A Sensor Fault-Resilient Framework for Predictive Emission Monitoring Systems

Daniele Angelosante and Marco Guerriero
ABB Switzerland Ltd, Corporate Research Center

Gregorio Ciarlo and Nunzio Bonavita
ABB SpA, Industrial Automation Measurement Analytics

***Abstract*—The acronym PEMS stands for Predictive Emission Monitoring Systems and designates software analyzers able to provide a reliable and real-time estimate of emission concentrations by means of a data-driven model using real process measurements as input data. The model is built by resorting to measured process values along with true emission values from a portable Continuous Emission Monitoring System (CEMS) during the *data collection* period. Once on-line, PEMS performance in terms of emission prediction accuracy is strongly affected by the quality of the sensors input data. In order to ensure that the performance requirements imposed by the regulatory environmental agencies are met, a so-called Sensor Evaluation System (SES) must be included in the design of a new *Robust PEMS* (R-PEMS). The main goal of this paper is to introduce a technical solution that is capable of: i) detecting whether the sensors input data to the PEMS is faulty; ii) identifying which sensor is faulty; iii) whenever possible, substituting the faulty sensor input with a *reconciled* value with the objective of recovering the PEMS performance prior to the fault. Finally, we empirically verify the performances of the proposed SES using a real data set collected at a oil refinery.**

I. Introduction

Continuous acquisition of emission data is a standard legally enforced requirement for the process industry to monitor and control the pollutants released into the atmosphere and to verify that plant emissions do not exceed the thresholds defined by the regulations [1, 2].

The traditional solution employed by the industry to comply with the legislation is to monitor the emissions via hardware-based continuous emission monitoring systems (CEMS); such systems normally comprise analysers (to sample and identify the compositions of released flue gas) and an IT infrastructure to manage, record and store the emissions values [3, 4].

Predictive emission monitoring systems (PEMS) are an innovative technology able to estimate emissions on the base of the values of relevant process parameters to an accuracy comparable with that of the CEMS at a fraction of the cost. PEMS are widely recognized as an effective mean for emission monitoring [3]. Depending on the local legislation, they can be used either as a primary source of monitoring or as a backup of traditional analyzers. PEMS are software-based technologies developed to estimate pollutant concentrations through advanced data-driven (e.g., empirical) models built upon process data (e.g. fuel flow, load, ambient air temperature) [5–7]. The data-driven empirical model for emission monitoring is obtained by training a classical regression model, e.g., a neural network, with training data consisting in measured process data and measured emission obtained via portable CEMS. The interval wherein both real emission and process data are collected is referred to as *data collection period*. At the end of the data collection period, an empirical model can be trained to predict the emission from the process data as input, and the CEMS is removed from the plant. At this point, the emission will be predicted by feeding solely the process data to the trained empirical model,

Clearly, process data ought to be as accurate as possible for a reliable emission estimates [8–10] and regulation enforce procedures for automatically check the PEMS input data. In fact, PEMS shall contain a so-called *Sensor Evaluation System* (SES) for quality assurance of incoming process data. This is typically achieved by means of a pre-processing system that checks whether a sensor is functioning outside its approved operating envelope, at least daily. Upon approval, if it is observed that a sensor might have failed, the SES might also trigger a so-called *Data Reconciliation* block. Reconciled data are *soft-sensor* [11] data that are generated by a SES to replace that of a failed sensor. In this context, soft-sensor data (also known as inferential measurement) are referred to data that comes from an empirical (data-driven) model, instead of a physical hardware sensor. In the literature, this process is often called *Data Validation and Reconciliation* (DVR) [12, 13], and it is briefly described in one of the ensuing sections. The remainder of this document is as follows. In the subsections I-A and I-B the DVR concept and the scope and goals of the paper are highlighted, respectively. Assumptions are stated in subsection I-C. The Robust PEMS framework advocated in this paper is described in Section II. The algorithmic core of the SES unit is described in Section III. Numerical tests are presented in Section IV while concluding remarks are provided in Sec. V.

Notation. Throughout this report, vectors are expressed in bold face letters, while matrices are expressed in capital bold face letter. The symbol $\mathbb{R}$ represents the set of real numbers while $\mathbb{N}^+$ represents the positive natural numbers. The symbol $\preceq$ and $\succeq$ represent the element-wise ordering symbols. The input of a PEMS is the process sensor vector $\mathbf{x} := [x_1, \ldots, x_N]^\top \in \mathbb{R}^N$, where $^\top$ represents the transposition symbol. Clearly, a certain PEMS project will have several PEMS models, one for each variable to predict and monitor (e.g., O2, CO, CO2, NOx, etc...). When not confusion arises,

we will use $\mathbf{x}$ as the general PEMS input, i.e., the process variable sensor data fed into the PEMS model. If $\mathbf{x} \in \mathbb{R}^N$, for each $i \in [1,\dots,N]$, $\mathbf{x}_{(-i)} := [x_1,\dots,x_{i-1},x_{i+1},\dots,x_N] \in \mathbb{R}^{N-1}$ represents the $N-1$ vector of all process measurements except the ith. Here we denote with N the number of sensors inputs to the PEMS model. We denote with $\mathbf{x}_t$ the tth member of the training set, while with $\boldsymbol{x}_t$ the tth member of the test set. The symbols $\|\cdot\|$ and $\|\cdot\|_1$ denote the L_2 and L_1 norm of a vector, respectively, while L_0 denotes the number of non-zero entries of a vector.

A. Data validation and reconciliation

In its classical definition, DVR is a technology that uses process information and mathematical methods in order to automatically correct measurements in industrial processes [12, 13]. Hereinafter, DVR is described in its steady-state formulation. Extensions of DVR to dynamical process have also been considered but PEMS are typically considered static applications [12]. Let us define $\boldsymbol{x} := [x_1,\dots,x_N] \in \mathbb{R}^N$ the N-dimensional vector containing the measurements from N sensors deployed in the plant at a certain (unspecified) time. This vector contains process data such as temperatures, flows, pressures, etc. Conservation laws (mass, heat, etc, ...) can be typically captured via a vectorial function $\mathscr{F} : \mathbb{R}^N \to \mathbb{R}^C$, such that $\mathscr{F}(\boldsymbol{x}) = \mathbf{0}_C$, where C is the number of constraints imposed by the conservation laws. Assuming that for $n = 1,...,N$ there exist bounds $(x_{n,min};x_{n,max})$ such that $x_{n,min} \le x_n \le x_{n,max}$, and denoting σ_n for $n = 1,\dots,N$ the standard deviation of each sensor random measurement, DVR amounts of solving the following problem [12]:

$$\widehat{\boldsymbol{x}} := \arg \min_{\boldsymbol{x}_{min} \preceq \widetilde{\boldsymbol{x}} \preceq \boldsymbol{x}_{max}} \sum_{n=1}^{N} \left(\frac{\widetilde{x}_n - x_n}{\sigma_n} \right)^2 \text{ s.t. } \mathbf{F}(\widetilde{\boldsymbol{x}}) = \mathbf{0} \qquad (1)$$

where $\widehat{\boldsymbol{x}} := [\widehat{x}_1,\dots,\widehat{x}_N] \in \mathbb{R}^N$ is the reconciled variable and $\widetilde{\boldsymbol{x}} := [\widetilde{x}_1,\dots,\widetilde{x}_N] \in \mathbb{R}^N$ is the optimization variable. The problem in (1) is a constrained least-squares. Depending on the constraints imposed by the conservation laws, the problem can be convex or non-convex. When the problem is convex, a global minimum can be found via convex optimization. When the problem is non-convex a local minimum can be found via non-linear programming. The basic assumption of (1) is that the conservation equations are available. For our PEMS problems, this is not the case, and in Sec. III, we will address the problem of finding a form of $\mathscr{F}$ from training process data.

B. Scope and Problem Statement

The scope of this paper is to introduce the concept of Robust PEMS (R-PEMS) with the SES unit consisting of a set of algorithms for Sensor Fault Detection (SFD), Sensor Fault Isolation (SFI), and Reconciliation that can make PEMS robust to sensor errors. Typically errors can be classified in two categories: a) *instantaneous errors*, which are errors that occur sporadically; b) *persistent errors*, which are caused by sensor faults and are most likely to persist in consecutive measurements. Instantaneous errors are not in the scope of this work. In fact, PEMS is not strongly affected by isolated errors since the prediction performance is typically averaged over a long time [14]. Furthermore, instantaneous gross data can be detected resorting to classical outlier detection [15]. Our approach is designed with three main goals in mind:

1) *Detect*: assess whether a new test process vector $\boldsymbol{x}_t$ at time t is a *good* or *bad* point, that is, a bad point is a point that resembles the points in the training set whereas a bad point is one that does not resemble any point in the training set; In case, it is a good point, the point is fed to the current PEMS empirical model;
2) *Isolate*: in case the point is bad, we want to identify which sensor is faulty;
3) *Reconcile*: in case the faulty sensor is isolated, we want to reconstruct the bad data from the faulty sensor with data which are calculated from the rest of the other good (i.e., fault free) sensors data.

C. Assumptions

A fundamental assumption throughout this paper is that of *single sensor fault*. That is, we are assuming that persistent sensor failure are so sporadic that the probability that two sensors fails contemporaneously or in a short time difference is negligible. Therefore, our approach relies upon the fact that we can detect, isolate and reconcile a sensor before another one fails. Clearly, the operator will get the recovery report to signal that a sensor has failed, and he will be asked to replace the failed sensor in the next planned service.

In principle, the assumption of single sensor fault could be removed but the proposed algorithms (in particular the SFI) require significant modifications [1].

In designing the SES, we do *not* assume any knowledge of the process. But we are assuming that we can observe the process via the sensors that are deployed into the system. In particular, we are assuming the existence of a *training set*, $\mathbf{X} := [\mathbf{x}_1,\dots,\mathbf{x}_T] \in \mathbb{R}^{N\times T}$ where $\mathbf{x}_t$ represents the measured point at time t and T is the length of the training set. In a static application such as PEMS, the points do not need to be equispaced. The data $\mathbf{X}$ are observed during the data collection period (that can last weeks or months) and it can be the same adopted for training the PEMS empirical model (for the latter also the measurements of the real emission through the CEMS will be required).

We assume that the training set $\mathbf{X}$ captures a *wide* variety of operating points of the process. Clearly, if an operating point is not observed during the measurement campaign, certain algorithms might detect faulty sensors even if the process is simple working at an operating point different from those examined during the data collection period. We also assume

[1] In the subsection III-C we sketch a possible solution for the multiple sensor faults.

that in the acquisition of the training set $\mathbf{X}$, no persistent sensor failure is present, while instantaneous fault can be present and removed via visual inspection and automatic outlier detection methods. In the ensuing sections the concept of Sensor Fault Detection (SFD), Sensor Fault Isolation (SFI), and Reconciliation (RECON) is introduced.

II. Toward a Robust PEMS: Sensor Fault Detection, Sensor Fault Isolation, Reconciliation

The concept of Robust PEMS (R-PEMS) is depicted in Fig. 1.

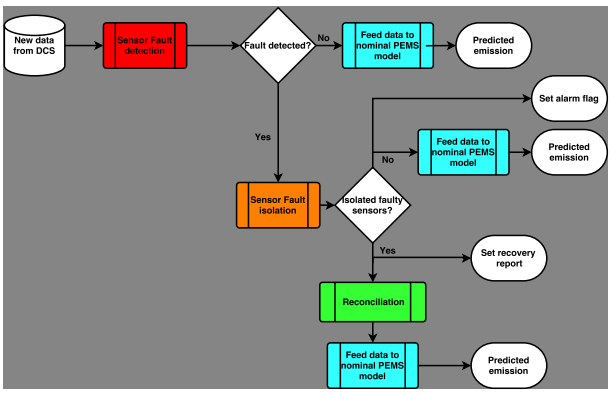

Fig. 1: The concept of R-PEMS: SFD, SFI, Reconciliation.

The existing PEMS implementation is depicted in the cyan block. This block gets as input a set of process variable data and give as output the predicted emission value. We retain this block from the current industry best practise and we do not attempt at optimizing it.

It is envisioned that the R-PEMS application acts as a pre-processing layer according to three different steps:

SFD: The input of the SFD algorithm is the process vector $\boldsymbol{x}_t$ at time t and the output of the SFD is a binary variable SFD_t which states whether the point $\boldsymbol{x}_t$ is faulty ($\text{SFD}_t = 1$) or not faulty ($\text{SFD}_t = 0$). In case the point $\boldsymbol{x}_t$ is not faulty, $\boldsymbol{x}_t$ is fed to the existing PEMS and the emission at time t corresponding to the process vector $\boldsymbol{x}_t$ is evaluated. If $\text{SFD}_t = 1$, we keep observing the evolution of SFD_τ for $t < \tau \leq t + t_{\text{SFD}}$, where $t_{\text{SFD}} \in \mathbb{N}^+$ is an SFD activation length. If $\text{SFD}_\tau = 1$ for $t < \tau \leq t + t_{\text{SFD}}$, we declare that the process vector has one or more persistent faults, and we activate the SFI block. Clearly, large SFD_τ will help in keeping low false alarm rate but will end up in a fault detection activation delay. It is worth point out here that SFD_τ needs to be selected by the PEMS designer to strike a balance between avoiding false alarm and managing small detection delay.

SFI: Once the fault has been declared, the SFI aims at isolating the faulty sensor. Let us recall that we are enforcing the single sensor fault hypothesis. The input of the SFI is a set of consecutive process vectors. Let us denote with t^* the SFI activation time. The input of the SFI could be the set of process vectors $\{\boldsymbol{x}_t\}_{t=t^*}^{t^*+t_{\text{SFI}}-1}$, where t_{SFI} is the SFI length, and the additional delay for SFI is t_{SFI}. Clearly, our rationale is to feed the SFI with points with a sensor fault. Therefore, we want to make sure that only point after t^* are fed to the SFI block. Nevertheless, if isolation delay is of utmost importance, one could think to feed also some point before t^*. For instance, the input of the SFI could also be the set of process vectors $\{\mathbf{x}_t\}_{t=t^*-\lfloor t_{\text{SFI}}/2 \rfloor}^{t^*+\lfloor t_{\text{SFI}}/2 \rfloor}$. In this case, the additional delay for SFI is $\lfloor t_{\text{SFI}}/2 \rfloor$. The output of the SFI is an index $\widehat{i}_{\text{SFI}} \in [1,\ldots,N]$ of the fault sensor.

RECON: Once the faulty sensor index $\widehat{i}_{\text{SFI}}$ has been identified, the last step of the R-PEMS pre-processing is the data reconciliation, i.e., the substitution of the measurement of the faulty sensor with a reconciled value which is calculated from the non-faulty sensors. The cores of the reconciliation step are N *recovery functions* $\ell_n : \mathbb{R}^{N-1} \to \mathbb{R}$ such that the nth sensor can be estimated as a function of the other $N-1$ sensors, i.e.

$$\bar{x}_n := \ell_n(\boldsymbol{x}_{(-n)}), \text{ for } n = 1,\ldots,N, \tag{2}$$

where $\bar{x}_n$ is the nth reconciled value. Let us define the vectorial function $\mathbf{L} : \mathbb{R}^N \to \mathbb{R}^N$, such that

$$\mathbf{L}(\boldsymbol{x}) := [\ell_1(\boldsymbol{x}_{(-1)}),\ldots,\ell_N(\boldsymbol{x}_{(-N)})]^\top. \tag{3}$$

It is worth point out here that the whole reconciled process vector can be obtained as

$$\bar{\boldsymbol{x}} = \mathbf{L}(\boldsymbol{x}) \tag{4}$$

where $\bar{\boldsymbol{x}} := [\bar{x}_1,\ldots,\bar{x}_N]^\top$. If the SFD and SFI blocks have identified that the ith sensor is faulty, then, the reconciliation process substituted the original test set point $\boldsymbol{x} = [x_1,\ldots,x_N]^\top$ with the point $\text{RECON}(\boldsymbol{x}) = [x_1,\ldots,x_{i-1},\ell_i(\boldsymbol{x}_{(-i)}),x_{i+1},\ldots,x_N]^\top$. Clearly, $\text{RECON}(\boldsymbol{x})$ does not depend on the faulty data x_i. The form of the vectorial function $\mathbf{L}(\boldsymbol{x})$ is not specified at the moment. It is worth pointing out here that $\mathbf{L}(\boldsymbol{x})$ is not known a priori and it captures the *dependencies* among differen sensors. We will assume that we can *learn* $\mathbf{L}(\boldsymbol{x})$ from the training set $\mathbf{X}$ and several methods will be advocated in this report to perform such a task, including Locally-Weighted Regressions (LWR) [16, 17] and Feedforward Neural Networks (FFNN) [18], to name a few.

III. Sensor-Centric Balance Equation for SFD, SFI, and RECON

As described in Sec. I-A, the idea of sensor DVR is to detect sensor failures based on multiple (redundant) sensor measurements and the constraint relations among them [19, 20]. Its principle is generally based on consistency checking between the observed behavior of the process provided by the sensors and the expected behavior given by a mathematical representation of the process. Such a mathematical representation, that intrinsically contains the analytical redundancy of the system, can be derived from first principle equations, i.e., mass/energy conservation laws, Ohm/Kirchoff laws in electric networks, etc. However, in many situations, physics-based equations may be difficult to obtain due to the complexity of the process and high process dimensionality [20]. As an alternative to physical-model techniques, methods based on Principal Component

Analysis (PCA) are also used to detect sensor faults [21]. However PCA has its sore point in being a linear technique and most engineering problems are highly nonlinear. To overcome the nonlinear problem, Auto Encoders (AEs) can be used [22, 23]. This approach can be very general provided that the AEs capture the interdependencies among the data. This might require very complex network and a complex mechanism of training, and, therefore, large training data.

In this section, instead, we advocate a method for *learning* an approximate form of balance equations. It is worth pointing out here that we do not assume any process knowledge for performing this task. Any prior knowledge can be embedded in this formalism to improve the learning process. To be specific, given the training set $\mathbf{X}$ we aim at discovering the vectorial function $\mathscr{F}(\mathbf{x}) = \mathbf{0}_C$ of Sec. I-A. To make the above challenging problem tractable, we enforce two assumptions:

Assumption 1: The process we want to analyze is static. [2]

The lack of knowledge on the process, forces us to make general assumptions on the form of $\mathscr{F}$. In particular, we assume the following.

Assumption 2: The vectorial function $\mathscr{F}$ has the following "Sensor-Centric Balance Equation" (SCBE) structure:

$$\mathbf{x} = \mathbf{F}(\mathbf{x}) \rightarrow \mathbf{x} - \mathbf{F}(\mathbf{x}) = \mathbf{0}_N \tag{5}$$

with

$$\mathbf{F}(\mathbf{x}) := [f_1(\mathbf{x}_{(-1)}), \ldots, f_N(\mathbf{x}_{(-N)})]^\top. \tag{6}$$

In other words, we cannot enforce mass/energy balance equations but we assume that each of the N sensors can be reconstructed from the rest of the sensors (therefore the name sensor-centric balance equation). That is, the function $f_i(\mathbf{x}_{(-i)})$ act as a soft-sensor for the ith sensor and is fed from the data of the other $N-1$ sensors.

It can be easily seen that the equation (5) refers to a particular form of $\mathscr{F}$, that is, $\mathscr{F}(\mathbf{x}) = \mathbf{x} - \mathbf{F}(\mathbf{x})$, therefore the number of constraints equals the number of sensors.

The problem of learning the sensor-centric balance equation is converted to learning N scalar functions with $N-1$ arguments. Several approaches for learning $f_i(\mathbf{x}_{(-i)})$ could be adopted. In the ensuing section, we advocate a non-parametric method method, that is, Locally Weighted Regressions (LWR) and, successively, we will clarify the usage of SCBE for SFD, SFI and RECON. We have also attempted to learn $f_i(\mathbf{x}_{(-i)})$ via classical parametric algorithms, i.e., Feedforward Neural Network (FFNN) [25], but for space limits we decided not to include it in this paper.

A. Locally-weighted Regressions

With LWR we do not aim at giving an explicit functional form of $f_i(\mathbf{x}_{(-i)})$ and, therefore, of $\mathbf{F}(\mathbf{x})$. On the other hand, we aim at providing a tool for *evaluating* the functions $f_i(\mathbf{x}_{(-i)})$ and, therefore, of $\mathbf{F}(\mathbf{x})$ for any $\mathbf{x}$ given the training set $\mathbf{X}$.

The crux of LWR is to evaluate $\mathbf{F}(\mathbf{x})$ as a local regression of the points in $\mathbf{X}$ that are *close* (in some sense) to the point $\mathbf{x}$. LWRs have been introduced in [16, 17] and, hereinafter, they will be only quickly reviewed.

Let $\mathbf{x}$ be the point around which we want to evaluate $\mathbf{F}(\mathbf{x})$ and let $\mathbf{x}_t$ the tth point in the training set, that is, $\mathbf{X} := [\mathbf{x}_1, \ldots, \mathbf{x}_T]$. Let us consider the evaluation of the ith function $f_i(\mathbf{x}_{(-i)})$, that is, we want to find the functions that reconstruct the ith sensor given the other $N-1$ sensors. Let $\mathbf{X}_{(-i)} := [\mathbf{x}_{1\,(-i)}, \ldots, \mathbf{x}_{T\,(-i)}] \in \mathbb{R}^{N-1 \times T}$ the original training set without the ith row of the ith sensor data. Let $\mathbf{x}_{(+i)} := [x_{1,i}, \ldots, x_{T,i}]^\top \in \mathbb{R}^T$ be the column vector with the entries of the ith row of $\mathbf{X}$.

The core of the LWR is a distance function d which defines how distant are two vectors. Without loss of generality, for the sake of clarity, we can consider d to be the Euclidean distance, i.e., $d(\mathbf{x}, \mathbf{y}) := \|\mathbf{x} - \mathbf{y}\|$. In LWR, the function $f_i(\mathbf{x}_{(-i)})$ is evaluated as a local linear regression, that is

$$f_i(\mathbf{x}_{(-i)}) = \theta_0 + \boldsymbol{\theta}^\top \mathbf{x}_{(-i)} \tag{7}$$

where θ_0 and $\boldsymbol{\theta} \in \mathbb{R}^{N-1}$ are, respectively, the intercept and the regression coefficients, and are obtained for every point $\mathbf{x}$ via weighted least-squares, that is, by solving the following problem:

$$(\theta_0, \boldsymbol{\theta}) = \arg\min_{\widetilde{\theta}_0, \widetilde{\boldsymbol{\theta}}} \sum_{t=1}^{T} d(\mathbf{x}, \mathbf{x}_{t\,(-i)})(x_{t,i} - \widetilde{\theta}_0 - \widetilde{\boldsymbol{\theta}}^\top \mathbf{x}_{t\,(-i)})^2 \tag{8}$$

The above problem amounts to a linear regression and it can be solved in closed form with simple algebraic operations. However, the above problem needs to be solved for each $f_i(\mathbf{x}_{(-i)})$, therefore, N times. Furthermore, as the regression coefficients depends on the evaluation point $\mathbf{x}$, the problems in (8) need to be solved for every new point $\mathbf{x}$.

B. SCBE-based SFD

Once the SCBE has been learnt (via LWR, FFNN, or any other method), SFD can be performed. In fact, abnormal situations that occur due to sensor faults induce changes in sensor measurements.

Let us consider a test process sensor data vector $\boldsymbol{x}$. In the fault-free case, we will have:

$$\boldsymbol{x} = \mathbf{F}(\boldsymbol{x}) + \boldsymbol{x}^e \tag{9}$$

where $\boldsymbol{x}^e$ denotes the error vector due to both measurement noise and model mismatch. From the training data set $\mathbf{X}$ of fault-free process sensor data, we can estimate the expected size of the error $\boldsymbol{x}^e$, and we expect its norm to be *small*, that is

$$\|\mathbf{x} - \mathbf{F}(\mathbf{x})\| < \gamma, \tag{10}$$

[2]The problem of discovering nonlinear dynamics of the process [24] is out of the scope of this work. We plan to tackle this problem in future works.

for every point $\mathbf{x}$ in the training set. γ is the size of the error that one should expect when the process sensor training data $\mathbf{x}$ are approximated with the process soft-sensor data $\mathbf{F}(\mathbf{x})$ in the fault-free case. The crux of the SCBE-based SFD is to monitor the size of the error between the process sensor test data $\boldsymbol{x}$ and the process soft-sensor data $\mathbf{F}(\boldsymbol{x})$, i.e., $\|\boldsymbol{x}-\mathbf{F}(\boldsymbol{x})\|$. In particular, a fault is declared in case this error exceeds γ, i.e.,

$$DS_{SCBE}(\boldsymbol{x}) := \|\boldsymbol{x}-\mathbf{F}(\boldsymbol{x})\| > \gamma \tag{11}$$

where $DS_{SCBE}(\boldsymbol{x})$ is the decision statistic for the sensor-centric balance equation SFD.

The γ threshold in (11) is set to guarantee a certain fixed false alarm from the training and validation set. If the false alarm is to be minimized, an effective strategy is set γ as the maximum of the decision statistics over the training set plus a correction factor.

Remark 1: It is important to note that that SCBE might suffer from the fact that for some sensors it is not possible to find a relationship of the form $x_i = f_i(\mathbf{x}_{(-i)})$ or, better, this relationship is uninformative and entails a large residual error, that is the SFD unit will generate false positives. In the test cases encountered in this study the hypothesis of the existence of a high quality sensor balance equation holded. If this is not the case, the form of $\mathbf{F}$ can be modified using some domain knowledge about the process, i.e., certain rows corresponding to the sensors for which the sensor-centric balance equation has to be removed.

C. SCBE-based SFI

The goal of SFI is to identify which sensor out of N has failed. Without loss of generality, if a fault is declared and the SFI is activated, then it turns out that the process sensor data vector is affected by an unknown signature vector.

We envision that the test process sensor data vector $\boldsymbol{x}$ is of the form

$$\boldsymbol{x} = \boldsymbol{x}^* + \boldsymbol{x}^e + \boldsymbol{e} \tag{12}$$

where $\boldsymbol{x}^*$ is the fault-free process sensor vector (that we would get in case of no fault) and, in case of fault of the ith sensor, $\boldsymbol{e}$ is of the form $\boldsymbol{e} = e\mathbf{1}_i$ with $i \in [1,\dots,N]$ and $\mathbf{1}_i$ being the N-dimensional Cartesian basis vector with all zero entries except in the position i where the entries is equal to 1 (this reflects our assumption of single sensor failure only) and $e \in \mathbb{R}$ indicating the magnitude of the fault signature.

We now attempt at estimating the size of the error $\boldsymbol{e}$ to identify the faulty sensor as originally done in [26]. If the error was in the ith sensor, the most likely error vector $\boldsymbol{e} = e\mathbf{1}_i$ would be the one minimizing the mismatch between the process sensor data and the process soft-sensor reconstructed data, i.e.,

$$E_i = \min_{e\in\mathbb{R}} \|\boldsymbol{x}+e\mathbf{1}_i - \mathbf{F}(\boldsymbol{x}+e\mathbf{1}_i)\|. \tag{13}$$

Then, the most likely faulty sensor is the one providing the smallest index E_i for $i = 1,\dots,N$. Mathematically, we have that $\widehat{i}_{\text{SFI}}$ is given by:

$$\widehat{i}_{\text{SFI}} = \arg\min_{i\in[1,\dots,N]} E_i. \tag{14}$$

Finally, the SFI can be achieved solving the following problem:

$$\widehat{i}_{\text{SFI}} = \arg\min_{i\in[1,\dots,N]} \left[\min_{e\in\mathbb{R}} \|\boldsymbol{x}+e\mathbf{1}_i - \mathbf{F}(\boldsymbol{x}+e\mathbf{1}_i)\|\right] \tag{15}$$

With the minimization in (13), we are basically seeking the error vector $\boldsymbol{e}$ to compensate for the sensor-centric *imbalance* situation created by the sensor fault. We can find it by minimizing the imbalance residual with the goal to obtain the new balance equation:

$$\boldsymbol{x}+\boldsymbol{e} = \mathbf{F}(\boldsymbol{x}+\boldsymbol{e}) \tag{16}$$

It turns out that, if the index $\widehat{i}_{\text{SFI}}$ is indeed the index to correct, then

$$E_{\widehat{i}_{\text{SFI}}} < \gamma \tag{17}$$

This constitutes a sanity check on whether the SFI was correct or not.

Remark 2: Inspired by the works in [26, 27], the formulation as in (15) can be extended to the case of multiple faults as follows:

$$\hat{\boldsymbol{e}}_s = \arg\min_{\boldsymbol{e}_s} \|\boldsymbol{x}+\boldsymbol{e}_s - \mathbf{F}(\boldsymbol{x}+\boldsymbol{e}_s)\| + \lambda\|\boldsymbol{e}_s\|_1 \tag{18}$$

where the regularization parameter λ governs the trade-off between satisfying the SCBE and minimizing the number of faulty sensors. Here $\boldsymbol{e}_s \in \mathbb{R}^N$ is the vector of sensors fault signatures and it is assumed to be sparse (the number of faulty sensors is small compared to the total number of sensors). This assumption is integrated in the optimization problem by introducing an ℓ_1 regularization term into the objective function (18). It is straightforward to note that the problem in (18) resembles that in (15) when $\|\boldsymbol{e}_s\|_0 = 1$, that is $\boldsymbol{e}_s = \boldsymbol{e}$. We plan to include the analysis of the multiple faulty sensors in a future work.

D. SCBE-based RECON

Once we have detected and isolated the faulty sensor, the reconciliation of the fault data is performed, again, adopting the SCBE. We have highlighted in (5) the recovery process. However, in Sec. II we did not specified a form of the recovery function $\mathbf{L}(\boldsymbol{x})$. At this point, it is natural that our preferred way to select the recovery function $\mathbf{L}(\boldsymbol{x})$ is via the learnt SCBE, therefore, hereinafter we use the following recovery function for data reconciliation:

$$\mathbf{L}(\boldsymbol{x}) = \mathbf{F}(\boldsymbol{x}) \tag{19}$$

where $\mathbf{F}(\boldsymbol{x})$ is defined in Sec. III and can be estimated, among other methods, via those highlighted in Section III-A.

E. Implementation Issues

In this subsection, we aim at highlighting some key implementation issues that were overlooked during the description of the methods to keep it lean, but are key to the successfulness of the methods.

Normalization. The data are centered around zero and normalized to have unit variance before any processing;

Filtering. In a real application, measurement points will arrive consecutively over time. Since we are focusing on persistent errors, filtering (that is, integrating) the decision statistic in (11) over a certain interval, is beneficial to come up with a more reliable detection. The length of the filter will end up in a decision delay however that has to be contrasted with the performance gain in detection reliability.

IV. Tests

In this section, we empirically verify the performances of the proposed SES unit that has been introduced in the previous sections. We are considering a data set collected during a real PEMS project. The original dataset consists of data produced by more than 100 sensors. Out of these sensors (also known as tags in the industrial plant parlance), only 29 are used for the PEMS of several gases, that is $\mathbf{x} \in \mathbb{R}^{29}$. In this paper, we are focusing only one particular gas for the sake of brevity, O_2, and we will be dealing only with the 8 sensors as input to the prediction model of O2, that is, $\mathbf{x} \in \mathbb{R}^8$. The dataset collected during the collection period consists of 4160 time ordered observations from 8 different sensors. The dataset is split into a training set, validation set and testing set. Random splitting which is typically used in machine learning, cannot be used with time series data that by definition occur in a temporal sequence. This is the reason why we adopted a splitting strategy with the objective of preserving the temporal structure of the series as untouched as possible [28]. We use a systematic sampling method (which we call $4-1-1$ splitting method) that splits the dataset in discrete blocks. Specifically we build the training set by combining non-overlapping blocks of 4 consecutive data points that are down-sampled from the original dataset every 6 data points. The validation set and the testing set consist of the subsequences obtained by down-sampling the original data set every 5 data points starting from the 5th entry and the 6th entry of the data set, respectively. The training, validation and testing datasets will have the size of 2774, 693 and 693, respectively.

A. Simulation parameters

In this study we focus our attention on the *persistent faults* characterized by continuity of the fault signature occurrence leading to an observable pattern that we can learn over time [29]. These criteria draw from experience and are applicable to a wide types of faulty sensor readings including sensor drifts and biases. Within the class of persistent faults, we have experimentally observed that, among all possible faults, the *sensor freeze fault* (at time t_{freeze} the sensor output gets stuck at a fixed value) was the hardest to detect with general purpose fault detector. Clearly, such a fault can be detected with a dedicated test on the derivative or variance. Such a rule would nevertheless not detect other type of persistent fault such as offset and bias. In this section, we want to analyze how our SFD/SFI methods would perform on sensor freeze. Results on the performance of the algorithms for sensor drifts and offsets are omitted due to space limits and they will be provided in future works. We have simulated a sensor freeze fault in each of the 8 sensors. From the original test set, we have created 8 test sets wherein the ith sensor is frozen at its value at time $t = 300$ (we have performed test for various freezing times but for brevity they are not be reported in this paper). Figures 2-9 show the frozen and the original data values for sensors 1-8, respectively.

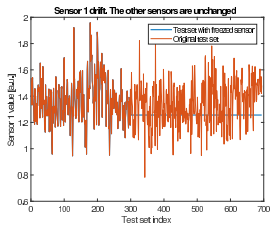

Fig. 2: Sensor 1: original vs frozen.

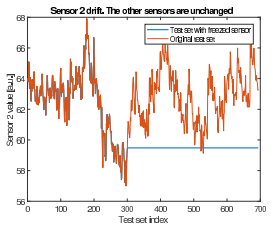

Fig. 3: Sensor 2: original vs frozen.

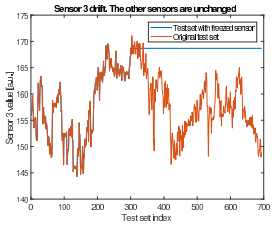

Fig. 4: Sensor 3: original vs frozen.

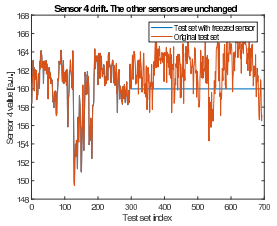

Fig. 5: Sensor 4: original vs frozen.

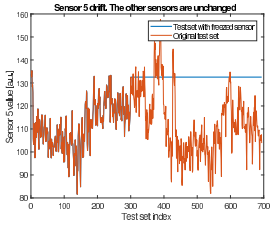

Fig. 6: Sensor 5: original vs frozen.

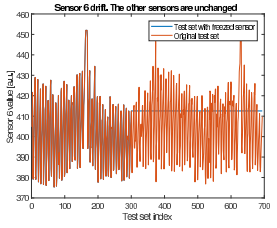

Fig. 7: Sensor 6: original vs frozen.

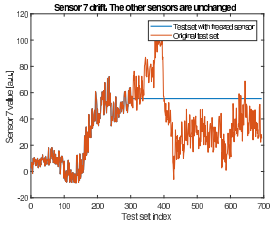

Fig. 8: Sensor 7: original vs frozen.

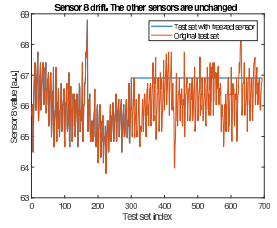

Fig. 9: Sensor 8: original vs frozen.

B. Results

The performances of the SCBE-based SFD with a filter length of 50 of the decision statistics along with the decision threshold for the SFD are shown in Figs. 10-17. The method is effective in identifying faults for all sensors. In fact, the decision statistics start increasing at the point of the fault (i.e., $t = 300$) and right after the decision threshold is exceeded. It is still to be noticed that the margin in the SFD of sensor 1 is small.

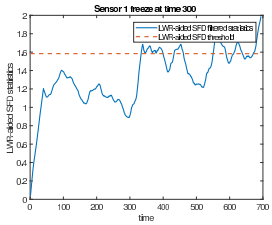

Fig. 10: SCBL-LWR SFD for Sensor 1 freeze.

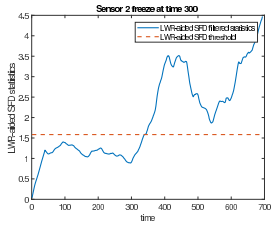

Fig. 11: SCBL-LWR SFD for Sensor 2 freeze.

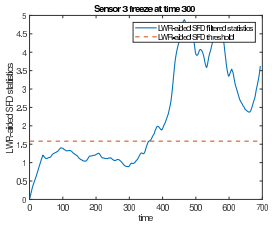

Fig. 12: SCBL-LWR SFD for Sensor 3 freeze.

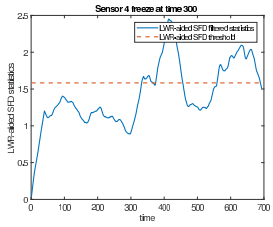

Fig. 13: SCBL-LWR SFD for Sensor 4 freeze.

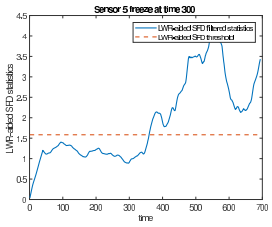

Fig. 14: SCBL-LWR SFD for Sensor 5 freeze.

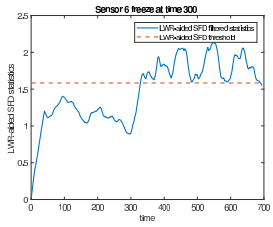

Fig. 15: SCBL-LWR SFD for Sensor 6 freeze.

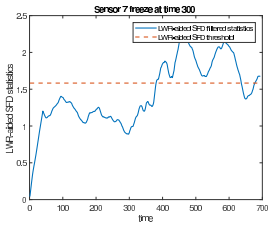

Fig. 16: SCBL-LWR SFD for Sensor 7 freeze.

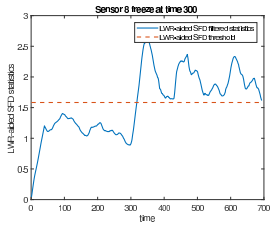

Fig. 17: SCBL-LWR SFD for Sensor 8 freeze.

Due to the positive results of the SFD, we have performed the SFI algorithm. The SFI rule in (15) is tested. In particular, the evolution of the statistics E_i vs the integration time is plotted in Figs. 18-25. We have considered a maximum integration time of 50 samples starting from the SFD trigger time. In Figs. 18-25 we have highlighted with solid black thick lines the sensor corresponding to the minimum error as advocated in (15). From 18-25 it is apparent that the rule in (15) effectively isolates the faulty sensor, as the cumulative sum of E_i of the faulty sensor is always the lowest.

So far we have assessed the effectiveness of the SFD and SFI algorithms. We aim now at assessing the performance of the whole R-PEMS, that is, we want to see which is the performance degradation of the Classical PEMS and the Robust PEMS in the presence of a sensor freeze. We have observed that the PEMS is very sensible to fault in sensor 1, in the sense that small error in sensor 1 reflects in large error in the emission prediction. Such error can easily exceed the 10% acceptance limit imposed by regulations [8]. The original Classical PEMS performance with no fault, along with the performance of the Classical PEMS and R-PEMS with sensor 1 freeze at time 300, are shown in Fig. 26, wherein the relative accuracy over the test set is calculated. The x axis represents the block index. Indeed, we have split the 693 test set points into 9 block of $T = 70$ points and the last of $T = 63$ points and we have calculated the relative accuracy (RA), that is $\text{RA} := \frac{\frac{1}{T}\sum_{t=1}^{T}|p_t - v_t|}{\frac{1}{T}\sum_{t=1}^{T}|v_t|}$, where p_t are the predicted (via the PEMS) and the ground truth measures (via the CEMS) emission values at time t. From Fig. 26 it is clear that, until the fault occurs, the Classical PEMS and the R-PEMS perform identically. After the fault, which occurs in blocks 5 and successive, the Classical PEMS and R-PEMS have a performance impairment with respect to the fault free case and the performance of both Classical and R-PEMS exceed the 10% limit. After some time, that is, at block 7 and successive, the fault has been detected, isolated and reconciled as per subsections III-B-III-D. At this point, the Classical PEMS is fed with the frozen sensor value while the R-PEMS is fed with the reconciled values of sensor 1. After the fault has been detected, the R-PEMS has a slight degradation with its RA exceeding the 10% limit. However, after this transient period, the R-PEMS is able to recover and bring back its RA below the critical 10% limit while the Classical PEMS will still perform above the 10% limit.

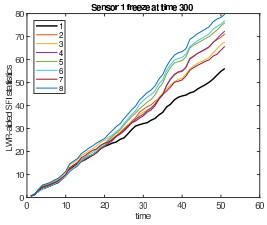

Fig. 18: SCBL-LWR SFI for Sensor 1 freeze.

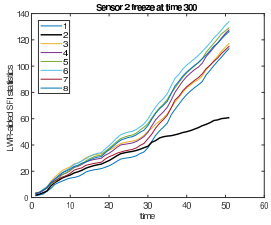

Fig. 19: SCBL-LWR SFI for Sensor 2 freeze.

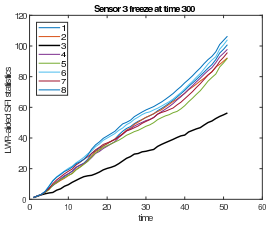

Fig. 20: SCBL-LWR SFI for Sensor 3 freeze.

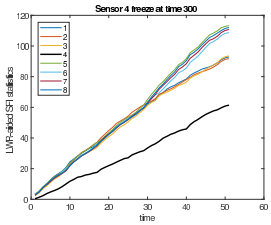

Fig. 21: SCBL-LWR SFI for Sensor 4 freeze.

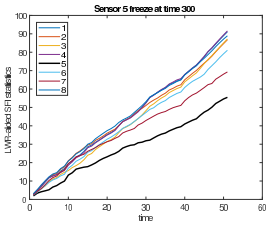

Fig. 22: SCBL-LWR SFI for Sensor 5 freeze.

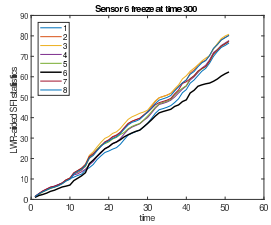

Fig. 23: SCBL-LWR SFI for Sensor 6 freeze.

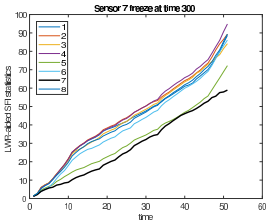

Fig. 24: SCBL-LWR SFI for Sensor 7 freeze.

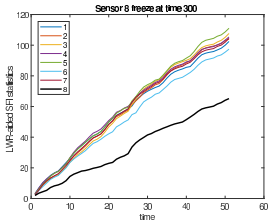

Fig. 25: SCBL-LWR SFI for Sensor 8 freeze.

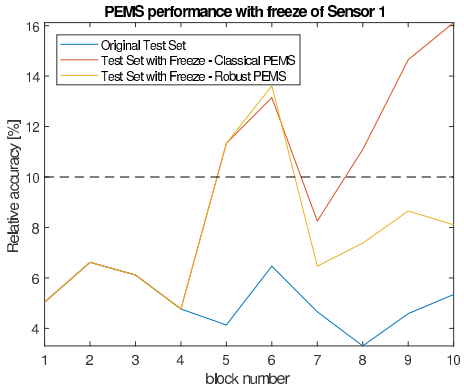

Fig. 26: PEMS performance in presence of freeze in sensor 1: Classical PEMS vs Robust PEMS.

V. Conclusions

In this paper, we introduce the concept of Robust PEMS for making conventional PEMS resilient to sensor faults. We have proposed a novel formalism based on the estimation of the sensor-centric balance equation, wherein a special form of balance equations are learnt using classical LWR methods. This approach is the core technology enabling effective SFD, SFI and RECON. Finally, preliminary tests, performed using real data sets, have shown how the Robust PEMS is able to comply with the regulations by recovering its performance below the tight limits on pollutant emissions. In future, we plan to address the case of multiple sensor faults and corroborate our ongoing studies with additional numerical tests on new data sets.

References

[1] J. Jahnke, *Continuous Emission Monitoring*. Wiley, 2000.

[2] E. Arioni, N. Bonavita, and M. Paco, "Keeping an eye on emissions," *Hydrocarbon Engineering Magazine*, vol. 18, no. 10, pp. 43–49, October 2013.

[3] R. Paira, G. Ganapathiraman, and A. Avery, "Emission monitoring systems global market research study," August 2017. [Online]. Available: https://www.arcweb.com/market-studies/emission-monitoring-systems.

[4] [Online]. Available: http://new.abb.com/products/measurement-products/analytical.

[5] J. Keeler and R. Ferguson, "Commercial applications of soft sensors: The virtual on-line analyzer and the software cem," in *Proceedings of the International Forum for Process Analytical Chemistry*, 1996.

[6] N. Bonavita and F. Callero, "Model-based emission monitoring," *ABB Review*, no. 3/11, pp. 58–63, 2011.

[7] N. Bonavita and G. Ciarlo, "Inferential sensors for emission monitoring: an industrial perspective," *Frontiers in Environmental Engineering (FIEE)*, vol. 3, 2014.

[8] EPA, "Performance specification 16 for predictive emissions monitoring systems," 2005. [Online]. Available: https://www.epa.gov.

[9] CEN, "Stationary source emissions - predictive emission monitoring systems (pems) - applicability execution and quality assurance," 2017.

[10] B. B. G. Ciarlo, E. Bonica and N. Bonavita, "Assessment and testing of sensor validation algorithms for environmental monitoring applications," *Chemical Engineering Transactions*, vol. 57, 2017.

[11] L. Fortuna, S. Graziani, A. Rizzo, and M. Xibilia, *Soft Sensors for Monitoring and Control of Industrial Processes*, ser. Advances in Industrial Control. Springer London, 2007.

[12] S. Narasimhan and C. Jordache, *Data Reconciliation and Gross Error Detection*. Gulf Professional Publishing, 1999.

[13] M. M. Câmara, R. M. Soares, T. Feital, T. K. Anzai, F. C. Diehl, P. H. Thompson, and J. C. Pinto, "Numerical aspects of data reconciliation in industrial applications," *Processes*, vol. 5, no. 4, 2017.

[14] S. J. Qin, H. Yue, and R. Dunia, "Self-validating inferential sensors with application to air emission monitoring," *Industrial & Engineering Chemistry Research*, vol. 36, no. 5, pp. 1675–1685, 1997.

[15] P. J. Rousseeuw and A. M. Leroy, *Robust Regression and Outlier Detection*. New York, NY, USA: John Wiley & Sons, Inc., 1987.

[16] W. S. Cleveland, "Robust locally weighted regression and smoothing scatterplots," *Journal of the American Statistical Association*, vol. 74, no. 368, pp. 829–836, 1979.

[17] W. S. Cleveland and S. J. Devlin, "Locally weighted regression: An approach to regression analysis by local fitting," *Journal of the American Statistical Association*, vol. 83, no. 403, pp. 596–610, 1988.

[18] K.-I. Funahashi, "On the approximate realization of continuous mappings by neural networks," *Neural Networks*, vol. 2, no. 3, pp. 183 – 192, 1989.

[19] E. Chow and A. Willsky, "Analytical redundancy and the design of robust failure detection systems," *IEEE Transactions on Automatic Control*, vol. 29, no. 7, pp. 603–614, Jul 1984.

[20] S.-C. Lee and C.-E. Park, "Sensor value validation based on implicit sensor redundancy for reliable operation of power plants," *IEEE Transactions on Energy Conversion*, vol. 20, no. 2, pp. 373–380, June 2005.

[21] R. Dunia, S. J. Qin, T. F. Edgar, and T. J. McAvoy, "Identification of faulty sensors using principal component analysis," *AIChE Journal*, vol. 42, no. 10, pp. 2797–2812, 1996. [Online]. Available: http://dx.doi.org/10.1002/aic.690421011

[22] J. Keeler, J. Havener, D. Godbole, and R. Ferguson, "Virtual continuous emission monitoring system with sensor validation," Apr. 5 2000, eP Patent 0,712,509.

[23] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *Proceedings of the 23rd ACM SIGKDD*, ser. KDD '17, 2017.

[24] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences*, vol. 113, no. 15, pp. 3932–3937, 2016.

[25] C. M. Bishop, *Neural networks for pattern recognition*. Oxford university press, 1995.

[26] D. Gorinevsky, S. Boyd, and S. Poll, "Estimation of faults in dc electrical power system," in *2009 American Control Conference*, June 2009, pp. 4334–4339.

[27] B. Li, H. Yu, J. Dauwels, and K. S. Low, "Sensor fault detection by sparsity optimization," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014, pp. 7614–7618.

[28] S. Lohr, *Sampling: Design and Analysis*, ser. Sampling: Design and Analysis. Duxbury Press, 1999, vol. 1.

[29] K. Ni, N. Ramanathan, M. N. H. Chehade, L. Balzano, S. Nair, S. Zahedi, E. Kohler, G. Pottie, M. Hansen, and M. Srivastava, "Sensor network data fault types," *ACM Transactions on Sensor Networks (TOSN)*, vol. 5, no. 3, p. 25, 2009.