomposition from Z. We use the singular value decomposition (SVD) to extract linear features, whereas a kernel based method is presented to extract features from nonlinear time series. We also present an intrinsic dimension based test [4], [5] to determine whether an linear or nonlinear feature extraction is appropriate for a given time series.

Since the extracted features summarize the underlying dynamics of time series, their trajectory reflects the evolution of time series. We will show that for many time series their evolution trajectories demonstrate stronger regularities in the feature space than in the original data space. Therefore, we further propose two techniques to model the trajectory traces in the feature space. Two techniques are proposed for that purpose. The vector-autoregressive (VAR) method, which has been shown be effective to describe a variety of trajectory shapes [6], is presented to model feature trajectory

This work was performed while the first author was an intern at NEC Laboratories America, Princeton, NJ.

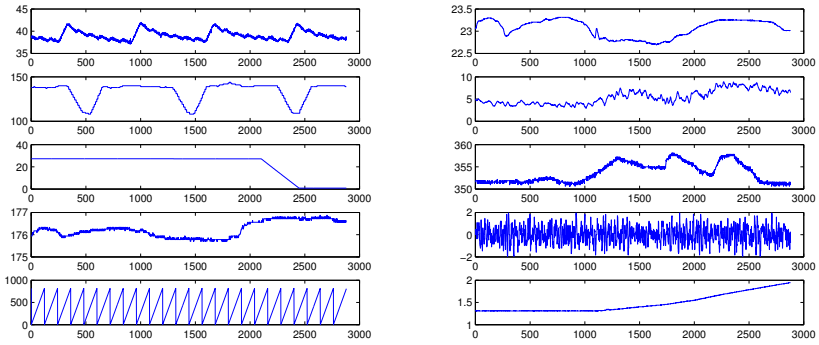


Figure 1: Heterogeneous time series from a physical system.

with regular patterns. We also present a density based approach to model feature trajectories with less regularities.

Since our feature extraction and trajectory modeling are performed on each individual time series, our method can be easily parallelized to handle data from a massive amount of sensors. That is, we divide the time series into several groups and each machine builds the profiles of series from a specific group. The learned profiles can be used to monitor the online status of time series. Given a new observation x_t of series x, we retrieve the model of x from the corresponding machine, compute the value of its feature vector, and compare it with that derived from the feature trajectory estimation. An alert will be issued if a large deviation is observed. Experimental results have demonstrated the good performance of our method in the system anomaly detection.

Contributions. This paper has made the following contributions: (1) We propose an integrated approach to profile heterogeneous time series collected from a large amount of sensors in physical systems. It first extracts relevant features from time series to represent the dynamics, followed by modeling the trajectory of feature values along the time. Our method has correctly profiled all the time series in a real power plant. It outperforms other state-of-the-art approaches in the application of anomaly detection. (2) In extracting features from time series, we have made significant improvements over current methods such as SSA to handle general behavior of time series. In addition to the linear subspace decomposition, our method also includes nonlinear feature extraction to model nonlinear time series. (3) Rather than focusing on individual feature values, we propose to model the evolution trace of time series features. Two trajectory modeling techniques are presented to handle different types of feature trajectories.

II. Extracting Features of Time Series Dynamics

Figure 2 illustrates the process of extracting relevant features from time series. We first use the sliding window based technique to construct a sequence of state vectors, i.e., the vectors that contain series' dynamics along the time. After that, we discover non-redundant features from state vectors. Both linear and nonlinear extraction methods

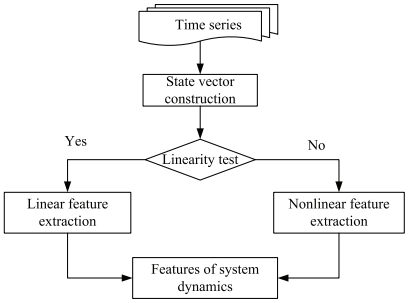


Figure 2: Feature extraction workflow.

are presented to cover various behaviors of time series. In addition, we also propose a statistical test procedure to determine the linearity of a given time series and hence choose the appropriate extraction method.

A. State Vector Construction

The time series in most physical systems are generated from underlying physical processes that can be modeled by some differential equations. That is, the current value x_t at time t is produced by a hidden function on its previous samples. While the underlying physical processes are usually unknown, that fact guides us to construct a state vector at each time t to cover the dynamics of time series at that moment. It is represented as a d-dimensional vector $\mathbf{z}_t \in R^d$ that contains the current observation x_t as well as its past $d-1$ samples.

$$\mathbf{z}_t = [x_{t-(d-1)}, ..., x_{t-1}, x_t]. \quad (1)$$

As long as the number of delayed samples in $\mathbf{z}_t$ is greater than the order of underlying physical process, the state $\mathbf{z}_t$ contains the full dynamics of $\mathbf{x}$ at time t. By constructing the vector z_t at every sample time, we can get the whole picture of the evolution in time series.

If the true value of d is known, the state vector can be directly used as the feature of time series. In practice, however, that information is unavailable. A common way to address that is to relax the d value at the beginning and then discover the compact representation of features from state vectors. In order to achieve that, we stack all the state vectors into a matrix $Z = [\mathbf{z}_1, \mathbf{z}_2, ...\mathbf{z}_n]^T$, which is called the Hankel matrix [7], and identify a low dimensional subspace

that contains all the information in Z. The projection of each state vector into that subspace is then regarded as the feature of time series $\mathbf{x}$.

There are different ways to discover the signal subspace from matrix Z. If time series $\mathbf{x}$ is linear, i.e., the current sample x_t is linearly dependent on its past observations, the singular value decomposition (SVD) can be directly applied to identify the subspace. However, nonlinear methods need to be utilized if the behavior of $\mathbf{x}$ violates the linear assumption. Before performing the corresponding feature extraction, we first propose a procedure the discover the linear or nonlinear characteristics of time series.

B. Intrinsic Dimension Based Test

A statistical test based on the concept of intrinsic dimension[4] is described here to discover the linear or nonlinear behavior of time seiries. Given a sequence of d-dimensional state vectors $\mathbf{z}_t$, $t = 1, \cdots$, the intrinsic dimension r describes the number of underlying variables that are needed to represent the state vector. Its estimation is based on the observation that for an r-dimensional data set, the number of pairs of points closer to each other than ν, as described in equation 2, is proportional to ν^r

$$C_n(\nu) = \frac{2}{n(n-1)} \sum_{i<j}^{n} \mathbf{1}_{||\mathbf{z}_i - \mathbf{z}_j|| < \nu} \tag{2}$$

where $\mathbf{1}$ is the indicator function of the event A. Then the intrinsic dimension r is defined as $r = \lim_{\nu \to 0} \lim_{n \to \infty} \frac{\log C_n(\nu)}{\log \nu}$. We compute $C_n(\nu)$ for different ν_i and fit a line through $[\log \nu_i, \log C_n(\nu_i)]$ to derive r.

Based on the estimated dimension r, we test whether the linear subspace decomposition is sufficient for time series $\mathbf{1}$. Given the state matrix $\mathbf{Z}$ from time series $\mathbf{x}$, we first apply the singular value decomposition $\mathbf{Z} = U \Sigma V^T$, where U and V represent the column and row spaces of $\mathbf{Z}$ respectively, and Σ are the singular values $\Sigma = diag(\lambda_1, ..., \lambda_r, \lambda_{r+1}, ..., \lambda_k)$, $k = \min\{d, n\}$. We then check whether the linear subspace with dimension r covers enough variances of the original space by computing

$$\gamma = \left[\frac{\lambda_1^2 + \lambda_2^2 + ... + \lambda_r^2}{\lambda_1^2 + \lambda_2^2 + ... + \lambda_k^2} \right]^{1/2} \tag{3}$$

If the γ value is larger than a predefined threshold, we use the linear subspace decomposition to extract features. Otherwise, we use the nonlinear feature extraction method.

C. Linear Feature Extraction

The linear subspace of matrix Z directly comes from the SVD results, i.e., the first r columns of U form the bases of signal subspace $U_s = [u_1, u_2, ..., u_r]$. Given a state vector $\mathbf{z}$, we can obtain its low-dimensional representation in the signal subspace as $\mathbf{y}_t = U_s^T \mathbf{z}_t$. Since $\mathbf{y}_t$ effectively summarizes the information in $\mathbf{z}_t$, we use it as the feature of time series $\mathbf{x}$ at time t.

D. Nonlinear Feature Extraction

For time series with nonlinear behaviors, we use a nonlinear function $\Phi(.)$ to transform the state vector $\mathbf{z}_t$ into $\Phi(\mathbf{z}_t)$, and then identify the signal subspace in the transformed vector space. The transformed vector space is also called 'feature space' in the literature, but we do not use the term here to avoid the confusion with the feature space described in Section III.

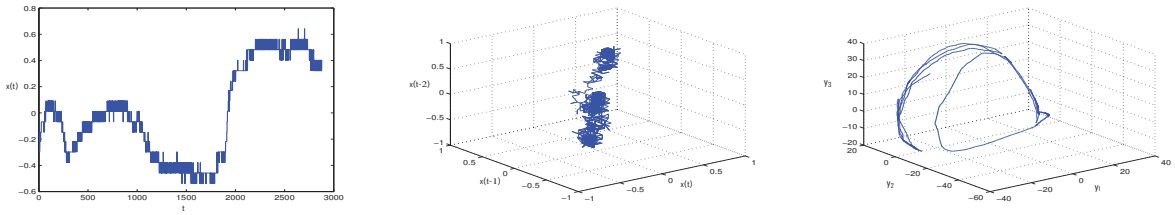
The base vectors $\{\tilde{u}_1, \cdots, \tilde{u}_r\}$ of signal subspace from transformed vectors $\Phi(Z) = [\Phi(\mathbf{z}_1), \Phi(\mathbf{z}_2), ..., \Phi(\mathbf{z}_n)]^T$ can be obtained by the eigen-decomposition of covariance matrix $C = \frac{1}{n} \sum_{j=1}^{n} \Phi(\mathbf{z}_j) \Phi(\mathbf{z}_j)^T$, which can be solved by using the kernel trick [8]. That is, instead of computing the mapping $\Phi(\mathbf{z}_j)$, we use common kernel functions such as the Gaussian kernel and polynomial kernel to represent their dot product $\mathcal{K}_{i,j} =< \Phi(\mathbf{z}_i) \cdot \Phi(\mathbf{z}_j) >$ in the eigen-decomposition, which turns into solving $\mathcal{K}\boldsymbol{\alpha} = \lambda \boldsymbol{\alpha}$, where matrix $\mathcal{K} = [\mathcal{K}_{i,j}]$ is called the kernel matrix, and $\boldsymbol{\alpha}$ is its eigen-vector associated with eigen-value λ.

It was shown in [9] that the kth base vector $\tilde{u}_k$ of the signal subspace, $k = 1, \cdots, r$, can be derived from $\boldsymbol{\alpha}^k$, i.e., the eigen-vector associated with kth largest eigen value, $\tilde{u}_k = \sum_{j=1}^{n} \alpha_j^k \Phi(\mathbf{z}_j)$, where α_j^k is the jth element in $\boldsymbol{\alpha}^k$. Given a transformed state vector $\Phi(\mathbf{z}_t^T)$, its projection on that base vector is $\tilde{u}_k^T \Phi(\mathbf{z}_j^T) = \sum_{j=1}^{n} \alpha_j^k \Phi(\mathbf{z}_j^T) \Phi(\mathbf{z}_j^T)$, which can be computed by the kernel function $\sum_{j=1}^{n} \alpha_j^k \mathcal{K}(\mathbf{z}_i, \mathbf{z}_j)$. The nonlinear features of time series are the projections of $\Phi(\mathbf{z}_t^T)$ on all the r base vectors in the transformed vector space.

III. Modeling System Dynamics

The profiling of data samples in the projected feature space have been studied for general data types [5], [10]. For example, [5] used the statistic of Hotelling T^2 to describe the distribution of projected features. However, since this paper deals with time series, we expect that feature vectors also exhibit temporal patterns in their evolutions. As an example, given a time series in Figure 3(a), its extracted 3-dimensional features in Figure 3(c) demonstrate strong regularity in the evolution path. We also plot the 3-dimensional state vectors (x_t, x_{t-1}, x_{t-2}) in Figure 3(b). Clearly the points in feature space demonstrate strong trajectory patterns. This is because the feature vector includes all the dynamics of time series. By modeling the trajectory of feature vectors, we can build a profile of time series dynamics.

Two techniques are proposed to model the feature vector trajectory. The Vector-Autoregression(VAR) is used first to capture trajectories with strong regularities. We will show that VAR model can cover a majority of time series we encountered in real systems. However, for some time series whose feature trajectories do not fit the VAR model well, we also provide a density based model to describe their behaviors. By combining those two complementary techniques, we can build the profile for all time series received in real applications.



(a) Time series (b) State vectors (c) Feature trajectory

Figure 3: An example of (a) the original time series from a real physical system; (b) the trajectory of 3-dimensional state vectors; (c) the trajectory of extracted features.

A. Vector-Autoregression Model

It has been shown that VAR model is effective to describe trajectories with strong regularities [6]. Given a sequence of feature vectors $\mathbf{y}_0, \cdots, \mathbf{y}_t$, VAR describes the relation between current feature $\mathbf{y}_t$ and its past p fetaures:

$$\mathbf{y}_t = \mathbf{c} + \sum_{i=1}^{p} \Pi_i \cdot \mathbf{y}_{t-i} + \varepsilon(t) \tag{4}$$

where Π_i is a $r \times r$ transition matrix, and $\mathbf{c}$ is the intercept vector. The noise $\varepsilon(t)$ reflects the influence of external randomness on the physical system.

We transform equation 4 into a compact expression $\mathbf{y}_t = \mathbf{B}\mathbf{w}_t + \varepsilon(t)$, where $\mathbf{B}$ is a $r \times (rp+1)$ parameter matrix $\mathbf{B} = [\mathbf{c}, \Pi_1, ..., \Pi_p]$ and $\mathbf{w}_t = [\mathbf{1}^T, \mathbf{y}_{t-1}^T, ..., \mathbf{y}_{t-p}^T]^T$. We assume the noise follows Gaussian distribution $\varepsilon(t) \sim \mathcal{N}(0, \Sigma)$, and use the least square method to estimate transition matrices and the intercept vector.

$$\hat{\mathbf{B}}_{k\times(kp+1)} = \left[\sum_{t=1}^{N} \mathbf{y}_t \mathbf{w}_t^T\right] \left[\sum_{t=1}^{N} \mathbf{w}_t \mathbf{w}_t^T\right]^{-1}. \tag{5}$$

We use the Bayesian Information Criteria (BIC) to determine the optimal lag p in equation 4. After obtaining model parameters $\{\hat{\mathbf{c}}, \hat{\Pi}_1, \hat{\Pi}_2, ..., \hat{\Pi}_p\}$, we can predict the next feature value based on past p features by equation 4.

Note that VAR model may not *always* lead to a good profile of time series. For some feature trajectories with a lot of randomness, it will produce large errors $\varepsilon(t)$ and hence is not reliable. Here we use the R-square(R^2) statistic to measure the goodness of fit in VAR, which is computed as

$$R^2 = 1 - \frac{\sum_{t=1}^{N} ||\mathbf{y}_t - \hat{\mathbf{y}}_t||^2}{\sum_{t=1}^{N} ||\mathbf{y}_t - \bar{\mathbf{y}}||^2} \tag{6}$$

where $\mathbf{y}_t$ is the feature vector, $\hat{\mathbf{y}}_t$ is its estimation based on equation 4, and $\bar{\mathbf{y}}$ is the mean $\bar{\mathbf{y}} = \sum_{t=1}^{N} \mathbf{y}_t / N$. Equation 6 can be interpreted as $R^2 = 1 -$ (residual sum of squares)/(total sum of squares). A large R^2 indicates that the model provides a good fit to the data.

B. Density Estimation Model

A non-parametric method is also proposed here to handle feature trajectories with low fitness in VAR model. It relies on the density distribution of feature vectors in the trajectory. However, instead of modeling feature vectors directly, we model the increment of feature vector $\Delta\mathbf{y}_t = \mathbf{y}_{t+1} - \mathbf{y}_t$ to reflect the temporal coherence in the trajectory. The kernel density based regression is used to describe the feature increment: $\Delta\mathbf{y}_t = \sum_{k=1}^{N_n(\mathbf{y}_t)} w(\mathbf{y}_t, \mathbf{y}_k)\Delta\mathbf{y}_k$, which leads to

$$\mathbf{y}_{t+1} = \sum_{k=1}^{N_n(\mathbf{y}_t)} w(\mathbf{y}_t, \mathbf{y}_k)(\mathbf{y}_{k+1} - \mathbf{y}_k) + \mathbf{y}_t \tag{7}$$

where $w(\mathbf{y}_t, \mathbf{y}_k)$ represents the weights of $\Delta\mathbf{y}_k$ in the regression, and $\mathbf{y}_k$ is one of the $N_n(\mathbf{y}_t)$ nearest neighbors of $\mathbf{y}_t$. That is, given all the feature vectors, we search the nearest neighbors of $\mathbf{y}_t$ and include the closest $N_n(\mathbf{y}_t)$ features in the density model. The motivation here is that if two feature vectors are similar, their path increment Δ should also be similar. Therefore we only include the most similar features in estimating the path increment.

In addition, among the selected feature $\mathbf{y}_k$s, those that are closer to $\mathbf{y}_t$ should contribute more to the estimation of $\mathbf{y}_{t+1}$ in equation 7. We use the kernel function $w(\mathbf{y}_t, \mathbf{y}_k) = \frac{K_h(||\mathbf{y}_t - \mathbf{y}_k||)}{\sum_{k=1}^{N_n(\mathbf{y}_t)} K_h(||\mathbf{y}_t - \mathbf{y}_k||)}$ to determine the weight value, where $K_h(b) = \frac{1}{h}K\left(\frac{b}{h}\right)$, $K(\cdot)$ is a kernel function, and h is the bandwidth of the kernel. In the experiment, we use $K(\cdot)$ as the Gaussian kernel and the bandwidth h is determined by employing a simple plug-in rule described in [11].

1) Comments.: We have presented two complementary strategies to model the feature trajectory. While the VAR model is effective in capturing the shape of trajectory, the density based approach is applicable to general feature distributions. We use VAR here to get an *accurate* prediction of features based on past feature values, whereas the density model is used to improve the model *coverage* for various time series collected from real systems. From the results in a real power plant system, we observe that around 93% of the 997 time series can be modeled by VAR model with a high fitness score. The remaining time series are then handled by the density model.

IV. Anomaly Detection in Time Series

Our proposed method is used to detect anomalies in time series, which are data points that significantly deviate from the normal pattern. The anomaly detection includes a training stage and a detection stage. In the training, we

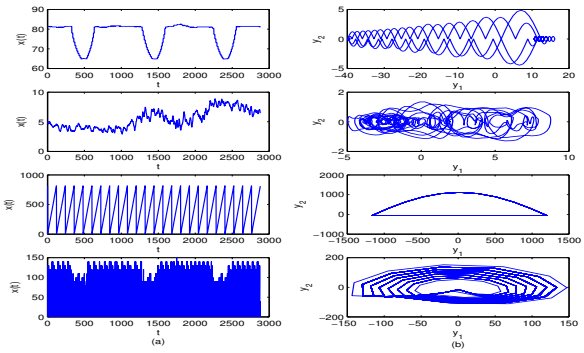


Figure 4: Four time series (left) and their extracted feature trajectories (right).

apply our method to normal time series x_t, $t = 0, \cdots, t$. As a result, we obtain the extracted features and a model that correctly profiles the evolution of feature vectors.

In the detection stage, given a new observation x_{t+1}, we perform the following steps to determine whether it is normal or abnormal. (1) Construct the state vector $\mathbf{z}_{t+1}$ from x_{t+1} and its past observations; (2) Compute the new feature value $\mathbf{y}_{t+1}$ based on the method in Section II; (3) Use past feature vectors to predict the new feature $\hat{\mathbf{y}}_{t+1}$ based on the method described in Section III; (4)If there is a large deviation between $\mathbf{y}_{t+1}$ and $\hat{\mathbf{y}}_{t+1}$, i.e., $\|\hat{\mathbf{y}}_{t+1} - \mathbf{y}_{t+1}\| \geq \delta$, x_{t+1} is regarded as abnormal and an alarm is generated.

The threshold δ for anomaly detection is determined based on the trade-off between false positives and negatives in different applications. In our experiments, the threshold δ is determined as the maximum residual obtained from the training process.

V. Evaluation

We apply the proposed method to model a large number of time series from a real power plant, and use it to detect anomalies in the system. In the following, we first describe the collected data in Section V-A. Section V-B then presents the training results from the data. The results of detection are described in Section V-C. Section V-D then evaluates the the effect of different sliding window size d, described in Section II-A, on the performance of proposed algorithm. In addition, we further look into the results and investigate the contribution of nonlinear features in the model performance, which will be described in Section V-E.

A. Data Description

The data contain 997 time series collected from sensors located at various positions in a real power plant. The sensors collected the operational data every 30 seconds. Our data cover a whole day of system operation, in which each time series contain 2880 samples. Some of the time series are illustrated in Figure 1. We can see that those time series vary dramatically in terms of shape, variance, covariance, trend, seasonal variation and periodicity.

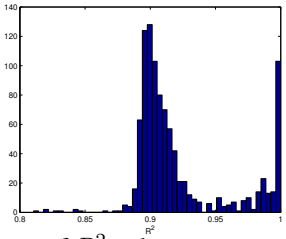


Figure 5: Histogram of R^2 values generated by VAR model.

We divide the time series into training and test sets. The training data contains 1920 samples of all time series, and the test set takes the remaining samples. We use the training data to build a model for each time series, and then use the test data to evaluate the performance of the model.

B. Model Construction

We follow the steps described in previous sections to build a profile for each time series. In the feature extraction stage, we compare the γ value in Equation (3) with 0.9 to determine the intrinsic dimension. Figure 4 plots some time series and their extracted feature vectors. As we can see, the evolution of feature vectors demonstrate strong regularities with different shapes such as polynomial, sinusoid, circle and ellipses. Note that here we only plot the first two dimension of feature vectors in Figure 4(b), whereas in reality feature vectors can have more than two dimensions.

Since many time series demonstrate regular feature trajectories, our VAR based technique can correctly model feature trajectories from a majority of time series. Figure 5 presents the histogram of VAR model fitness R^2s from all time series. It shows that most time series have a high fitness score on the VAR model. Here we choose the threshold 0.893. As long as the R^2 value is greater than the threshold, we use VAR to model the feature trajectory of time series. It turns out that VAR can cover 922 time series, around 92.48% of total series. For remaining time series, we use the density based approach to model their behaviors.

C. Anomaly Detection

Two commonly used metrics to evaluate a model in anomaly detection are the detection rate and false alarm rate. Since all the experimental data are collected from the system normal operations, the test time series can only be used to evaluate the false alarm rate of proposed method. In order to evaluate the detection rate, we manually inject some anomalies in the test data to mimic some faulty situations in system operations. By tuning different values of detection threshold δ as described in Section IV, we can obtain a receiver operating characteristic (ROC) curve to fully describe the performance of the algorithm.

In order to demonstrate the advantage of our proposed method, we compare three different algorithms to model time series and then detect anomalies: 1) the autoregressive (AR) model [1], 2) the singular spectrum analysis (SSA) [12], and 3) our proposed method. We compare our method

	AR model	d=10		d=50		d=100	
		SSA	**Proposed**	SSA	**Proposed**	SSA	**Proposed**
# false alarms	881	181	**5**	1864	**9**	3632	**9**

Table I: The total number of false alarms generated by the AR model, SSA model and our proposed method with different sliding window sizes.

anomaly level	AR model	d=10		d=50		d=100	
		SSA	**Proposed**	SSA	**Proposed**	SSA	**Proposed**
weak	0.8107	0.8867	**0.9007**	0.8933	**0.9187**	0.9000	**0.9413**
middle	0.8607	0.9067	**0.9227**	0.9067	**0.9360**	0.9133	**0.9567**
strong	0.8993	0.9133	**0.9460**	0.9133	**0.9567**	0.9400	**0.9807**

Table II: The detection precision rate generated by the AR model, SSA model and our proposed method with different sliding window sizes.

with AR model because AR is one of the most widely used methods for time series modeling. The SSA model is used here for comparison because it also uses a sliding window to extract state vectors from time series and decomposes the state matrix into different subspaces. However the SSA method does not model feature vectors, but instead reconstructs the original time series by removing the noise part in the decomposition.

Since both SSA and our proposed method use the slide window to construct state vectors, the size of sliding window d may affect the performance of algorithms. In the experiments we have tried different window sizes, and the methods are compared under each individual d value. In Section V-D we will further explain the effect of window size to the model performance.

1) False Alarm Rate: In the testing period, we use the maximum residual in the training process as the threshold to flag an alarm if $\|\hat{\mathbf{y}}_{t+1} - \mathbf{y}_{t+1}\| \geq \delta$. We count the number of alarms generated in the test time series. Since the original tets data are all from normal system operations, the number of observed alarms are false alarms. Table I shows the total number of false alarms generated by the AR model, SSA model and the proposed method with different sliding window sizes.

As we can see, our proposed method can significantly reduce the false alarms in anomaly detection. Compared with hundreds of alarms generated by other methods, our method only produce less than 10 false alarms.

2) Detection Rate: As all the data are from system normal operations, we randomly add synthesized anomaly event values. We randomly select a time t, and we inject 5 consecutive noise by choosing the value from $\{(1+\rho)\max\{x_t\}_{k=t-w}^{t+w}, (1-\rho)\min\{x_t\}_{k=t-w}^{t+w}\}$ for window size w. According to noise amplitude parameter ρ, we compare the anomaly detection performance for different level of anomalies (failures), namely weak, middle and strong. It is more difficulty to detect a weak anomaly as the failure value would be more similar to normal data. We count it a success of anomaly detection when it flags one alarm during $[t, t+4]$.

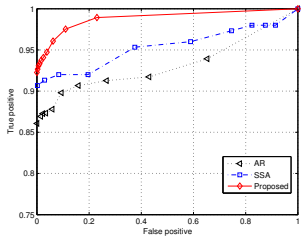


Figure 6: The ROC curves of anomaly detection generated by the AR model, SSA model and our proposed model, under the middle level anomaly and $d = 50$.

We repeat this process $K = 10$ times.

We define the detection precision $= \frac{1}{K}\left[\sum_{i=i}^{K}\left(\frac{1}{N}\sum_{j=1}^{N}\mathbf{I}(d_{ij})\right)\right]$ where $\mathbf{I}(\cdot)$ is an index function of detection success defined as

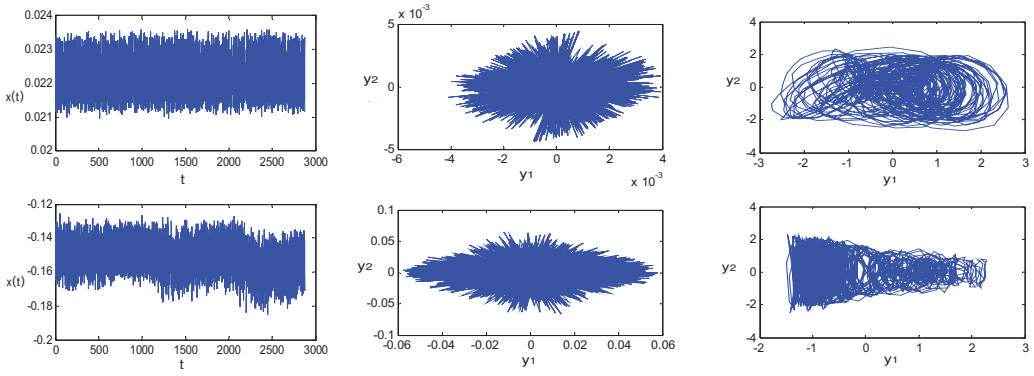
$$\mathbf{I}(d_{ij}) = \begin{cases} 1 & \text{if anomaly detection success} \\ & \text{at experiment } i \text{ for time series } j, \\ 0 & \text{otherwise} \end{cases}$$

We compare the results with three failure levels: week, middle and strong, and changing the embedding size d.

Table II shows the detection rate of our linear subspace decomposition based method compared to AR model and SSA model. From Table II and Table I, for middle level anomaly with $d = 50$, we can see that compared with AR model, only linear subspace decomposition based method can achieve 7.53 percent improvement of detection rate, while significantly remove the false alarms (reduced from 881 to 9); compared with SSA model, our method can achieve 1.33 percent improvement of detection rate, and can reduce false alarms from 1864 to 9. Also we set different normalized anomaly flag threshold and we got the ROC curves as shown in Figure 6. Actually, our method can improve the two baselines even more when the anomaly level is weak, which make the anomaly detection more difficult. For example, for weak level anomaly with $d = 50$, our method improves AR model by 10.8 percent, and improves SSA model by 1.8 percent respectively.

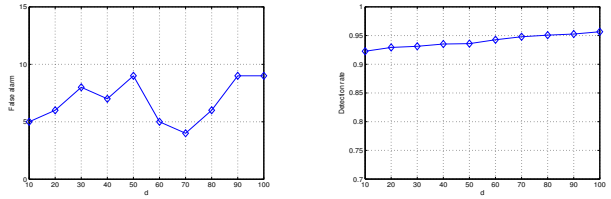
D. Effects of Sliding Window Size

The sliding window size d in Section II-A determines the dimension of state vectors, which will affect the performance of anomaly detection. A large window can contain more complete time series dynamics and hence make the model more accurate. However increasing the window size will also introduce more noise and a lot of redundant information, which may lead to an overfitted model. As shown in Table I and II, with the increase of d, the anomaly detection rate is improved, whereas more false positives are also introduced. However, compared with the SSA method, our method does not have a significant increase of false alarms for large ds. This is probably because the feature trajetory model can filter out irrelevant dynamics to make the model more robust.



(a) Time series (b) Linear (c) Nonlinear

Figure 8: Examples of feature trajectory extracted by linear and nonlinear subspace decomposition. (a) original time series; (b) feature trajectory extracted by linear subspace decomposition; (c) feature trajectory extracted by nonlinear subspace decomposition.



(a) False alarm. (b) Detection rate.

Figure 7: Anomaly detection precision versus the sliding window size.

Since the number of false positives generated by our method is quite small, we mainly investigate the effect of sliding window size on the anomaly detection rate. We set different values of d and perform experiments to get the corresponding detection rates, which is shown in Figure 7. It illustrates that detection rate improved with the increase of s. However, In the experiments we use 10.

E. Do Nonlinear Features Really Help?

We further look into the results to pick those nonlinear time series and check whether our nonlinear feature extraction in Section II-D contributes to the improvement of detection performance compared with linear method. Among the 997 time series, we get 9.3% of them whose feature vectors are obtained by nonlinear extraction. We apply linear feature extraction on those time series as well, and compare the performance between linear and nonlinear features.

It turns out that nonlinear features can produce more regular shapes in their evolution trajectories. Figure 8 plots two nonlinear time series and their features extracted by linear and nonlinear methods respectively. While it is hard to find strong geometry patterns in the trace of linear features in Fig. 8(b), the trajectories of nonlinear features are much more smooth as shown in Fig. 8(c). This is because nonlinear features correctly embed the dynamics of time series.

We compare the anomaly detection rate of those time series with linear and nonlinear features. Table III illustrates

Anomaly level	weak		middle		strong	
Method	linear	nonlinear	linear	nonlinear	linear	nonlinear
Detection rate	0.5524	**0.8000**	0.7079	**0.8413**	0.8619	**0.9206**

Table III: Comparison of anomaly detection rate using linear and nonlinear features applied to nonlinear time series.

the results with different levels of anomalies. For the middle level anomaly, nonlinear method can improve linear method by 13.34% in the detection. For the weak level anomaly nonlinear method can improve linear method by 24.76%. Such performance improvements are significant enough to conclude that the nonlinear feature extraction is a necessary part in our proposed method, which can better profile nonlinear time series and enhance the detection rate. Although in the system we studied, the nonlinear time series only occupy around 10% of total series, we believe that more nonlinear time series will be encountered in other complex systems.

VI. Related Work

Our work is related to two areas of interest: time series modeling, and the anomaly detection for complex physical systems. Time series modeling is an active research area in several communities including data mining and a number of methods have been proposed. The most widely used techniques include the auto-regressive (AR), moving average (MA), and ARMA models, which assume the current observation of time series is linearly dependent on its past values or some noise items [1]. In order to handle nonlinear behaviors, the paper [2] provides a survey on using support vector regression (SVR) to model time series. Kalman filters and state space models [1] have also been widely used for sequence and time series data, and has a wide range of applications.

Understanding and capturing the underlying system dynamics have been widely investigated in time series analysis [13], [14]. For example, the delay coordinate embedding [14] was proposed to study the chaotic behavior of time series.

However, that method requires expensive computations to obtain a reliable delay parameter as well as the embedding dimension, which is not well suited to handle a massive number of time series. The singular spectrum analysis (SSA) [12] avoids such an issue by setting the delay as one and using a fixed embedding dimension. But SSA decomposes time series based on the least square criterion, which is optimal only when the underlying dynamics is linear. In addition, SSA mainly focuses on the decomposition and reconstruction of time series to study its individual component such as trend and periodic items. It does not model the whole trajectory of time series in the embedded feature space. Compared with those methods, our approach handles both linear and nonlinear dynamics of time series, which can properly address the various behaviors in big sensor data.

Trajectory mining is also attracting a lot of attentions recently due to the increasing popularities of spatial temporal data in many applications. For instance, [15] proposed a trajectory clustering algorithm to group similar trajectories. The paper [16] estimated the traffic density given a lot of vehicle trajectories. In this paper we borrow the idea from trajectory mining to model the evolution path of time series in the feature space. However, our method does not compare different trajectories, but rather model the trace of each individual trajectory. We use the VAR model to describe trajectories with strong regularities, and also propose a density based technique to model irregular trajectories. As a result, different evolution patterns of time series features can be captured in our method.

Anomaly detection has been studied for a long time in data mining community. [17] presented an excellent and comprehensive survey. Some anomaly detection methods for time series data include [18]. The anomaly detection methods have been applied to a wide range of application domains including spacecraft systems [19], system health management[20], automobile [21], power system [22], and so on. With the increasing number of sensors installed in physical systems, we will see more data mining based applications to facilitate the system management tasks.

VII. Conclusions

This paper has proposed a method to model a large number of time series with heterogeneous behaviors. We have presented a technique to extract appropriate features from time series that can represent its underlying dynamics. We have also proposed two techniques to model the evolution pattern of feature vectors along the time. By combining the feature extraction and trajectory model, we have correctly profiled all the time series in a physical power plant. We have demonstrated the effectiveness of the proposed method to detect anomalies in that system.

References

[1] J. D. Hamilton, *Time-series analysis*. Princeton Univerity Press, 1994.

[2] N. Sapankevych and R. Sankar, "Time Series Prediction Using Support Vector Machines: A Survey," *Computational Intelligence Magazine, IEEE*, vol. 4, no. 2, pp. 24–38, 2009.

[3] A. Harvey and V. Oryshchenko, "Kernel density estimation for time series data," *Journal of Forecasting*, vol. 28.

[4] P. Grassberger and I. Procaccia, "Measuring the strangeness of strange attractors," *Physica D*, vol. 9, pp. 189–208, 1983.

[5] H. Chen, G. Jiang, and K. Yoshihira, "Monitoring high-dimensional data for failure detection and localization in large-scale computing systems," *IEEE Trans. on Knowl. and Data Eng.*, vol. 20, no. 1, Jan. 2008.

[6] Y. Tao, C. Faloutsos, D. Papadias, and B. Liu, "Prediction and indexing of moving objects with unknown motion patterns," ser. SIGMOD '04, 2004, pp. 611–622.

[7] J. R. Partington, *An Introduction to Hankel Operators*. Cambridge University Press, 1989.

[8] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[9] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, Jul. 1998.

[10] L. Ralaivola, F. d'Alche Buc, U. Pierre, and M. Curie, "Dynamical modeling with kernels for nonlinear time series prediction," in *NIPS*, 2003, p. 2004.

[11] M. P. Wand and M. C. Jones, *Kernel Smoothing*. Chapman and Hall/CRC, 1995.

[12] R. Vautard and M. Ghil, "Singular spectrum analysis in nonlinear dynamics, with applications to paleoclimatic time series," *Physica D*, vol. 35, no. 3, pp. 395–424, 1989.

[13] F. Takens, "Detecting strange attractors in turbulence," *Dynamical Systems and Turbulence*, vol. 898, pp. 366–381, 1981.

[14] M. Small, *Applied Nonlinear Time Series Analysis: Applications in Physics, Physiology and Finance*. World Scientific Publishing Company, 2005.

[15] H.-P. Kriegel, M. Renz, M. Schubert, and A. Zfle, "Statistical density prediction in traffic networks," in *SDM 2008*. SIAM, 2008, pp. 692–703.

[16] J.-G. Lee, J. Han, and K.-Y. Whang, "Trajectory clustering: a partition-and-group framework," ser. SIGMOD '07, 2007, pp. 593–604.

[17] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 15:1–15:58, Jul. 2009.

[18] J. Ma and S. Perkins, "Online novelty detection on temporal sequences," ser. KDD '03, pp. 613–618.

[19] R. Fujimaki, T. Yairi, and K. Machida, "An approach to spacecraft anomaly detection problem using kernel feature space," ser. KDD '05.

[20] S. Salvador and P. Chan, "Learning States and Rules for Detecting Anomalies in Time Series," *Applied Intelligence*, vol. 23, no. 3, pp. 241–255, Dec. 2005.

[21] R. Fujimaki, T. Nakata, H. Tsukahara, and A. Sato, "Mining abnormal patterns from heterogeneous time-series with irrelevant features for fault event detection," in *SDM*, 2008, pp. 472–482.

[22] H. Mori, "State-of-the-art overview on data mining in power systems," in *IEEE Power Systems Conference and Exposition, 2006.*