FedMSPC: A Federated Multivariate Statistical Process Control Framework For Privacy-Preserving Process Modeling Across Company Borders

Du Nguyen Duy, David Gabauer, and Ramin Nikzad-Langerodi

Abstract The ongoing transition from a linear (produce-use-dispose) to a circular economy poses significant challenges to current state-of-the-art information and communication technologies. In particular, the derivation of integrated, high-level views on material, process, and product streams from (real-time) data produced along value chains is challenging for several reasons. Most importantly, sufficiently rich data is often available yet not shared across company borders because of privacy concerns which make it impossible to build integrated process models that capture the interrelations between input materials, process parameters, and key performance indicators along value chains. In the current contribution, we propose a privacy-preserving, federated multivariate statistical process control (FedMSPC) framework based on Federated Principal Component Analysis (PCA) and Secure Aggregation to foster the incentive for closer collaboration of stakeholders along value chains. We tested our approach on two industrial benchmark data sets - SECOM and ST-AWFD. Our empirical results demonstrate the superior fault detection capability of the proposed approach compared to standard, single-party (Multiway) PCA. Furthermore, we showcase the possibility of our framework to provide privacy-preserving fault diagnosis to each data holder in the value chain to underpin the benefits of secure data sharing and federated process modeling.

Keywords Multivariate statistical process control · Federated learning · Privacy-preserving machine learning · Circular economy

D. Nguyen Duy · D. Gabauer · R. Nikzad-Langerodi (✉)
Software Competence Center Hagenberg, Hagenberg, Austria
e-mail: ramin.nikzad-langerodi@scch.at

D. Nguyen Duy
e-mail: du.nguyen.duy@scch.at

D. Gabauer
e-mail: david.gabauer@scch.at

1   Introduction

The importance of data exchange along value chains has been broadly recognized for mastering the transition from linear to circular economy [1, 2]. However, as individuals and corporations are increasingly concerned about how their data are being used, the emphasis on data privacy and security has become a major global topic. There are now data protection obligations that organizations must strictly follow [3]. Therefore, it is challenging, if not impossible, in many situations to transfer data across company borders. This landscape poses new challenges that traditional information and communication technologies in general, and process modeling approaches in particular, cannot handle appropriately.

More specifically, traditional process modeling workflows usually involve collecting and fusing data into a common site where a data-driven model is built. However, this is no longer feasible since data are not owned by a single entity but rather generated and distributed among different companies along a value chain. On the one hand, organizations do not want to share private data because of the fear of trade secrets leaks, and on the other hand due to regulations or geographical restrictions. As a result, even though sufficiently rich process data is available, they often exist in small and fragmented silos and cannot be integrated to enable a broader view of the whole value chain. This is a major obstacle in areas where it is well known that material properties, process parameters, and KPIs are intercorrelated across company borders, for example, in the steel or paper industry [4, 5].

A functional solution to overcome the problem of data fragmentation and isolation is Federated Learning (FL), a concept first proposed by Google in 2016 [6]. The main idea behind FL is to build a centralized model based on data scattered among multiple parties without requiring participants to share sensitive information. In the meantime, FL has gained increasing attention, both from research and industry perspectives. However, a preliminary literature review shows that most of the current work is focused primarily on deep neural networks and their application to computer vision problems [7, 8].

Multivariate Statistical Process Control (MSPC) is an umbrella term for a set of advanced statistical methods for modeling, monitoring, and controlling the operating performance of processes that are widely adopted in the process industry. More specifically, MSPC techniques extract features from high-dimensional and highly correlated process data by means of latent variable (LV) based modeling techniques. The models are then used to monitor processes in real-time, assess their performance, and identify deviations from normal operating conditions (NOC). Therefore, MSPC provides a basis for increasing process security, sustainability, and continuous improvement. Although some recent research efforts have been devoted to adopting federated learning in the field of MSPC, limited progress has been made [9, 10]. Previous studies have focused on federated PCA [11]. However, the application of PCA to MSPC, i.e., for online monitoring, fault detection, and fault diagnosis yet to be proposed. In addition, Multiway PCA (MPCA), an extension of PCA for modeling batch process data has not been investigated, and thus, to the best of our knowledge, the FL paradigm has so far not been adopted for building MSPC-type process models across company borders that preserve the privacy of each contributing party. Moreover, the incentive mechanism,

an essential aspect of FL, has not been discussed in the existing literature. A fair value-distribution structure is critical in order to motivate the different parties to actively collaborate in the model training and inference process [12]. The collaboration might be wasteful without meaningful incentives because the participating parties will not carry out efficient contributions.

In this work, we propose a general federated multivariate statistical process modeling framework (FedMSPC) where different companies along a value chain can together build a shared process monitoring model in a federated and privacy-preserving manner. To fully protect confidential data, FedMSPC uses a combination of two privacy techniques: Data Masking and Secure Aggregation [11]. In particular, each participant unfolds and encrypts data in the local environment using a well-designed masking method. Then, all encrypted data are transferred to a third-party server, which securely aggregates these data and trains an MPCA model on the concatenated joint (encrypted) data matrix. Finally, each participant decrypts the federated output of the model, using its private key, and reconstructs the information related to their process. More specifically, each participant obtains only the portion of the shared loadings matrix that corresponds to the variables that they contribute. Thus, this information is not accessible to the other parties. In addition, all participants will share the explained variance corresponding to the selected principal components (PCs). Using such results, all data holders can collaboratively estimate the scores, process-monitoring statistics (e.g. Hotelling's T^2 and Q-statistic) as well as variable contributions, thereupon conduct fault detection and diagnosis.

In order to showcase the feasibility of the framework, we propose Federated Principal Component Analysis (FedPCA), which is based on the idea of Federated Singular Value Decomposition (FedSVD) proposed in [11], as the modeling method. FedSVD basically provides lossless privacy guarantees and is thus ideally suited for building federated MSPC models. However, in [11], the authors employ FedPCA under a horizontally partitioned scenario, where the data from the contributing parties share the same feature rather than the sample space (i.e. horizontal FL). However, in value chains, input materials are processed sequentially by different companies and the corresponding data is thus vertically partitioned, i.e., the data share the same sample space but different feature spaces. Therefore, we will concentrate on vertical FedPCA instead. In addition, we will investigate the application of FedPCA in modeling batch processes.

2 Methodology

2.1 Multivariate Statistical Process Control

PCA is among the best-known and most widely adopted MSPC techniques for modeling high-dimensional and highly correlated process data, and will be introduced in the following.

Principal Component Analysis (PCA) is often applied in MSPC to transform a dataset with highly correlated variables into an uncorrelated dataset while preserving only the

systematic variation. There are various techniques for building a PCA model, Singular Vector Decomposition (SVD) being the most popular one. Suppose the original dataset denoted by $X \in \mathbb{R}^{m \times n}$ contains m observations and n process variables. SVD corresponds to the following decomposition:

$$
\begin{aligned}
X &= U \Sigma V^T \\
&= \begin{bmatrix} U_r & U_0 \end{bmatrix} \begin{bmatrix} \Sigma_r & 0 \\ 0 & \Sigma_0 \end{bmatrix} \begin{bmatrix} V_r & V_0 \end{bmatrix}^T ,
\end{aligned}
\tag{1}
$$

where $U \in \mathbb{R}^{m \times m}$ is the left singular matrix, $\Sigma \in \mathbb{R}^{m \times n}$ is the diagonal matrix holding the singular values, and $V \in \mathbb{R}^{n \times n}$ is the right singular matrix. $V_r \in \mathbb{R}^{n \times r}$ derived from V is called the loadings matrix. The number of principal components r can be determined based on a certain criterion, e.g., the cumulative explained variance, and usually it holds that $r \ll n$. The loadings are the coefficients of the variables from which the PCs are computed. The sign of the loading shows whether the correlation between the PCs and the corresponding variable is positive or negative while its absolute value indicates how strongly the variable influences the PCs. Therefore, they are often used to quantify variable importance.

Projecting X onto the subspace spanned by the selected PCs reduces the dimensionality of the column space from n to r. The result of the transformation is the scores matrix $T_r \in \mathbb{R}^{m \times r}$

$$
T_r = X V_r .
\tag{2}
$$

The reconstruction of X can be estimated from T_r and V_r:

$$
\hat{X} = T_r V_r^T
\tag{3}
$$

The residual matrix E is defined as the errors between X and $\hat{X}$ and can be calculated as

$$
E = X - \hat{X} = X - X V_r V_r^T .
\tag{4}
$$

Multi-way principal component analysis (MPCA) is an extension of PCA for monitoring batch processes that are broadly seen in industries where batch and semi-batch process operations are common, for example, in the chemical or pharmaceutical industry [13, 14].

Suppose each batch run has J variables measured at K time intervals. Similar data exist in the same form for each of the I batch runs. Therefore, all the data can be arranged in an array $X(I \times J \times K)$.

In order to apply PCA, the dataset has to be converted into a two-dimensional array. There are multiple ways to unfold a 3D dataset. However, the most meaningful approaches are batch-wise and variable-wise unfolding. In the proposed approach, we will employ batch-wise unfolding since variable-wise unfolding is not feasible for federated MSPC. In batch-wise unfolding, the 2D array is formed by unfolding the array X so that each of its vertical slices contains the observed variables for all batches at a
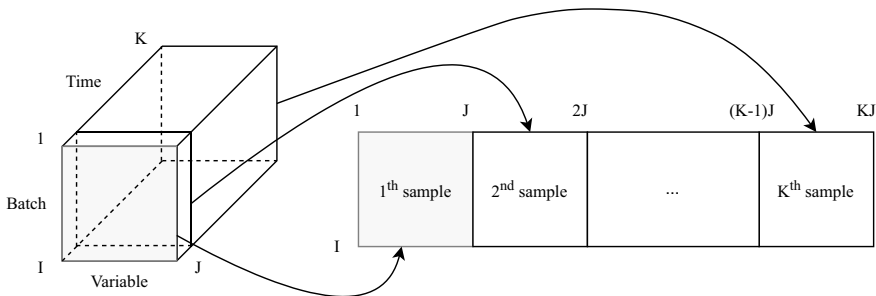
Fig. 1 An illustration of the batch-wise unfolding method

specific time instance. The result is a 2D matrix of the shape $(I \times KJ)$. An illustration of this unfolding technique is shown in Fig. 1.

After the data is unfolded, PCA is performed to retrieve the scores and loadings matrices. Hotelling's T^2, Q-statistics, and the corresponding variable contributions can then be calculated in a similar manner as for PCA.

From an operational point of view, it is preferable to monitor the batch as it progresses in order to anticipate process faults and to take timely actions to prevent out-of-batch-specification events. However, in this situation, a major obstacle is that the new batch is required to have KJ columns such as the NOC data used for training the model. This is impossible when the batch has not been completed because at time interval $k < K$ the new batch x only has kJ columns. In [14], the authors proposed a simple solution to overcome this problem which is to use only the portion of the loadings matrix that corresponds to the elapsed time period until the current time interval k to calculate the new scores vector

$$t_r^{[k]} = x\tilde{V}_r(\tilde{V}_r^T\tilde{V}_r)^{-1}, \tag{5}$$

where $\tilde{V}_r = V_r[1 : kJ]$ contains the first kJ columns of V_r. In the proposed approach, we will use this same approach to handle the situation where batches are incomplete.

Fault detection and diagnosis One of the most popular applications of PCA-based MSPC is fault detection and diagnosis. This is often done based on control chart statistics, such as Hotelling's T^2 and the so-called Q-statistic. While a high Q-statistic indicates a change in covariance structure, a high T^2 indicates that although the sample is described well by the model (i.e. through a linear combination of the loadings) it is unusual in terms of the linear combination. Suppose there is a PCA model generated from NOC data. When a new sample $x \in \mathbb{R}^{1 \times n}$ arrives, T^2 and Q can be calculated as:

$$T^2 = t_r\Lambda_r^{-1}t_r^T, \tag{6}$$

where $t_r = xV_r$ and $\Lambda_r = \Sigma_r^2$ and

$$Q = \sum_{i=1}^{n}(x_i - \hat{x}_i)^2. \tag{7}$$

The upper confidence limit for the T^2 statistic can be computed from the F-distribution

$$T_\alpha^2 = \frac{r(m-1)}{r-m} F_{r,m-r,\alpha}, \tag{8}$$

where m and r denote the number of NOC samples and the number of PCs, respectively. The upper confidence limit for the Q-statistic can be computed from the approximate distribution

$$
\begin{aligned}
Q_\alpha &= \theta_1 \left(1 - \frac{\theta_2 h_0 (1 - h_0)}{\theta_1^2} + \frac{z_\alpha \sqrt{2\theta_2 h_0^2}}{\theta_1} \right)^{1/h_0} \\
\theta_i &= \sum_{j=r+1}^{l} (\mathbf{\Lambda}_{j,j})^i, \, i = 1, 2, 3 \\
h_0 &= 1 - \frac{2\theta_1 \theta_3}{3\theta_2^2},
\end{aligned}
\tag{9}
$$

where z_α is the standard normal deviate corresponding to the upper $(1 - \alpha)$ percentile, $\mathbf{\Lambda}_{jj}$ is the eigenvalue associated with the jth loading vector, and l is the number of non-zero eigenvalues calculated from the data.

A sample might be considered faulty if either T^2 or Q exceeds the predefined control limits T_α^2 or Q_α. When a fault is detected, variable contributions to T^2 and Q can be calculated as [14]

$$T_{cont}^2 = x \mathbf{V}_r \mathbf{\Sigma}^{-1} \mathbf{V}_r^T = t_r \mathbf{\Sigma}^{-1} \mathbf{V}_r^T \tag{10}$$

and

$$Q_{cont} = (x - \hat{x})^2. \tag{11}$$

Variables with high contributions are diagnosed as candidates for the cause of the fault.

2.2 Federated Multivariate Statistical Process Control (FedMSPC)

Figure 2 illustrates the proposed framework. Assume we have g data holders with the i-th data holder owning data matrix $\mathbf{X}_i \in \mathbb{R}^{m \times n_i}$ and these data holders aim at fitting a PCA model on the concatenated matrix $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_g]$, where $\mathbf{X} \in \mathbb{R}^{m \times n}$ and $n = \sum_{i=1}^{g} n_i$. In this case, the full results of PCA include $\mathbf{\Sigma} \in \mathbb{R}^{m \times n}$ and $\mathbf{V}^T = [\mathbf{V}_1^T, ..., \mathbf{V}_g^T] \in \mathbb{R}^{n \times n}$, where $\mathbf{V}_i^T \in \mathbb{R}^{n \times n_i}$ is the portion of the loadings matrix corresponding to the variables contributed by the i-th data holder.

The aim of this contribution is to design a privacy-preserving system that guarantees that (1) during the computation, data is not leaked to any other data holder and (2)

the loadings matrix V must be vertically and secretly distributed among data holders. While the first requirement is commonly shared by all privacy-preserving applications, the second requirement is more specific to federated MSPC. As explained in the previous chapter, loadings can reveal sensitive information about how variables interact with the principal components and can be used to calculate the contributions of variables to T^2 and Q statistics. Therefore, data holder i should know only $V_i \in \mathbb{R}^{n_i \times n}$ that contains coefficients corresponding to its contributed variables. Furthermore, V_i has to be unknown to all other involved parties. In order to realize these goals, we propose an approach based on FedSVD [11].

To apply FedSVD-based PCA as proposed in [11], the system requires a Trusted Authority (TA) to handle key generation, and a Computation Service Provider (CSP) to take care of data aggregation and model building. Algorithm 1 shows the overall model-building workflow. Since Σ, the number of NOC samples, the number of variables, and the number of principal components are shared by all data holders, the control limits for Hotelling's T^2 and Q-statistic can be estimated using Eqs. 6 and 7 as for standard PCA.

After a FedPCA model is built, suppose a new sample $x = [x_1, ..., x_g]$, where $x_i \in \mathbb{R}^{1 \times n_i}$, is generated. Algorithm 2 is used to calculate scores, Hotelling's T^2 and Q-statistics, and the contribution of each variable to these quantities. Using these monitoring values, all data holders can check whether the sample/batch is faulty or not, i.e. if the monitoring statistics lie above their critical limits. However, since V_i is secretly owned by each data holder, the computation of Hotelling's T^2- and Q-contributions can only be done locally. This might be greatly beneficial since all participating companies might know that a sample/batch is faulty, and they can check whether the problem might be associated with their production line. However, each company only sees the contri-
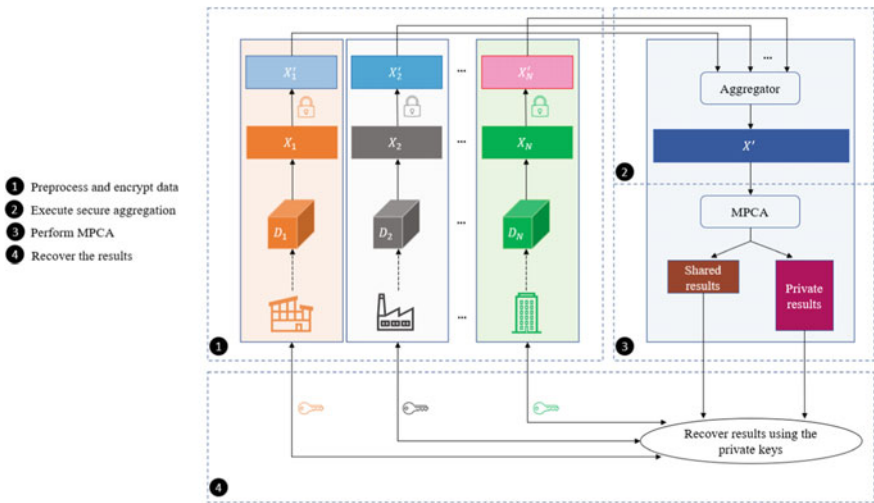

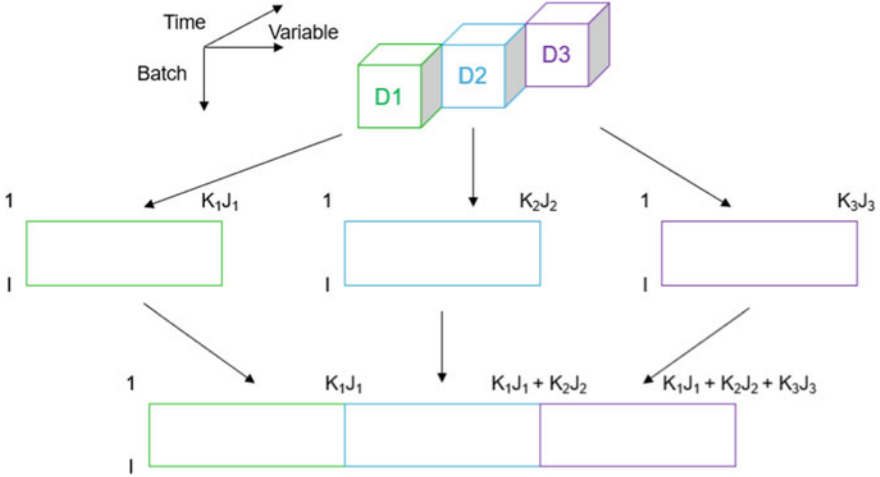**Fig. 2** The architecture of the federated process modeling framework

Fig. 3 An illustration of how local data is mapped batch-wise in FedMPCA

bution of its own variables to the fault, which provides feedback on how to improve its own process to benefit the entire value chain.

For batch process data, FedPCA cannot be applied directly. Thus, we propose an extension called FedMPCA that includes a data unfolding step before encryption and transfer to the CSP. However, unlike the transition from PCA to MPCA, not all unfolding techniques are applicable in the federated scenario. While variable-wise unfolding is invalid because local data do not share the same feature space, batch-wise unfolding is undertaken as shown in Fig. 3.

Suppose there are g participating data holders and data holder i owns a batch data set of shape $I \times J_i \times K_i$ where I, J_i, and K_i are the number of batches, the number of variables, and the number of time intervals, respectively. When batch-wise unfolding is employed, the unfolded data of data holder i is $X_i \in \mathbb{R}^{m \times n_i}$ where $m = I$ and $n_i = K_i J_i$. Therefore, the joined data $X = [X_1, \dots, X_g] \in \mathbb{R}^{m \times n}$, where $n = \sum_{i=1}^{g} n_i = \sum_{i=1}^{g} K_i J_i$. The full results of MPCA consist of $\Sigma \in \mathbb{R}^{m \times n}$ and $V^T = [V_1^T, \dots, V_g^T] \in \mathbb{R}^{n \times n}$, where $V_i^T \in \mathbb{R}^{n \times n_i}$ is the portion of the loadings matrix corresponding to the (unfolded) variables contributed by data holder i. Similar to FedPCA, at the end of FedMPCA model training, data holder i receives Σ and V_i^T and can cooperate with the other data holders, the TA, and the CSP to calculate scores and the monitoring indexes.

In an online monitoring scenario, given all the preceding data holders have completed their processes, suppose the i-th data holder wants to calculate the monitoring indexes and variable contributions for an in-progress batch at time interval $k < K$. The corresponding data can be expressed as $x = [x_1, \dots, x_i]$. The procedure is described in Algorithm 3. Once the scores are obtained, data holders can calculate Hotelling's T^2,

Q-statistics and contribution of variables to the two indexes in a similar manner as it is done for completed batches (described in lines 16-37 of Algorithm 2).

Algorithm 1: FedPCA Training

Input: $X = [X_1, \ldots, X_g]$

Output: Σ, r and $V^T = [V_1^T, \ldots, V_g^T]$

Constraint: Data holder i's data is not leaked, and it receives Σ, r and V_i^T as results.

1 **Function** FedPCA.train($[X_1, \ldots, X_g]$)

2   **TA** do:

3   Generate orthogonal matrices $P \in \mathbb{R}^{m \times m}$, $B \in \mathbb{R}^{n \times n}$.

4   Then split B^T into $[B_1^T, \ldots, B_g^T]$ where $B_i^T \in \mathbb{R}^{n \times n_i}$.

5   **end**

6   **Data Holders** do:

7   **for** $i = 1 \to g$, *Data holder i* **do**

8   Download P, B_i^T from TA and compute

9        $X_i^{'} = P X_i B_i$

10   **end**

11   **end**

12   **CSP** do:

13   Aggregate $X^{'}$:

14      $X^{'} = \sum_{i=1}^{g} X_i^{'} \quad \left(= \sum_{i=1}^{g} P X_i B_i = P X B \right)$

15   Perform standard SVD:

16      $X^{'} = U^{'} \Sigma V^{'T}$

17   **end**

18   **Data Holders** do:

19   **for** $i = 1 \to g$, *Data holder i* **do**

20   Download Σ from CSP.

21   Determine the number of principal components r based on the cumulative sum of explained variance.

22   Generate a random matrix $R_i \in \mathbb{R}^{n_i \times n_i}$

23   Mask B_i^T through: $[B_i^T]^{R_i} = B_i^T R_i$

24   Send $[B_i^T]^{R_i}$ to CSP.

25   **end**

26   **end**

27   **CSP** wait to receive data and do:

28   **if** *Receive $[B_i^T]^{R_i}$ then* **then**

29   Compute $[V_i^T]^{R_i} = V^{'T} [B_i^T]^{R_i} \quad (= V^{'T} B_i^T R_i = V_i^T R_i)$

30   Send $[V_i^T]^{R_i}$ back to data holder i.

31   **end**

32   **end**

33   **Data Holders** do:

34   **for** $i = 1 \to g$, *Data holder i* **do**

35   Receive $[V_i^T]^{R_i}$ from CSP.

36   Recover V_i^T by $V_i^T = [V_i^T]^{R_i} R_i^{-1}$.

37   **end**

38   **end**

39 **End**

Algorithm 2: FedPCA Inference

Input: $x = [x_1, ..., x_g]$

Output: t_r, Q-statistics, Hotelling's T^2, $Q_{cont} = [Q_{cont,1}, ..., Q_{cont,g}]$,
$\quad\quad T_{cont}^2 = [T_{cont,1}^2, ..., T_{cont,g}^2]$

Constraint: Data holder i's data is not leaked, and it receives t_r, Q-statistics, Hotelling's T^2,
$\quad\quad Q_{cont,i}$, $T_{cont,i}^2$ as results.

1   **Function** FedPCA.predict($[x_1, ..., x_g]$)

2   **TA** do:

3   Generate a random number p.

4   **end**

5   **Data Holders** do:

6   **for** $i = 1 \rightarrow g$, *Data holder i* **do**

7   Download the random number p from TA.

8   Calculate local scores: $t_{r,i} = x_i \boldsymbol{V}_{r,i}$

9   Encrypt the local scores with p: $t_{r,i}^{'} = pt_{r,i} = px_i \boldsymbol{V}_{r,i}$

10   Send $t_{r,i}^{'}$ to CSP.

11   **end**

12   **end**

13   **CSP** do:

14   Aggregate $t_r^{'}$: $t_r^{'} = \sum_{i=1}^{g} t_{r,i}^{'}$ $\left(= p\sum_{i=1}^{g} x_i \boldsymbol{V}_{r,i} = px\boldsymbol{V}_r\right)$

15   **end**

16   **Data Holders** do:

17   **for** $i = 1 \rightarrow g$, *Data holder i* **do**

18   Download $t_r^{'}$ from CSP.

19   Recover the scores t_r: $t_r = \frac{t_r^{'}}{p}$

20   Calculate T^2: $T^2 = t_r \boldsymbol{\Sigma}^{-2} t_r^T$

21   Calculate $T_{cont,i}^2 = t_r \boldsymbol{\Sigma}^{-1} \boldsymbol{V}_{r,i}^T$

22   Calculate local reconstruction errors e_i: $e_i = x_i - t_r \boldsymbol{V}_{r,i}^T$

23   Calculate $Q_{cont,i}$: $Q_{cont,i} = e_i^2$

24   Calculate local Q-statistics: $Q_i = e_i e_i^T$

25   Encrypt Q_i using p: $Q_i^{'} = pQ_i$

26   Send $Q_i^{'}$ to CSP.

27   **end**

28   **end**

29   **CSP** do:

30   Aggregate $Q^{'}$: $Q^{'} = \sum_{i=1}^{g} Q_i^{'}$ $\left(= p\sum_{i=1}^{g} Q_i = pQ\right)$

31   **end**

32   **Data Holders** do:

33   **for** $i = 1 \rightarrow g$, *Data holder i* **do**

34   Downloads $Q^{'}$ from CSP.

35   Recover Q by $Q = \frac{Q^{'}}{p}$

36   **end**

37   **end**

38 **End**

Algorithm 3: FedMPCA Incomplete Batch

Input: $x = [x_1, ..., x_i]$
Output: $t_r^{[k]}$
Constraint: Data holder i's data is not leaked, and it receives $t_r^{[k]}$ as the result.

1 **Function** `FedMPCA.predict([x_1, ..., x_i])`
2   **TA** do:
3   Generate a random number p and an random matrix $\boldsymbol{W} \in \mathbb{R}^{r \times r}$
4   **end**
5   **Data Holders** do:
6   **for** $j = 1 \rightarrow i$, *Data holder j* **do**
7   Download the p and $\boldsymbol{W}$ from TA.
8   Calculate $t'_{r,j}$ and $\boldsymbol{F}'_j$ as follows:
9        $t'_{r,j} = px_j \tilde{\boldsymbol{V}}_{r,j} \boldsymbol{W}$
10        $\boldsymbol{F}'_j = \tilde{\boldsymbol{V}}_{r,j}^T \tilde{\boldsymbol{V}}_{r,j} \boldsymbol{W}$
11   where $\tilde{\boldsymbol{V}}_{r,j} = \boldsymbol{V}_{r,j}$ for $j < i$, and $\tilde{\boldsymbol{V}}_{r,j} = \boldsymbol{V}_{r,j}[1 : kJ_j]$ when $j = i$. Note that in this case, $\tilde{\boldsymbol{V}}_r^T = [\tilde{\boldsymbol{V}}_{r,1}^T, ..., \tilde{\boldsymbol{V}}_{r,i}^T]$.
12   Send $t'_{r,j}$ and $\boldsymbol{F}'_j$ to CSP.
13   **end**
14   **end**
15   **CSP** do:
16   Calculates $t_r^{[k]'}$:

$$t_r^{[k]'} = \sum_{j=1}^{i} px_j \tilde{\boldsymbol{V}}_{r,j} \boldsymbol{W} (\sum_{j=1}^{i} \tilde{\boldsymbol{V}}_{r,j}^T \tilde{\boldsymbol{V}}_{r,j} \boldsymbol{W})^{-1}$$

$$\left(= p \sum_{j=1}^{i} x_j \tilde{\boldsymbol{V}}_{r,j} \boldsymbol{W} \boldsymbol{W}^{-1} (\sum_{j=1}^{i} \tilde{\boldsymbol{V}}_{r,j}^T \tilde{\boldsymbol{V}}_{r,j})^{-1} \right)$$

$$\left(= px \tilde{\boldsymbol{V}}_r (\tilde{\boldsymbol{V}}_r^T \tilde{\boldsymbol{V}}_r)^{-1} \right)$$

$$\left(= pt_r^{[k]} \right)$$

     Broadcast $t_r^{[k]'}$ to all data holders.
17   **end**
18   **Data Holders** do:
19   **for** $j = 1 \rightarrow i$, *Data holder j* **do**
20   Recover the real scores using p:
21        $t_r^{[k]} = \frac{t_r^{[k]'}}{p}$
22   **end**
23   **end**
24 **End**

---

3 Experiments

As proof of concept, we applied FedPCA and FedMPCA to two industrial case studies from semiconductor manufacturing. The corresponding datasets SECOM[1] [16] and ST-AWFD[2] [17] have been published previously and are in the public domain.

3.1 General Settings

For both case studies, we first divided the data (X) variable-wise into two subsets (X_1 and X_2) corresponding to different process steps and assigned these to two (hypothetical) data holders. Subsequently, each subset was further split into a training, a validation, and a test set, i.e. $X_1 = \{X_1^{train}, X_1^{val}, X_1^{test}\}$ and $X_2 = \{X_2^{train}, X_2^{val}, X_2^{test}\}$. The partition was done in a way that the training set only contained NOC batches and the validation and test sets consisted of both NOC and faulty batches. The training set was used for training the model. The validation set was used to set control limits for Hotelling's T^2 and Q-statistic. By means of a grid search, we selected the lowest thresholds with the highest F1 score on the validation set as the control limits. The test set was utilized to evaluate the model performance. Four models were built to simulate three common real-world situations:

- Situation 1: One company has access to all data X and can use that data to build a fault detection model based on PCA (MPCA).
- Situation 2: Each company only has access to its data and can use that data to build a local fault detection model to detect faults that occurred in their production line. In this case, Company 1 owns the PCA1 (MPCA1), and Company 2 owns the PCA2 (MPCA2) model. If a fault is detected by one of the models, it is considered to be detected.
- Situation 3: Each company only has access to its data; however, the two companies cooperate to build a federated fault detection model based on FedPCA (FedMPCA).

Table 1 shows the training and test set used for each model. The number of principal components was chosen such that the cumulative sum of explained variance was ≥90%.

We evaluated the models in terms of the effectiveness to detect faulty batches and also fault diagnosis on the test set. A batch was considered faulty when either the T^2 or the Q-statistic exceeded the predefined control limits. The effectiveness was evaluated by the F1 score, and the fault diagnosis was assessed based on Hotelling's T^2- and Q-contribution plots.

---

[1] https://archive.ics.uci.edu/ml/datasets/SECOM.

[2] https://github.com/STMicroelectronics/ST-AWFD.

Table 1 Training set and test set for each model

Model	Training data	Validation data	Test data
PCA1 (MPCA1)	X_1^{train}	X_1^{val}	X_1^{test}
PCA2 (MPCA2)	X_2^{train}	X_2^{val}	X_2^{test}
PCA (MPCA)	$\{X_1^{train}, X_2^{train}\}$	$\{X_1^{val}, X_2^{val}\}$	$\{X_1^{test}, X_2^{test}\}$
FedPCA (FedMPCA)	X_1^{train}, X_2^{train}	X_1^{val}, X_2^{val}	X_1^{test}, X_2^{test}

Table 2 List of variables that belong to each data holder

Dataset	No. Variables	Variable name
X_1	21	$S_{15}, S_{27}, S_{33}, S_{36}, S_{48}, S_{60}, S_{62}, S_{64}, S_{118}, S_{122}, S_{124}, S_{125}, S_{131}, S_{134}, S_{145}, S_{153}, S_{184}, S_{201}, S_{206}, S_{288}, S_{342}$
X_2	17	$S_{421}, S_{426}, S_{427}, S_{430}, S_{435}, S_{454}, S_{461}, S_{470}, S_{478}, S_{492}, S_{511}, S_{520}, S_{525}, S_{560}, S_{569}, S_{572}, S_{574}$

Table 3 Summary of training and test set

Dataset	No. NOC samples	No. faulty samples	No. features
X_1^{train}	488	0	21
X_1^{test}	74	48	21
X_2^{train}	488	0	17
X_2^{test}	74	48	17

3.2 Case Study 1: SECOM Dataset

Data description. SECOM is a static dataset consisting of 1567 observations, each with 590 variables (S_1 to S_{590}) and one label for the quality test (–1 means the observation is normal, and 1 indicates the observation is faulty). As with any real-world dataset, SECOM contains missing values and irrelevant variables. As the focus of this case study is fault detection and diagnosis, we considered the 38 variables recommended by [15] and dropped all instances that contain missing values. According to [15], the selected variables can be divided into five workstations based on the property of the semiconductor manufacturing monitoring process. In this experiment, we assumed that X_1 and X_2 include all parameters of the first three and the last two workstations, respectively. Tables 2 and 3 show a summary of each dataset.

In this experiment, the optional validation set was not used and the control limits for T^2 and Q-statistic set according to Eqs. 6 and 7, respectively.

Table 4 Model performance on SECOM data

Model	TP	TN	FP	FN	F1 score
PCA	19	55	19	29	0.44
FedPCA	19	55	19	29	0.44
PCA1 + PCA2	14	53	21	34	0.33

Evaluation. The performance of all evaluated models is shown in Table 4. PCA and FedPCA show the same performance. This is expected because, in [11], the authors prove that FedSVD is a lossless method that produces the same results as standard SVD. Notably, FedPCA outperforms PCA1, PCA2, and their combination by achieving a higher F1 score which underpins the benefit of integrated versus local process models.

To evaluate the capability of our approach to diagnose faults, we used the trained models to generate contribution plots for Hotelling's T^2 and Q-statistic. Figures 4, 5, and 6 show contribution plots of some selected faults. In each of these figures, there are two plots. The upper plot shows the contribution of each input variable calculated by PCA1 and PCA2 (i.e. the local models). The lower one shows the variable contributions calculated by FedPCA. In the middle of each plot, there is a vertical dashed line representing the (hypothetical) company border. Note that each company can only reconstruct the contributions corresponding to the variables that they own using their private data (x_i), private loadings matrix (V_i), and the shared matrix Σ.

Figures 4 and 5 show two examples of faulty products that were detected by both FedPCA and the combination PCA1 + PCA2. In Fig. 4, it can be seen that for Sample 1, the two plots are quite similar and variables of Company 1 show a much higher contribution to the fault than those of Company 2 indicating that the fault is caused predominantly by the latter. Variables #3 and #4 are reasonable candidates for further root cause analysis. In contrast, in Fig. 5 FedPCA and the local models disagree in terms of the variables that contribute to the fault. Whereas the local models suggest a significant contribution from both parties (with variables from company 2 in fact showing higher overall contributions), the FedPCA model indicates that the fault is mostly associated with company 1. While Variable #14 shows the highest impact according to PCA1 and FedPCA, the most contributed variable according to PCA2 is Variable #32.

Figure 6 demonstrates an example where FedPCA detected a faulty sample that passed both PCA1 and PCA2. Even though the Q-statistic calculated by PCA1 is high, it doesn't surpass the control limit. For PCA2, it is clear that the Q-statistic is small. This example represents cases where the problem is caused by not only the process parameters of one data holder but by a combination/interaction of process parameters across the (hypothetical) company border. An advantage of FedPCA, in this case, is that it can make use of all available data to increase performance, and at the same time, the root cause (sensitive information) is known by the corresponding data holder only, and hidden to the other parties. While the second company can reliably claim the problem did not come from their process production and request the first company to do a checkup. It won't know the exact contribution of input variables of the first company.
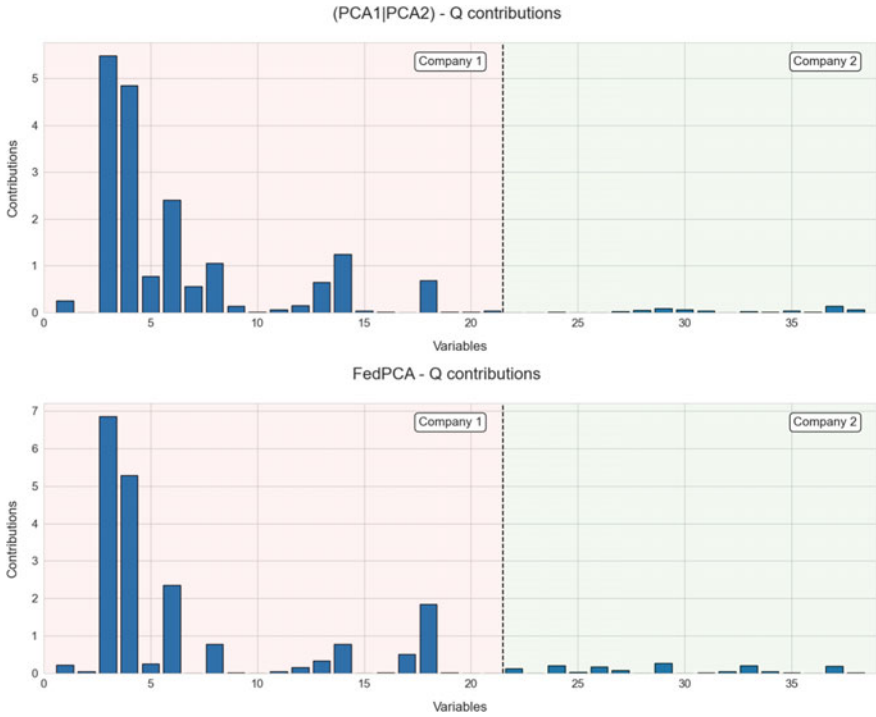
Fig. 4 *Q* contribution plots generated for sample 1

On the other hand, the first company can use the contribution plots together with their know-how to optimize their machine settings to reduce out-of-specification events later in the value chain.

3.3   Case study 2: ST-AWFD Dataset

Data description. ST-AWFD is a batch dataset that contains a total of 1156 batches with 20 variables and an average of 100 samples per batch. Each batch is labeled as normal or faulty through a temporal reference window. The production process is divided into two steps called Step 1 and Step 2. Depending on the batch, the length of each step might differ.

In order to apply batch-wise MPCA and FedMPCA, all the batches must have the same length. Therefore, in this experiment, we have only selected batches with lengths of 110 consisting of 65 observations for Step 1 and 45 observations for Step 2. After the data cleaning phase, there are 966 batches left which include 648 NOC batches and 318 faulty batches. A summary of the data partition is shown in Table 5.
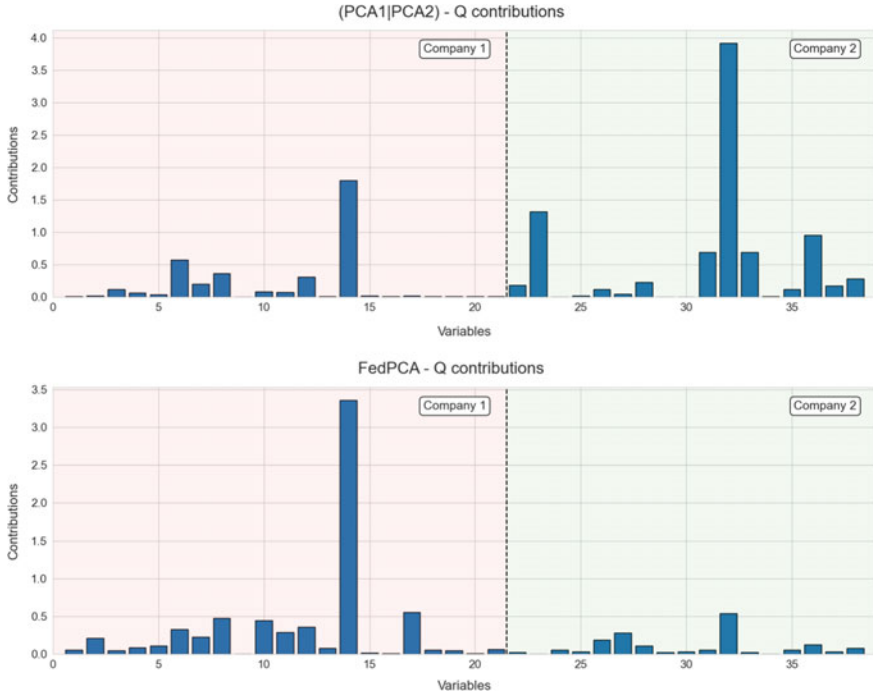
Fig. 5 Q contribution plots generated for sample 2

Table 5 A summary of training, validation, and test set used in the experiment

Dataset	No. NOC samples	No. faulty samples	No. features	No. time intervals
X_1^{train}	482	0	20	65
X_1^{val}	83	159	20	65
X_1^{test}	83	159	20	65
X_2^{train}	482	0	20	45
X_2^{val}	83	159	20	45
X_2^{test}	83	159	20	45

The control limit for the T^2 statistic was calculated by Eq. 6. For the Q-statistic, initially, Eq. 7 was used for determining the confidence limit. However, we found that the returned threshold was too conservative and led to a high number of false positives for all models. Therefore, the control limit for the Q-statistic was obtained by cross-validation using the training and validation set (described in 3.1) instead.

Evaluation The performance of all evaluated models is shown in Table 6. Similar to case study 1, MPCA and FedMPCA returned the same performance and outperformed the combination of MPCA1 and MPCA2. In this experiment, the contribution plots
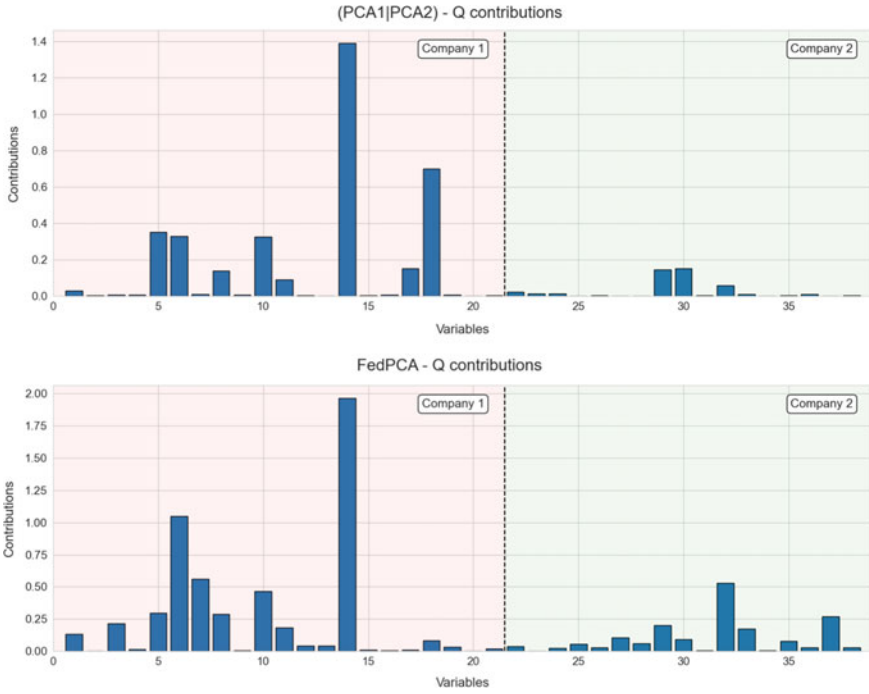
Fig. 6 *Q* contribution plots generated for sample 3

Table 6 Model performance on ST-AWFD data

Model	TP	TN	FP	FN	F1 score
MPCA	159	83	0	0	1
FedMPCA	159	83	0	0	1
MPCA1 + MPCA2	159	74	9	0	0.97

were also generated for faulty batches. However, because the number of variables is large, it is difficult to judge the difference between the plots without further process knowledge (results not shown).

Altogether, our results on the two case studies underpin the benefits of federated, PCA-based process modeling in terms of better fault detection performance and more informative fault diagnosis that takes into account the interactions between process parameters across (hypothetical) company borders.

4 Conclusion

In the present work, we proposed a framework for enabling privacy-preserving, federated multivariate statistical process control (FedMSPC) of process chains involving multiple consecutive process steps operated by different companies. In particular, we have employed federated PCA following secure aggregation (vertical concatenation) of batch-wise unfolded (and encrypted) datasets from the participating parties and demonstrated the application of such models for federated fault detection and privacy-preserving fault diagnosis. To the best of our knowledge, this is the first study that proposes a solution to this problem. Importantly, our approach provides new incentives for, and underpins the benefits of, closer collaboration of stakeholders along value chains. In order to exploit the full potential of federated MSPC, future work will be devoted to the development of approaches to (i) cope with unequal batch lengths, (ii) derive federated MSPC models with dependent variables, and (iii) enable process control.

References

1. Pagoropoulos A, Pigosso DC, McAloone TC (2017) The emergent role of digital technologies in the circular economy: a review. Proc CIRP 64:19–24
2. Pennekamp J, Henze M, Schmidt S, Niemietz P, Fey M, Trauth D, Bergs T, Brecher C, Wehrle K (2019) Dataflow challenges in an internet of production: a security and privacy perspective. In: Proceedings of the ACM workshop on cyber-physical systems security and privacy, pp. 27–38
3. Voigt P, Bussche A (2017) The EU general data protection regulation (GDPR): a practical guide. https://doi.org/10.1007/978-3-319-57959-7
4. Winning M, Calzadilla A, Bleischwitz R, Nechifor V (2017) Towards a circular economy: insights based on the development of the global ENGAGE-materials model and evidence for the iron and steel industry. Int Econ Econ Policy 14. https://doi.org/10.1007/s10368-017-0385-3
5. Molina-Sánchez E, Leyva-Díaz JC, Cortés-García FJ, Molina-Moreno V (2018) Proposal of sustainability indicators for the waste management from the paper industry within the circular economy model. Water 10:1014. https://doi.org/10.3390/w10081014
6. Konečný J, McMahan HB, Ramage D, Richtárik P (2016) Federated optimization: distributed machine learning for on-device intelligence. arXiv:1610.02527
7. Li L, Yuxi F, Mike T, Kuo-Yi L (2020) A review of applications in federated learning. Comput Ind Eng 149:106854. ISSN 0360-8352. https://doi.org/10.1016/j.cie.2020.106854
8. Qinbin L, Bingsheng H, Dawn S (2021) Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 10713–10722

9. Hartebrodt A, Nasirigerdeh R, Blumenthal DB, Röttger R (2021) Federated principal component analysis for genome-wide association studies. In: 2021 IEEE international conference on data mining (ICDM). IEEE, pp 1090–1095
10. Grammenos A, Mendoza Smith R, Crowcroft J, Mascolo C (2020) Federated principal component analysis. Adv Neural Inf Process Syst 33:6453–6464
11. Chai D, Wang L, Fu L, Zhang J, Chen K, Yang Q (2021) Federated singular vector decomposition. arXiv:2105.08925
12. Yang Q, Liu Y, Chen T, Tong Y (2019) Federated machine learning: concept and applications. ACM Trans Intell Syst Technol (TIST) 10(2):1–19
13. Nomikos P, MacGregor JF (1994) Monitoring batch processes using multiway principal component analysis. AIChE J 40(8):1361–1375
14. Nomikos P, MacGregor JF (1995) Multivariate SPC charts for monitoring batch processes. Technometrics 37(1):41–59
15. Arif F, Suryana N, Hussin B (2013) Cascade quality prediction method using multiple PCA+ID3 for multi-stage manufacturing system. IERI Proc. 4:201–207. https://doi.org/10.1016/j.ieri.2013.11.029
16. McCann M, Johnston A (2008) SECOM. UCI Mach Learn Repository
17. Furnari G, Vattiato F, Allegra D, Milotta FLM, Orofino A, Rizzo R, De Palo RA, Stanco F (2021) An ensembled anomaly detector for wafer fault detection. Sensors 21:5465. https://doi.org/10.3390/s21165465
18. Ji Z, Lipton ZC, Elkan C (2014) Differential privacy and machine learning: a survey and review. arxiv:1412.7584v1
19. Bonawitz K, Ivanov V, Kreuter B, Marcedone A, McMahan HB, Patel S, Ramage D, Segal A, Seth K (2016) Practical secure aggregation for federated learning on user-held data. arxiv:1611.04482v1