Systems & Control Transactions

Original Research Article
Peer Reviewed Conference Proceeding
ESCAPE 36 - European Symposium on Computer Aided Process Engineering
Sheffield, UK. 21-24 June 2026

PSE PRESS

Industrial batch process online fault detection using machine learning

Oliver Pennington[a], Adam Wilson[b], Carolina Cruz[b], and Dongda Zhang[a]*

[a] University of Manchester, Department of Chemical Engineering, Manchester, Greater Manchester, UK
[b] KEIT Industrial Analytics, Didcot, Oxfordshire, UK
* Corresponding Author: dongda.zhang@manchester.ac.uk.

ABSTRACT

As industries pursue more sustainable and flexible manufacturing strategies, batch processes continue to play a vital role across a wide range of applications. Batch operations offer the ability to handle diverse feedstocks and accommodate varying product specifications. These processes are broadly used in sectors such as pharmaceuticals, specialty chemicals, food production, and bioprocesses, where precise control over reaction conditions and product quality is essential. However, maintaining optimal conditions in a batch process can be challenging due to the minimal opportunities for mid-batch interference. This work focuses on a real industrial batch process that frequently sees batches with poor yields resulting in large financial losses. Despite utilizing a mid-infrared spectrometer analyzing the batch medium in real-time, the reduced product accumulation observed in faulty batches is not evident until over a third of the batch time has passed, by which point the batch is not economically viable to abort. This study applies and compares various machine learning based fault detection strategies, including multiple Principal Component Analysis (PCA) variants and autoencoder variants, to identify faulty batches as soon as possible, since resetting reset the batches earlier can maximize overall productivity. The findings of this study offer faster and more robust fault detection than typical PCA for this industrial batch process, reducing time lost to faulty batches and improving overall productivity. This work supports the transition towards autonomous and digitalized batch manufacturing while providing an in-depth comparison between several online fault detection strategies on real industrial data.

Keywords: Fault Detection, Batch Process, Machine Learning, Process Control, Modelling

INTRODUCTION

Batch processes remain a cornerstone of modern manufacturing systems due to their versatility and robustness, particularly in applications requiring product customization, process flexibility, or adaptability to evolving feedstock supply chains. Despite these advantages, fault detection in batch processes remains challenging due to inherent process complexity, time-varying dynamics, and measurement noise. As a result, distinguishing genuine process faults from normal operational variability requires robust monitoring frameworks capable of reliably identifying true anomalies in real time.

PCA is a well-established multivariate statistical method that has been widely applied to fault detection and process monitoring in batch processes [1–4]. Owing to its simplicity, interpretability, and strong theoretical foundation, PCA serves as a natural benchmark against which more advanced fault detection strategies can be compared. However, conventional PCA treats observations as independent and does not explicitly account for process dynamics, which limits its effectiveness for batch operations.

To address this limitation, Dynamic PCA (DPCA) extends the traditional PCA framework by incorporating time-lagged measurements, such that consecutive datapoints are combined into a single model input. This enables DPCA to capture temporal correlations that are intrinsic to batch processes. In addition, this study considers a gradient-space PCA approach, motivated by the observation that rates of change in process variables can be more informative than raw concentration measurements for identifying abnormal process behavior.

Furthermore, a moving-window Dynamic Functional

Principal Component Analysis (DFPCA) framework is also investigated. DFPCA employs functional representations, that can be nonlinear, to model batch trajectories, allowing intricate process dynamics to be captured while simultaneously smoothing noisy measurements. These capabilities are particularly advantageous in batch environments where measurement noise and nonlinearity can obscure early fault signatures.

Finally, autoencoder-based methods are explored due to their ability to model nonlinear relationships that cannot be captured by linear PCA-based techniques. Autoencoders are Artificial Neural Networks (ANNs) that reconstruct input data after compression through a lower-dimensional latent space. In this study, a dynamic (or lagged) autoencoder variant is considered, where consecutive time points are treated as a single input, analogous to DPCA. As with DPCA and DFPCA, the number of incorporated lags can be adjusted to control the extent to which process dynamics are embedded within the latent representation.

The primary objective of this study is to detect faulty batches as early as possible – ideally before deviations become apparent in the raw process data – while maintaining a clear distinction between normal and faulty operation to ensure confidence in fault detection outcomes. The proposed methodologies are therefore evaluated based on their ability to quickly and reliably distinguish faulty batches from normal operation in real time. The remainder of this paper is organized as follows: the *Methodology* section describes the data structure, the calculation of fault detection metrics, and implementation details of each faut detection strategy; the *Results and discussion* section compares the performance of the different fault detection strategies; and overall *Conclusions* are provided in the final section.

METHODOLOGY

Data availability

The thirteen batches recorded in this study span at least 3000 minutes each with concentration measurements of eight components taken every 2.1 minutes using a mid-infrared spectrometer. Three faulty batches have previously been identified; the remaining ten non-faulty batches are partitioned into eight randomly selected training batches and two validation batches. Faulty batches visibly deviate from normal operation around the 1000-minute mark, which this study aims to improve upon using the various process monitoring strategies. Data is passed through a short-span median filter and random-walk Kalman filter, which is well suited to processes with frequent measurements and slow dynamics [5], to smooth data while still being suitable to online application.

Fault detection metrics

Fault detection can be conducted by monitoring two different metrics over a moving window. If we consider the process data $\mathbf{X}$ to be reconstructed as $\widetilde{\mathbf{X}}$ with residual error $\mathbf{E}$ as in Equation (1):

$$\mathbf{X} = \widetilde{\mathbf{X}} + \mathbf{E}, \tag{1}$$

then through a dimensionality reduction framework, such as PCA, we can calculate both latent space and reconstruction space monitoring statistics. The latent space Hotelling's T^2 statistic and reconstruction space Q-residual can be calculated as in Equations (2) and (3):

$$T^2 = \mathbf{z}^\top \mathbf{\Lambda} \mathbf{z}, \tag{2}$$

$$Q = \|\mathbf{e}\|_2^2, \tag{3}$$

where $\mathbf{z}$ is the latent score vector, $\mathbf{\Lambda}$ contains latent variances, and $\mathbf{e}$ is the residual vector.

In this study, we consider a 50-input moving window for fault detection as opposed to single points to improve the robustness of fault detection. This study primarily focuses on the Hotelling's T^2 metric due to its superior overall performance on this case study.

Fault detection strategies

In this section, all fault detection strategies are applied to the dataset which is standardised such that the training dataset has a mean of zero and a standard deviation of one, with the exception of gradient-space PCA which is standardized after variable gradients are approximated.

Principal Component Analysis

Under PCA, the reconstruction of the standardised data matrix $\mathbf{X}$ shown in Equation (1) is therefore written as in Equation (4):

$$\mathbf{X} = \mathbf{Z}\mathbf{P}^\top + \mathbf{E}, \tag{4}$$

where $\mathbf{Z}$ is the matrix of scores in the latent space that correspond to $\mathbf{X}$, and $\mathbf{P}$ is the loading matrix. In this work, Singular Value Decomposition (SVD) is utilized to compute the loading matrix due to its efficiency and computational stability [6].

To keep only essential linear relationships and avoid capturing noise, the Principal Components (PCs) explaining the least variance are removed. The reconstructed dataset using the first k PCs is shown in Equation (5):

$$\widetilde{\mathbf{X}} = \sum_{i=1}^{k} \mathbf{Z}_i \mathbf{P}_i^\top. \tag{5}$$

Although PCA is a widely applied method for process monitoring and fault detection, it is limited by its inability to capture nonlinear and temporal relationships, preventing PCA from incorporating dynamic and complex relationships that can dominate batch processes.

Dynamic Principal Component Analysis

One way in which PCA can be improved is to incorporate past information, not just current measurements. Dynamic Principal Component Analysis (DPCA) is similar to PCA; the only difference is that the input data for DPCA is not a single vector of current variable values, it is instead a series of consecutive vectors of the variables up to the current measurement. The number of consecutive measurements included in each input is referred to as the number of 'lags'. In this study, we use a 100-lag DPCA approach that allows the DPCA model to capture temporal relationships that can be more informative than raw measurements in batch processes. The PCA dataset $\mathbf{X}$ with N measurements of each process variable can be augmented into a DPCA dataset $\mathbf{W}$ with L lags as in Equations (6) and (7):

$$\mathbf{w}(n) = \left[\mathbf{x}(n)^\top, \mathbf{x}(n-1)^\top, \dots, \mathbf{x}(n-L+1)^\top\right]^\top, \tag{6}$$

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}(L-1)^\top \\ \mathbf{w}(L)^\top \\ \vdots \\ \mathbf{w}(N-1)^\top \end{bmatrix}. \tag{7}$$

Gradient-Space Principal Component Analysis

Another way in which temporal information can be incorporated is using gradient-space PCA, which is conducted on the derivatives of process variables, as opposed to their actual values. Since batch processes often show first order dynamics, for example in chemical, fermentation, and some crystallization systems, the gradient-space can be more informative than the absolute value of process variables.

The main challenge with gradient-space PCA is the fact that insufficiently smoothed data will see noise amplification in the gradient-space [7]. Therefore, a Locally Weighted Scatterplot Smoothing (LOWESS) filter is used so that smoothness can be tuned by altering the filter span [8]. However, the LOWESS filter span causes a delay between when a measurement is taken and when the corresponding smoothed value is returned. Therefore, gradient-space PCA can only be implemented if the delay is sufficiently small. In this study, measurements are taken frequently, relative to the batch time horizon, and a relatively small span is required, thus facilitating the application of gradient-space PCA.

After data smoothing, the gradient of each process variable $\dot{x}_j^{(v)}(t)$ is approximated as in Equation (8):

$$\dot{x}_j^{(v)}(t) = \frac{x_j^{(v)}(t) - x_j^{(v)}(t-\Delta t)}{\Delta t}, \tag{8}$$

where $x_j^{(v)}(t)$ is the measurement of variable v in batch j at time t, and Δt is the time between measurements. Thereafter, the gradient-space data is standardised to allow PCA to be conducted.

Functional Principal Component Analysis

Functional Principal Component Analysis (FPCA) considers each process trajectory as a continuous function over time, rather than a vector of measurements. By modeling the smoothed temporal evolution of the profile, FPCA can better capture nonlinear dynamic patterns. For the multivariate instance where each batch trajectory consists of V functional variables $x_j^{(v)}(t), v = 1, \dots, V$, the variation profile of each process variable is described by the weighted summation of B_v basis functions $\phi_b^{(v)}(t)$ as in Equation (9):

$$\tilde{x}_j^{(v)}(t) = \mu^{(v)}(t) + \sum_{b=1}^{B_v} c_{jb}^{(v)} \phi_b^{(v)}(t). \tag{9}$$

Each observation j collects the vector of coefficients as $\mathbf{c}_j = \left(c_{j1}^{(1)}, \dots, c_{jB_1}^{(1)}, c_{j1}^{(2)}, \dots c_{jB_2}^{(2)}, \dots, c_{j1}^{(V)}, \dots, c_{jB_V}^{(V)}\right)$ which are collected in the matrix $\mathbf{C}$. PCA is applied to the coefficient matrix to yield the loading matrix $\mathbf{P}$ and the matrix of shared multivariate FPCA scores, as in Equation (10):

$$\mathbf{C} = \mathbf{Z}\mathbf{P}^\top + \mathbf{E}. \tag{10}$$

The eigenfunctions $\psi_i^{(v)}(t)$ of variable v are obtained by expressing the appropriate loading block in its basis as in Equation (11):

$$\psi_i^{(v)}(t) = \sum_{b=1}^{B_v} P_{bi}^{(v)} \phi_b^{(v)}(t), \tag{11}$$

where $P_{bi}^{(v)}$ represents the loading rows of $\mathbf{P}$ associated with variable v. Multivariate FPCA scores Z_{ji} therefore satisfy Equation (12):

$$Z_{ji} = \sum_{v=1}^{V} \sum_{b=1}^{B_v} c_{jb}^{(v)} P_{bi}^{(v)}, \tag{12}$$

which demonstrates that a single set of scores summarizes the joint variation across all variables. Similarly to PCA, the coefficients that explain the least variance are removed, meaning only k scores remain, as in $\mathbf{z}_j = \left(Z_{j1}, Z_{j2}, \dots, Z_{jk}\right)$, giving the approximate reconstruction of the coefficients $\tilde{\mathbf{c}}_j$ as in Equation (13):

$$\tilde{\mathbf{c}}_j = \sum_{i=1}^{k} Z_{ji} P_{\cdot i}. \tag{13}$$

Ultimately, variable profiles are reconstructed using the multivariate scores and eigenfunctions as in Equation (14):

$$\tilde{x}_j^{(v)}(t) = \mu^{(v)}(t) + \sum_{i=1}^{k} Z_{ji} \psi_i^{(v)}(t), \tag{14}$$

where the scores Z_{ji} are shared across all variables, unlike the coefficients $c_{jb}^{(v)}$ in Equation (11).

Within the FPCA framework described in Equations (9-14) there exist several options, namely different basis function types as well as the period over which the basis functions are fit. This study considers B-spline basis functions due to their flexibility potentially being able to capture complex batch dynamics. However, the

additional flexibility can capture noise, and so this study also considers an orthonormal polynomial approach as such functions can provide a smoother fit. Since this study focuses on thirteen batches, fitting the basis functions to the entire profile would yield just thirteen observations per variable, which is impractical for trying to identify statistically significant faults. Therefore, this study considers a dynamic (moving-window) FPCA (DFPCA) approach where basis functions are fit to the 100-lag segments from the DPCA data $\mathbf{W}$ in Equation (7).

Autoencoders

Autoencoders are a type of ANN that learn compact representations of higher-dimensional data through data reconstruction. Autoencoders can capture nonlinear relationships, meaning they can potentially uncover the more complex relationships in batch processes that standard PCA cannot. As illustrated in Figure 1, an autoencoder is composed of an encoder that compresses the original data $\mathbf{x}$ onto the lower-dimensional latent space $\mathbf{z}$, which is passed into the decoder that then reconstructs the original input data $\tilde{\mathbf{x}}$.

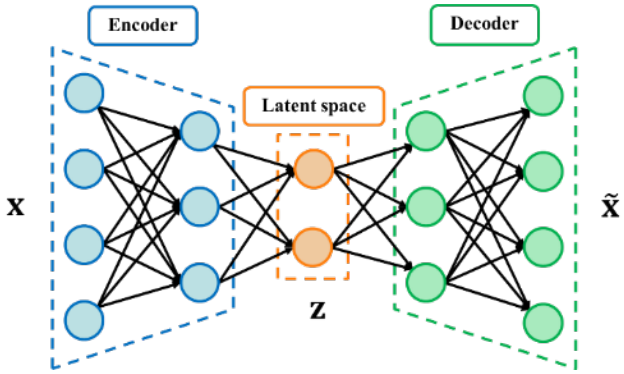

Figure 1. Example of an autoencoder architecture with 4-dimensional input, a 3-dimensional hidden layer in the encoder, and a 2-dimensional latent space. The decoder architecture always reflects that of the encoder, but the decder has its own unique weights and biases.

The autoencoder can be described by Equations (15a) and (15b):

$$\mathbf{z} = f_{\text{enc}}(\mathbf{x}), \tag{15a}$$

$$\tilde{\mathbf{x}} = f_{\text{dec}}(\mathbf{z}), \tag{15b}$$

where $f_{\text{enc}}(\cdot)$ is the function representing the encoder and $f_{\text{dec}}(\cdot)$ is the function representing the decoder. Autoencoders can also be trained on the 100-lag segments from the DPCA data $\mathbf{W}$ in Equation (7), so this study also considers a 100-lag autoencoder to capture temporal relationships, not just an autoencoder for modelling $\mathbf{X}$.

Autoencoder training focuses on minimizing the reconstruction loss described by the Mean Squared Error (MSE) loss function over J observations in Equation (16):

$$MSE = \frac{1}{J}\sum_{j=1}^{J}\left\|\mathbf{x}_j - \tilde{\mathbf{x}}_j\right\|_2^2. \tag{16}$$

The number of epochs (times passed through the training dataset during training) can be increased to improve fitting to the training dataset. However, using a higher number of epochs is both more computationally expensive and can lead to overfitting of the training data and poor fitting of the validation dataset. Another way in which the validation performance of the autoencoder can be improved is to introduce a penalization term into the loss function that penalizes the size of the autoencoder weights and biases during training, thus shrinking redundant parameters. In this study, 250 epochs are used alongside L2 regularization (sum of squared weights and biases). Finally, different autoencoder architectures are tested to identify the best performer on validation data.

It is also worth noting that since the autoencoder latent variables are not guaranteed to be orthonormal or centred like PCA scores, the Hotelling-type T^2 fault detection metric is approximated using a slightly different formula shown in Equation (17):

$$T^2_{AE} = (\mathbf{z} - \bar{\boldsymbol{z}})^{\top}\boldsymbol{\Sigma}_{\boldsymbol{z}}^{-1}(\mathbf{z} - \bar{\boldsymbol{z}}), \tag{17}$$

where $\boldsymbol{\Sigma}_{\boldsymbol{z}}$ is the latent covariance matrix, and $\bar{\boldsymbol{z}}$ is the latent mean vector.

RESULTS AND DISCUSSION

Principal Component Analysis performance

As mentioned in the *Fault detection metrics* section, we consider a 50-input moving window for fault detection. In this window we monitor the percent of datapoints that exceed the 99% confidence limit of the T^2 statistic for both normal (training and validation) and faulty batches. The earliest window that demonstrates the clearest distinction between normal and faulty batches is chosen as the best fault detection window, the results of which are illustrated in Figures 2-4.

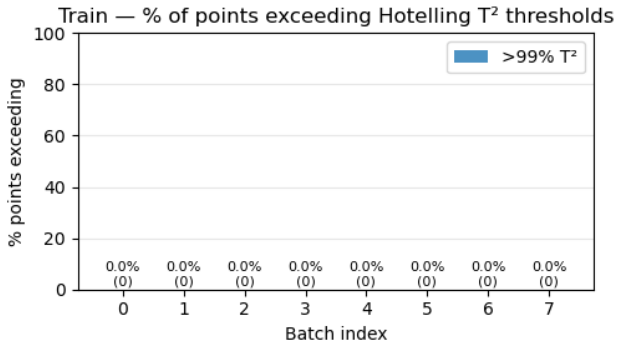

Figure 2. Bar chart showing percent of datapoints in each (normal operation) training batch where T^2 threshold is violated.

Looking at Figures 2 and 3, it is evident that the PCA does not flag any of the datapoints within this 50-datapoint window as exceeding the 99% confidence limit of the T^2 statistic, meaning PCA can correctly classify both the training and validation batches as normal operation.

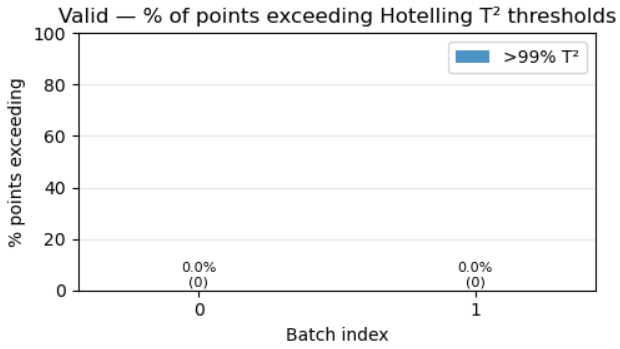

Figure 3. Bar chart showing percent of datapoints in each (normal operation) validation batch where T^2 threshold is violated.

Moreover, Figure 4 shows that PCA uncovers several instances where data within the fault detection window exceeds the 99% confidence limit of the T^2 statistic, therefore providing clear distinction between the normal and faulty batches.

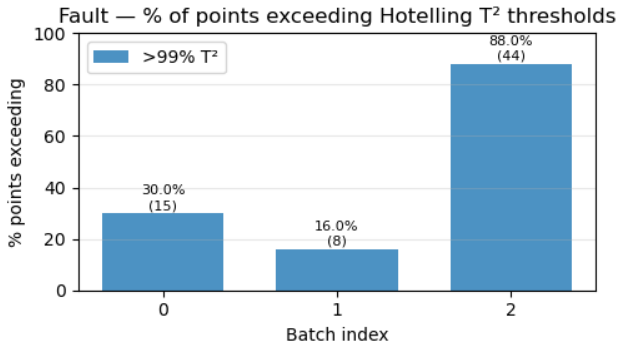

Figure 4. Bar chart showing percent of datapoints in each faulty test batch where T^2 threshold is violated.

The optimal fault detection window for the benchmark PCA requires data up to the 638th minute, a significant improvement upon observing faulty batches through raw data at around 1000th minute. If $\%FP$ is the highest percent of faults detected in any single normal batch and $\%TP$ is the lowest percent of faults detected in any single faulty batch, the fault detection margin $\%FDM$ is calculated as in Equation (18):

$$\%FDM = \%TP - \%FP. \tag{18}$$

The fault detection margin for the PCA benchmark is therefore 16% (since $\%FP = 0\%$, and $\%TP = 16\%$ for the benchmark PCA). The higher the fault detection margin, the more robustly the faulty batches can be distinguished from normal operation. Despite PCA reducing the time for faulty batch detection, there is still room for improvement in terms of both fault detection time and fault detection margin. This motivates the need to investigate more state-of-the-art process monitoring strategies.

Comparison of fault detection strategies performances with benchmark PCA

For the remaining fault detection strategies, performance is compared to the benchmark PCA in Table 1.

Table 1: Overall comparison between fault detection methods, showing online fault detection time and fault detection margin. The DFPCA abbreviations *Spln*. And *Poly*. stand for *B-spline* and *orthonormal polynomials*, respectively.

Model	Online fault detection time / min	Fault detection margin / %
PCA (*benchmark*)	638	16
100-lag DPCA	460	100
Gradient-space PCA	355	42
100-lag DFPCA (*Spln.*)	439	52
100-lag DFPCA (*Poly.*)	407	100
Autoencoder	344	0
100-lag Autoencoder	386	52

Looking at Table 1, all dynamic methods (those involving lagged data or operating in the gradient-space) outperform the benchmark PCA in terms of online fault detection time and margin. In fact, the only method that does not improve upon the benchmark PCA is the regular autoencoder. The stronger performance of gradient-space and lag-based strategies is likely because they better incorporate temporal relationships which can be more informative of the true state of a batch process.

It can also be seen that the two best performing approaches, in terms of fault detection margin, are 100-lag DPCA and 100-lag orthonormal polynomial DFPCA. Although the lag-based models use a substantially larger input space, due to the inclusion of time-lagged variables, it can achieve stronger dimensionality reduction than standard PCA, yielding a latent space that more effectively captures the system dynamics. For example, to explain 95% of the data variance, the benchmark PCA has an input dimension of 8 and latent space dimension of 6 (25% reduction), whereas the 100-lag DPCA has an input dimension of 800, and a latent dimension of just 18 (97.75% reduction). The inability for the latent space to capture system dynamics for lower-dimension inputs could be the reason that the standard autoencoder fails to distinguish faulty batches from normal operation, meanwhile the higher-dimensional 100-lag autoencoder

performed strongly with a 250 minute reduction in the fault detection time and 36-point reduction to the fault detection margin with respect to benchmark PCA.

The 100-lag DFPCA performs stronger when using orthonormal polynomials than B-splines, with a fault detection time reduction of over 30 minutes and a 48-point improvement in fault detection margin. Evidently, the use of smoother functions better describes the system which implies that the more flexible B-splines are capturing excessive noise instead of additional process information.

Although the fastest fault detection strategy is gradient-space PCA, it performs weaker than all the lag-based fault detection strategies in terms of fault detection margin, which is likely due to the aforementioned noise amplification in the gradient-space. This indicates that more smoothing is required to reduce the noise in the original data, however increased smoothing will both require a larger LOWESS filter span, thus raising the fault detection time, while also inadvertently removing some useful process dynamics.

Overall, the best performing fault detection strategy for this industrial batch process is found to be that using the 100-lag orthonormal polynomial DFPCA approach due to its excellent fault detection margin of 100% and a competitively low fault detection time of 407 minutes.

CONCLUSIONS

As batch processing continues to play a vital role in modern manufacturing and sustainable production, ensuring consistent and normal operation is becoming increasingly important. Through the use and comparison of these machine learning techniques, faulty batch detection can be achieved within the first 400 minutes; a significant improvement upon the 1000 minutes taken when observing the raw data and 638 minutes when utilizing standard PCA. In addition, a more statistically significant distinction can be found between faulty and non-faulty batches, meaning that fault detection robustness has been notably improved in comparison with the benchmark PCA. This study also demonstrates the benefits of utilizing fault detection strategies that exploit lag-based and gradient-space foundations for fault detection in batch processes. This study therefore paves the way for fault detection in industry-scale batch processes.

ACKNOWLEDGEMENTS

The authors thank BBSRC for providing funding for this project, along with KEIT Industrial Analytics who provided data and helped oversee the project.

AUTHOR IDENTIFIERS

Author ORCIDs:
Pennington O: 0009-0009-8119-871X
Zhang D: 0000-0001-5956-4618

REFERENCES

1. Gunther JC, Seborg DE, Baclaski J. Fault detection and diagnosis in industrial fed-batch fermentation. 2006 American Control Conference :6 pp. (2006). https://doi.org/10.1109/acc.2006.1657601
2. Zhang H, Lennox B. Fault detection and isolation in a fed-batch penicillin fermentation process. 2003 European Control Conference (ECC) :1670-1675 (2003). https://doi.org/10.23919/ecc.2003.7085204
3. Bicciato S, Bagno A, Soldà M, Manfredini R, Di Bello C. Fermentation diagnosis by multivariate statistical analysis. ABAB 102-103:049-062 (2002). https://doi.org/10.1385/abab:102-103:1-6:049
4. Tates AA, Louwerse DJ, Smilde AK, Koot GLM, Berndt H. Monitoring a PVC batch process with multivariate statistical process control charts. Ind. Eng. Chem. Res. 38:4769-4776 (1999). https://doi.org/10.1021/ie9901067
5. Li JY, Ambikasaran S, Darve EF, Kitanidis PK. A kalman filter powered by h2-matrices for quasi-continuous data assimilation problems. Water Resour. Res. 50:3734-3749 (2014). https://doi.org/10.1002/2013wr014607
6. Bakarji J. Singular Value Decomposition (SVD) and Its Relation to PCA. Riad El-Solh, Beirut 1107, Lebanon; 2025. Online lecture notes. (2020) Available from: https://www.ml4science.com/content/notes/05_svd/svd_demo
7. Chartrand R. Numerical differentiation of noisy, nonsmooth data. ISRN Applied Mathematics 2011:1-11 (2011). https://doi.org/10.5402/2011/164564
8. Cleveland WS. Robust locally weighted regression and smoothing scatterplots. Journal of the American Statistical Association 74:829-836 (2012). https://doi.org/10.1080/01621459.1979.10481038