# A Self-building and Cluster-based Cognitive Fault Diagnosis System for Sensor Networks

Cesare Alippi, *Fellow, IEEE,* Manuel Roveri, *Member, IEEE,* and Francesco Trovò, *Member, IEEE*

***Abstract*—Cognitive fault diagnosis systems differentiate from more traditional solutions by providing online strategies to create and update the fault-free and the faulty classes directly from incoming data. This aspect is of paramount relevance within the big data framework, since measurements are there immediately processed to detect and identify the upsurge of potential faults. The paper introduces a novel cognitive fault diagnosis framework for processes described by nonlinear dynamic systems that inspects changes in the existing relationships among sensors. The proposed framework is based on an evolving clustering algorithm that operates in the parameter space of time invariant linear models approximating such relationships. During the operational life, parameter vectors associated with models thought not to belong to the nominal state are either labeled as *outlier* or *fault*. New classes of faults, here considered to propagate to the model parameters according to an abrupt profile, are created on-line as they appear. At the same time, existing classes can merge, depending on the information content carried by incoming data.**

***Index Terms*—fault diagnosis, adaptive learning, evolving clustering**

## I. Introduction

Fault Diagnosis Systems (FDSs) are tools designed to detect, isolate, identify and, possibly, mitigate the occurrence of faults affecting complex systems. FDSs have been subject of extensive research for their relevance in real-world applications, e.g., see [1]–[4] for a comprehensive review. In their traditional framework it is required the availability of the fault-free nominal state and a "fault dictionary", containing the fault signatures. Both requests constitute a strong demand, hard to be met in most of real-world applications.

A novel and promising *cognitive* approach aims at designing FDSs able to automatically learn the nominal and the faulty states online, during the operational modality. Cognitive approaches generally rely on machine learning techniques to configure the nominal state and create the faulty ones without requiring any a-priori information about the fault signature or on fault time profile (e.g., [1], [5]).

Most of existing cognitive FDSs apply the learning mechanism only during the configuration phase [6]–[9], thus requesting availability of the fault dictionary at training time. More in detail, [6] presents a learning methodology for incipient failure detection based on online approximators aiming at both inspecting variations in the system due to faults and providing information about the detected faults in an online manner. In [7], a learning procedure for fault accommodation is given, under the assumption that the process is linear. [10] presents a scheme for online adaptive fault detection and accommodation. There, it is requested that the nominal fault-free state of the system is known. Differently from previous solutions, that confine the cognitive aspect solely to the training phase (hence not allowing the FDS to improve the fault dictionary during the operational life), [1] suggests the use of an unsupervised "clustering-labeling" method to automatically assign observations either to the nominal or the faulty class. Unfortunately, no technical details about the implementation of the solution are given.

There is a large literature addressing the design of cognitive FDSs for specific applications [8], [9], [11]–[20], with cognitive mechanisms mostly applied during the training phase of the FDS. For example, [9] presents a supervised method for fault classification which exploits a recursive learning of a radial basis function network in chemical processes. [8] suggests a cluster-labeling approach based on Self Organizing Maps for fault diagnosis applied to a quality inspection of tape deck chassis. [11] describes a FDS specifically designed for fault isolation in power transformers based on evolving neural networks. In [12], the authors propose an intelligent FDS for electric motors based on ART-Kohonen Neural Networks: new faults can be included in the dictionary thanks to the design of a case-based reasoning learning system. Several FDSs based on fuzzy neural networks have been presented in the literature [13]–[20], mainly addressing specific applications (e.g., bearing [14], induction motors [15], transformers [16], [17], marine propulsion engines [18], gearboxes [19], circuit transmission [20]), while [13] suggests a fault identification technique based on the joint use of a fuzzy logic and feedforward neural networks. All presented methods are either application specific or request availability of the fault dictionary, strong assumptions that we relax in the sequel.

More in detail, the paper presents a cognitive FDS working in the parameter space of Linear Time-Invariant (LTI) models approximating the investigated process dynamics over time. The proposed FDS, which extends the solution presented in [21], relies on a novel evolving-clustering algorithm able to learn the nominal state of the process during an initial training phase and create, update and maintain the fault dictionary automatically during the operational life. During the training phase, the cognitive FDS characterizes the nominal fault-free state and, in the following operational phase, assesses approximating models by labeling them as fault-free, instances of a new faulty class or outliers. A sound theoretical framework justifies the use of approximating linear models to detect changes.

C. Alippi, M. Roveri and F. Trovò are with the Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Piazza L. da Vinci 32, 20133 Milano, Italy e-mail: {alippi,roveri,trovo}@elet.polimi.it.

The main contributions of the paper can be summarized as proposing:

a) the design of an evolving FDS based on an adaptive clustering algorithm working in the space of approximating model parameters, able to characterize faults whose effect induce an abrupt change in the model parameters;
b) the theoretical justification for the use of a sequence of LTI models approximating the (possibly) nonlinear dynamic system within a fault diagnosis framework;
c) an evolving-clustering algorithm that takes advantage of temporal and spatial dependencies of the estimated parameters, whereas clustering solutions present in the literature usually consider only the spatial aspect e.g., [22]–[25].

The structure of the paper is as follows. Section II reviews the theoretical framework justifying the use of LTI models as building blocks to construct the cognitive FDS. Section III introduces the proposed cognitive FDS and Section IV details the aspects related to the on-line creation of the fault dictionary. Experimental results on both synthetic and real datasets are presented and discussed in Section V. Concluding remarks are finally given in Section VI.

## II. Problem Formulation

In the following, we consider a time invariant dynamic system whose model description is unavailable and a sensor network acquiring scalar measurements - or datastreams - from the system. Selection of the most appropriate placement for the sensors is outside the scope of this paper (the interested reader can refer to [26]–[29] for a comprehensive investigation of the displacement problem). We assume that changes in the system can be detected by inspecting changes in the functional relationships among sensor data. Each relationship between two generic sensors is described as presented in the sequel and the final decision about the change detection is taken at the network level, by relying on the framework proposed in [30].

In the following, each sensor-to-sensor relationship is modeled as a single time invariant process $\mathcal{P}$ (extension to relationships described by a finite set of non-overlapping processes $\{\mathcal{P}_1, \ldots, \mathcal{P}_\psi\}$ is immediate) and is approximated with a LTI predictive model belonging to a family $\mathcal{M}$ parametrised in $\theta \in \mathcal{D}_\mathcal{M}$, $\mathcal{D}_\mathcal{M} \subset \mathbb{R}^p$ being a compact $C^1$ manifold. MISO linear predictive models [31], Extreme Learning Machines [32], Reservoir Computing Networks [33] are valuable instances for $\mathcal{M}$. In this paper, we opt for linear one-step-ahead predictive models in the form:

$$\hat{y}(t|\theta) = f\left(t, \theta, u(t), \ldots, u(t-\tau_u), y(t-1), \ldots, y(t-\tau_y)\right) \quad \forall t \in \mathbb{N}$$

where $f(\cdot) \in \mathbb{R}$ is the approximating function in predictive form [34], e.g., ARX, ARMAX, $u(t) \in \mathbb{R}^m$ and $y(t) \in \mathbb{R}$ are the model input and output at time $t$, respectively, and $\tau_u$ and $\tau_y$ are the orders of the input and output, respectively. Given a training sequence composed of $N$ couples $Z_N = \{(u(t), y(t))\}_{t=1}^N$ and a quadratic loss function, we define the structural risk [34] to be:

$$W_N(\theta) = \frac{1}{N} \sum_{t=1}^{N} \mathbb{E}_{(u,y)} \left[\varepsilon^2(t, \theta)\right]$$

and the empirical risk as:

$$V_N(\theta) = \frac{1}{N} \sum_{t=1}^{N} \varepsilon^2(t, \theta),$$

where $\varepsilon(t, \theta) = y(t) - \hat{y}(t|\theta)$ is the prediction error at time $t$. The optimal parameter $\theta^o \in \mathcal{D}_\mathcal{M}$ is defined as

$$\theta^o = \arg \min_{\theta \in \mathcal{D}_\mathcal{M}} \left[\lim_{N \to +\infty} W_N(\theta)\right].$$

An estimate $\hat{\theta} \in \mathcal{D}_\mathcal{M}$ of $\theta^o$ can be obtained by minimizing the empirical risk:

$$\hat{\theta} = \arg \min_{\theta \in \mathcal{D}_\mathcal{M}} V_N(\theta). \tag{1}$$

By relying on the theoretical framework developed by Ljung [34], [35], under the mild hypotheses that recent past data suffice to generate accurate approximations of $u(t)$ and $y(t)$, that $f(\cdot)$ is three time differentiable with respect to $\theta$, and satisfies Lipschitz conditions, and that the structural risk is a convex function in $\mathcal{D}_\mathcal{M}$, minimization of $W_N(\theta)$ provides a unique point $\theta^o$ such that:

$$\lim_{N \to \infty} \hat{\theta} = \theta^o \quad w.p.\ 1$$

and

$$\lim_{N \to \infty} \sqrt{N} \Sigma_N^{-\frac{1}{2}} (\hat{\theta} - \theta^o) \sim \mathcal{N}(0, I_p) \tag{2}$$

where

$$\Sigma_N = \left[W_N''(\theta^o)\right]^{-1} U_N \left[W_N''(\theta^o)\right]^{-1},$$
$$U_N = N\mathbb{E}\left[V_N'(\theta^o) V_N'(\theta^o)^T\right]$$

and $I_p$ is the identity matrix of order $p$.

The above result assures that, given a sufficiently large $N$, the estimated parameter vector $\hat{\theta}$ follows a multivariate Gaussian distribution with mean $\theta^o$ and covariance matrix $\Sigma_N$. Interestingly, the results presented in Sec. II contemplate the situation where $\mathcal{P} \notin \mathcal{M}$ i.e., a model bias $||\mathcal{M}(\theta^o) - \mathcal{P}|| \neq 0$ is present. This justifies the use of LTI models even when the dynamic system under investigation is non-linear. According to (2), estimated parameters $\hat{\theta}$ follow a multivariate Gaussian distribution both approximating linear and nonlinear systems, provided that a sufficiently large dataset is available. We emphasize that, in what follows, we are not interested in providing a high approximation accuracy, since LTI models are not used for prediction purposes (where nonlinearities in the system might induce a high prediction error) but for fault diagnosis ones. Parameter vectors are the features to be used for fault diagnosis and, since a change in the probability density function of the parameter-features is associated with structural changes in the process generating the data (and nonlinearity does not introduce structural changes), we can design a FDS based on an evolving-clustering algorithm operating in the parameter space.

Although the non-linearity aspect is contemplated by the theory, we might experience numerical problems in correspondence with an ill conditioned Hessian $W''_N$, e.g., following highly correlated inputs. However, we must comment that if $W''_N$ degenerates in rank then, given the linearity assumption for the considered approximation model, we should simply remove the linear dependent variables. In the case we wish to keep them for the (small) innovation they provide, a Levenberg-Marquardt correction $W''_N + \delta I_p$ ($\delta$ being a small positive scalar) should be introduced to grant a definite positive Hessian.

## III. Cognitive FDS

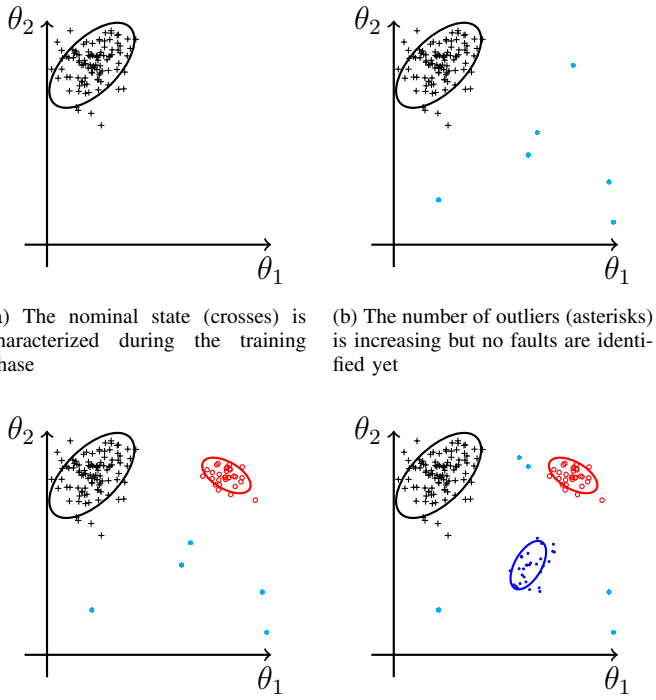
The FDS relies on an initial training phase needed to characterize the nominal state $\Psi$ by exploiting a fault-free training sequence $Z_M = \{(u(t), y(t))\}_{t=1}^{M}$. $Z_M$ is then windowed into non-overlapping batches of length $N$, each of which used to provide a parameter vector estimate $\hat{\theta}$. The outcome is the sequence $\Theta_L = (\hat{\theta}_1 \dots \hat{\theta}_i \dots \hat{\theta}_L), L = M/N$.

The proposed cognitive FDS is given in Alg. 1. From results delineated in Sec. II parameter vectors in $\Theta_L$ are distributed according to the Gaussian distribution, provided that $N$ is large enough, even though the system is non-linear. Thanks to Eq. 2 the nominal state $\Psi$ can be described as a Gaussian cluster composed by equivalent models (each cluster point is a model), whose mean vector $\bar{\theta}_\Psi$ and covariance matrix $S_\Psi$ can be estimated on $\Theta_L$ (Line 1). For cognitive diagnosis purposes we assign to the nominal state also the number $n_\Psi$ of parameter vectors used to estimate $\bar{\theta}_\Psi$ and $S_\Psi$ and the last time instant $t_\Psi$ for which a $\hat{\theta}_i$ was associated to the nominal state $\Psi$. At the end of the training phase $n_\Psi = L$ and $t_\Psi = L$ (Line 2). The extension to multi-class nominal states, e.g., representing different regimes of $\mathcal{P}$, would require considering a set of Gaussian clusters for $\Psi$.

During the operational life, the proposed FDS estimates parameter vectors from incoming not-overlapping $N$-sample data windows. The corresponding $\hat{\theta}_i$s are then either associated to the nominal state $\Psi$ or a generic $j$-th faulty one $\Phi_j$ present in the fault dictionary $\Phi = \{\Phi_1, \dots, \Phi_\phi\}$ ($\phi$ represents the number of current fault classes in the fault dictionary). If the assignment cannot be granted according to a given confidence level the parameter vector is currently considered to be an outlier and moved to the outlier set $O$.

At the beginning, both the fault dictionary and the outlier set are empty (Figure 1a) and are populated during the operational phase, as data come in. The outlier set is regularly inspected to determine whether a new state $\Phi_{\phi+1}$ has been there contained and needs to be generated (Figure 1b). If a parameter vector cannot be associated to either the nominal state or one of the faulty states according to the given confidence level, it is considered an outlier and moved to the outlier set $O$ (e.g., see the asterisks near the ellipse in the upper-right side of Figure 1d). Similarly, other "housekeeping" operations are executed on the existing structures (outlier and faulty sets), e.g., leading to the merge of two faulty states, whenever appropriate.

Details about the cognitive FDS algorithm are given in the sequel, while the creation of the fault set deserves a deeper discussion (Section IV).




(a) The nominal state (crosses) is characterized during the training phase

(b) The number of outliers (asterisks) is increasing but no faults are identified yet

(c) As soon as enough confidence is gathered for the presence of a new faulty state, a new cluster is created (circles) and instances added to it

(d) When a different fault is identified (dots), it is added to the fault dictionary.

Fig. 1. Cognitive FDS: an example

Here, we assume that a fault affecting $\theta^o$ abruptly moves the process from a stationary state to a new stationary one (abrupt fault). A faulty state $\Phi_j$ is hence characterized by a mean vector $\bar{\theta}_{\Phi_j}$ and a covariance matrix $S_{\Phi_j}$. The FDS stores the number $n_{\Phi_j}$ of vectors used to estimate $\bar{\theta}_{\Phi_j}$ and $S_{\Phi_j}$ and the latest time instant $t_{\Phi_j}$ where $\hat{\theta}_i$ was associated to $\Phi_j$.

The distance between a parameter vector $\hat{\theta}_i$ and the center of a cluster can be computed by means of the Mahalanobis distance:

$$m(\hat{\theta}_i, \Upsilon) = (\bar{\theta}_\Upsilon - \hat{\theta}_i)^T S_\Upsilon^{-1} (\bar{\theta}_\Upsilon - \hat{\theta}_i).$$

where $\Upsilon \in \{\Psi, \Phi_1, \dots, \Phi_\phi\}$. Since $\Psi$ and $\Phi_j \in \Phi$ are Gaussian clusters, a neighbourhood centered in $\bar{\theta}_\Upsilon$ can be induced by containing those $\hat{\theta}_i$s belonging to $\Upsilon$ with probability $1-\alpha_s$ [36], where $\alpha_s$ is a given confidence level. More specifically, the spatial neighbourhood is composed by those $\theta$s for which:

$$\frac{n_\Upsilon (n_\Upsilon - p)}{p(n_\Upsilon^2 - 1)} m(\theta, \Upsilon) \leq F_{p, n_\Upsilon - p, \alpha_s} \tag{3}$$

hold. $F_{p, n_\Upsilon - p, \alpha_s}$ is the Fisher's distribution quantile of order $1 - \alpha_s$ of parameters $p$ and $n_\Upsilon - p$. Similarly, a neighbourhood is assigned to each cluster $\Upsilon \in \{\Psi, \Phi_1, \dots, \Phi_\phi\}$ and constitutes the core of the fault identification phase of the FDS (Lines 6 and 14). The FDS algorithm also contemplates the case of $\hat{\theta}_i$ satisfying Eq. 3 for multiple clusters. In this case, $\hat{\theta}_i$ is associated to the cluster $\Upsilon^*$ (either nominal or faulty, Lines 7 and 15) minimizing:

$$\Upsilon^* = \min_{\Upsilon \in \{\Psi, \Phi_1, \dots, \Phi_\phi\}} \frac{n_\Upsilon (n_\Upsilon - p)}{p(n_\Upsilon^2 - 1)} m(\hat{\theta}_i, \Upsilon). \tag{4}$$

In other words $\hat{\theta}_i$ is assigned to the nearest cluster, provided that confidence $\alpha_s$ is attained. Once $\Upsilon^*$ has been determined, we set $t_{\Upsilon^*} = i$ (Lines 12 and 34). If $\hat{\theta}_i$ cannot be associated either to $\Psi$, or to $\{\Phi_1, \ldots, \Phi_\phi\}$, it is considered to be an outlier and inserted in $O$ (Line 36).

**Algorithm 1:** FDS Evolving Clustering Algorithm

```
1  Compute mean θ̄_Ψ and covariance matrix S_Ψ for the
   nominal state cluster ;
2  Set n_Ψ = L and t_Ψ = L ;
3  Set Φ = ∅ (φ = 0) and O = ∅;
4  Set α_s, η_t;
5  while A new θ̂_i is available do
6      if Eq. 3 holds for at least one Ψ_j ∈ Ψ then
7          Select Ψ* minimizing Eq. 4 ;
8          Associate θ̂_i to Ψ* ;
9          if |t_Ψ − i| ≤ η_t then
10             Update Ψ as in Eq. 6-8 ;
11         end
12         t_Ψ* ← i ;
13     else
14         if φ > 0 and Eq. 3 holds for at least one Φ_j ∈ Φ
            then
15             Select Φ* minimizing Eq. 4 ;
16             Associate θ̂_i to Φ* ;
17             if |t_Φ* − i| ≤ η_t then
18                 Update Φ* as in Eq. 6-8 ;
19                 for θ̂_h ∈ O do
20                     if Eq. 3 holds for Φ* then
21                         Remove θ̂_h from outlier set O ;
22                         Associate θ̂_h to Φ* ;
23                         if |t*_Φ − h| ≤ η_t then
24                             Update Φ* as in Eq. 6-8 ;
25                         end
26                     end
27                 end
28                 for Φ_j ∈ Φ, Φ ≠ Φ* do
29                     if Eq. 9, 10 hold for Φ*, Φ_j then
30                         Merge Φ*, Φ_j as in Eq. 11-14 ;
31                     end
32                 end
33             end
34             t_Φ* ← i ;
35         else
36             Insert θ̂_i in O ;
37             Create Ō according to Alg. 2 ;
38             if Ō ≠ ∅ then
39                 φ ← φ + 1 ;
40                 Create Φ_φ using θ̂_k ∈ Ō ;
41             end
42         end
43     end
44 end
```

The algorithm, after taking into account the "spatial" locality between parameter vectors, analyzes the "temporal" one, by evaluating to which level recent $\hat{\theta}$s have been associated to $\Upsilon^*$ (Lines 9 and 17), i.e.,

$$|n_{\Upsilon^*} - i| \leq \eta_t \tag{5}$$

where $\eta_t \in \mathbb{N}$ is a temporal threshold (when $\eta_t = 1$ the FDS verifies if two consecutive time vectors $\hat{\theta}_i$ and $\hat{\theta}_{i-1}$ have been assigned to the same cluster). This operation is important since we expect models built over time to be temporally dependent.

If $\hat{\theta}_i$ satisfies both the spatial (Eq. 3) and the temporal (Eq. 5) membership conditions, for cluster $\Upsilon^*$, it is inserted in there and its statistics are updated, since a new instance has been received (Lines 10, 18):

$$\bar{\theta}_{\Upsilon^*} \leftarrow \frac{n_{\Upsilon^*}}{n_{\Upsilon^*}+1}\bar{\theta}_{\Upsilon^*} + \frac{1}{n_{\Upsilon^*}+1}\hat{\theta}_i \tag{6}$$

$$S_{\Upsilon^*} \leftarrow \frac{n_{\Upsilon^*}-1}{n_{\Upsilon^*}}S_{\Upsilon^*} + \frac{n_{\Upsilon^*}+1}{n^2_{\Upsilon^*}}(\hat{\theta}_i - \bar{\theta}_{\Upsilon^*})(\hat{\theta}_i - \bar{\theta}_{\Upsilon^*})^T \tag{7}$$

$$n_{\Upsilon^*} \leftarrow n_{\Upsilon^*} + 1. \tag{8}$$

The aforementioned procedure might update cluster $\Upsilon_j$ so that it partly overlaps with another one $\Upsilon_k$. The algorithm handles the situation with a cluster merging procedure (Lines 28-30). The union of clusters $\Upsilon_j$ and $\Upsilon_k$ is performed if the following two conditions are jointly satisfied,

$$\frac{n_{\Upsilon_j}(n_{\Upsilon_k}n_{\Upsilon_j} - n_{\Upsilon_k} - p + 1)}{(n_{\Upsilon_k}+1)(n_{\Upsilon_j}-1)p} m(\bar{\theta}_{\Upsilon_j}, \Upsilon_k) \leq F_{p, n_{\Upsilon_k}n_{\Upsilon_j} - n_{\Upsilon_k} - p + 1, \frac{\alpha_m}{2}} \tag{9}$$

$$\frac{n_{\Upsilon_k}(n_{\Upsilon_j}n_{\Upsilon_k} - n_{\Upsilon_j} - p + 1)}{(n_{\Upsilon_j}+1)(n_{\Upsilon_k}-1)p} m(\bar{\theta}_{\Upsilon_k}, \Upsilon_j) \leq F_{p, n_{\Upsilon_j}n_{\Upsilon_k} - n_{\Upsilon_j} - p + 1, \frac{\alpha_m}{2}}, \tag{10}$$

i.e., if the cluster means $\bar{\theta}_{\Upsilon_j}, \bar{\theta}_{\Upsilon_k}$ have probability greater than $1 - \alpha_m$ to belong (to be drawn from) each other clusters. In Eq. 9 and Eq. 10, $F_{p, n_{\Upsilon_k}n_{\Upsilon_j} - n_{\Upsilon_k} - p + 1, \frac{\alpha_m}{2}}$ is the Fisher's distribution quantile of order $1 - \alpha_m/2$, with parameters $p$ and $n_{\Upsilon_k}n_{\Upsilon_j} - n_{\Upsilon_k} - p + 1$. Approximated results for the confidence $\alpha_m$ follow from the Bonferroni correction for multiple tests. If the above conditions are satisfied, the FDS merges the two clusters $\Upsilon_j$ and $\Upsilon_k$ to generate cluster $\Upsilon'$ defined as

$$\bar{\theta}_{\Upsilon'} \leftarrow \frac{n_{\Upsilon_j}}{n_{\Upsilon_j}+n_{\Upsilon_k}}\bar{\theta}_{\Upsilon_j} + \frac{n_{\Upsilon_k}}{n_{\Upsilon_j}+n_{\Upsilon_k}}\bar{\theta}_{\Upsilon_k}; \tag{11}$$

$$S_{\Upsilon'} \leftarrow S_{\Upsilon_j} + S_{\Upsilon_k} + \frac{n_{\Upsilon_j}n_{\Upsilon_k}}{n_{\Upsilon_j}+n_{\Upsilon_k}}(\bar{\theta}_{\Upsilon_j} - \bar{\theta}_{\Upsilon_k})(\bar{\theta}_{\Upsilon_j} - \bar{\theta}_{\Upsilon_k})^T; \tag{12}$$

$$n_{\Upsilon'} \leftarrow n_{\Upsilon_j} + n_{\Upsilon_k}; \tag{13}$$

$$t_{\Upsilon'} \leftarrow \max\{t_{\Upsilon_j}, t_{\Upsilon_k}\}. \tag{14}$$

The exact computation of the update for the covariance matrix is performed as in [37].

After a cluster update or the merge of two clusters, the proposed FDS checks if parameter vectors in the outlier set $O$ can now be associated either to the nominal state or one of the faulty ones (Lines 19-24).

## IV. On-line characterization of the fault dictionary

We addressed so far the procedure allowing the insertion of the parameter vectors in the nominal and faulty clusters and the merging of two faulty clusters. The remaining $\hat{\theta}_i$ are collected in the outlier set $O$, where further inspection is performed during the operational phase, to verify whether a new faulty state must be created or not.

With reference to Alg. 2, a new cluster needs to be created depending on the outcome of the Kolmogorov-Smirnov (KS) test (Line 4). The test compares the empirical Cumulative Distribution Function (CDF) of all the $\hat{\theta}$s estimated by the FDS during both the training and the operational phases and the CDF induced by considering the estimated nominal state $\Psi$ and faults $\{\Phi_1, \ldots, \Phi_\phi\}$. If the distribution of the $\hat{\theta}$s is no more coherent with the current set of clusters, a new cluster must be created and a new fault class inserted in $\Phi$. More in detail, the test is designed as:

$$H_0 : \hat{F} = F_\Gamma \text{ vs. } H_1 : \hat{F} \neq F_\Gamma$$

where $\hat{F}$ is the empirical CDF of all the $\hat{\theta}$s and $F_\Gamma$ is the distribution induced by Gaussian clusters $\Gamma = \{\Psi, \Phi_1 \ldots \Phi_\phi\}$. The KS test statistics takes into account the maximum distance between the two CDFs

$$D^p = \max_{0 \leq \alpha \leq 1} |\hat{F}(B_\alpha) - F_\Gamma(B_\alpha)|,$$

where $B_\alpha$ is the region in the parameter space such that $F_\Gamma(B_\alpha) = \alpha$ (see [38] for further details). As stated in [38], $D^p$ has the same distribution of the monodimensional KS distribution, so, for the KS-test, we can compare it with the asymptotic form of the KS distribution $K$ [39], [40]. Given a confidence level $\alpha_c$, the critical region of the KS test (i.e., for rejecting the null hypothesis $H_0$) is:

$$D^p > K_{\alpha_c} \tag{15}$$

where $K_{\alpha_c}$ is the quantile of order $1 - \alpha_c$ of the monodimensional $K$ distribution. The proposed statistical test suffers from the curse of dimensionality, i.e., it needs an exponentially increasing number of samples to be effective as the parameter vector dimension $p$ increases. Therefore, if needed, we suggest to apply a dimensionality reduction method to the parameter vectors $\hat{\theta} \in O$, e.g., based on Principal Component Analysis (PCA) [36] or Random Projection (RP) method [41].

Once the KS-test provides enough confidence to claim that a new cluster must be generated from the outlier set (i.e., hypothesis $H_0$ is rejected), suitable instances are removed from $O$ and the new cluster is created. We assume the availability of a supervisor that is able to label new faulty clusters, e.g., by providing the type of encountered fault. This allows us for creating online the fault dictionary. On the contrary, when the hypothesis $H_0$ is not rejected, Alg. 2 returns an empty set (Line 32).

It is worth noting that the Mahalanobis distance cannot be considered to measure parameter vector proximities in $O$, since the distribution of elements in the outlier set is unknown (i.e., we cannot assume that $\hat{\theta}$s $\in O$ are Gaussian distributed as they are not). To address this issue we defined the spatial-temporal norm on $\hat{\theta}_h, \hat{\theta}_j \in O$, inspired by the metric suggested in [42]:

$$||\hat{\theta}_h - \hat{\theta}_j||^2_\lambda = \lambda \frac{||\hat{\theta}_h - \hat{\theta}_j||^2}{2p} + (1 - \lambda) \frac{|h - j|}{i}$$

where $||\cdot||$ is the euclidean norm, $i$ is the last batch of data considered and $\lambda \in [0, 1]$ is a penalty factor balancing the spatial locality and the temporal one. A normalization procedure is required so that both the spatial and temporal components of the norm are constrained to the $[0, 1]$ interval. The FDS algorithm adopts the online normalization procedure described in [43].

To select parameter vectors for the new clusters, we adopted the *Mountain Method* [44]–[46], which identifies the density center for the $\hat{\theta}$s $\in O$ (Lines 7-12). Finally, this algorithm estimates the density as:

$$\Omega_{RMM}(c_j, \hat{\theta}_h; r) = \exp\left(-\frac{||\hat{\theta}_h - c_j||^2_\lambda}{2r^2}\right)$$

where $c_j \in \mathbb{R}^p$ is a center and $r$ is an influence radius parameters. The algorithm iteratively approximates:

$$c^* = \max_c \sum_{\hat{\theta}_h \in O} \Omega_{RMM}(c, \hat{\theta}_h, r).$$

The potential function $\Omega_{RMM}$ is robust to outliers (see [46] for a formal proof) and, since it decreases slowly when $||\hat{\theta}_h - c_j||_\lambda < r$ and fast if $||\hat{\theta}_h - c_j||_\lambda > r$, it defines a neighborhood around each class center $c_j$. For the purpose of the cluster creation a center will be initialized for each of the parameter vectors $\hat{\theta}_h$s $\in O$. As described in [44], $\eta_i$ of Alg. 2 represents both a tolerance threshold for the convergence of the iterative procedure to identify the cluster center and the maximum error of the optimization procedure. As one might imagine the method is rather sensitive to $r$, which highly influences the clustering results. Here, we suggested three different heuristics to identify a suitable value for the radius $r$:

- power estimate using correlation [46];
- median distance criterion [45];
- maximum edge length of minimum spanning tree under the normal distribution hypothesis [47].

At the end of the mountain method each parameter vector is associated with a set $O_s$ (Lines 14-15) and $\tilde{O}$, the set characterized by the largest cardinality, is selected as a new candidate cluster.

To identify the cluster shape of $\tilde{O}$ (we do not have a priori information about the covariance matrix of the novel cluster), a *Minimum Covariance Determinant* search method [48] is executed (Lines 16-29), i.e., a subset of elements $\bar{O} \subseteq \tilde{O}$ is selected s.t. the determinant of the parameter covariance is minimal. This method can be applied when the number of samples in $\tilde{O} \geq p$. When this condition is satisfied (Line 16), a new cluster is created: the mean and the covariance of the parameter vectors in $\bar{O}$ are computed, $n_{\Phi_{\phi+1}} = |\bar{O}|$, $t_{\Phi_{\phi+1}} = \max_{\hat{\theta}_h \in \bar{O}} h$ and the algorithm returns $\bar{O}$ (Line 27). Otherwise, when $\tilde{O} < p$, the algorithm returns the empty set $\emptyset$ (Line 29).

Note that the algorithm requires at least $n_\Upsilon = p + 1$ parameter vectors to create a cluster. More parameter vectors would allow a better characterization of the cluster itself since the variance of the estimation of the mean and the covariance matrix scales asymptotically as $\frac{1}{n_\Upsilon}$. Moreover, as time passes, more and more parameter vectors are inserted into the outlier set. To reduce as much as possible the creation of false classes we should consider an oblivion coefficient on the parameter vectors in the outlier set or mechanisms to discard the oldest ones (e.g., by setting a maximum value on the cardinality of the outlier set and keeping the new ones). The algorithm can be easily modified to take into account this case.

**Algorithm 2:** Fault cluster creation

Given an outlier set $O$
Set $\alpha_c \in (0,1)$, $\eta_i$;
Compute $D^p$ according to Eq. 15 ;
**if** $D^p > K_{\alpha_c}$ **then**
  Set $c_h = \hat{\theta}_h,\ \forall \hat{\theta}_h \in O$ and $err \geq \eta_i$;
  Compute $r$;
  **while** $err \geq \eta_i$ **do**
    **for** $j$ *s.t,* $\hat{\theta}_h \in O$ **do**
      $\hat{c}_j \leftarrow c_j$;
      $c_j \leftarrow \frac{\sum_{\hat{\theta}_h \in O} \Omega_{RMM}(c_j, \hat{\theta}_h; r)\hat{\theta}_h}{\sum_{\hat{\theta}_h \in O} \Omega_{RMM}(c_j, \hat{\theta}_h; r)}$;
    **end**
    $err = \max_k ||\hat{c}_j - c_j||_\lambda$;
  **end**
  Associate all centers $c_j, c_h$ s.t. $||c_j - c_h||_\lambda \leq 2\eta_i$ to a set, creating the sets $O_1, \ldots, O_S$ ;
  Let $\tilde{O} = \arg\max_{s \in \{1,\ldots,S\}} |O_s|$, i.e., the set with the largest cardinality ;
  **if** $|\tilde{O}| \geq p$ **then**
    Choose randomly $h = \frac{|\tilde{O}|+p+1}{2}$ elements in $\tilde{O}$ to define $\bar{O}$;
    Set $S^* = \sum_{\hat{\theta}_k \in \bar{O}} \frac{(\hat{\theta}_k - c^*)(\hat{\theta}_k - c^*)^T}{h-1}$;
    **while** $\bar{O} \neq \bar{O}'$ **do**
      $\bar{O}' \leftarrow \bar{O}$ ;
      **for** $\hat{\theta}_h \in \bar{O}$ **do**
        $d(\hat{\theta}_k) = (\hat{\theta}_k - c^*)(S^*)^{-1}(\hat{\theta}_k - c^*)^T$;
      **end**
      $\bar{O} \leftarrow \arg\min_{O' \subseteq \tilde{O}, |O'|=h} \sum_{\hat{\theta}_k \in O'} d(\hat{\theta}_k)$;
      $S^* \leftarrow \sum_{\hat{\theta}_k \in \bar{O}} \frac{(\hat{\theta}_k - c^*)(\hat{\theta}_k - c^*)^T}{h-1}$;
    **end**
    Return $\bar{O}$ ;
  **else**
    Return $\emptyset$;
  **end**
**else**
  Return $\emptyset$ ;
**end**

## V. Experimental Results

The aim of this section is to evaluate the effectiveness of the proposed cognitive FDS. As we have seen, each state of the process (either nominal or faulty) is a cluster of parameter vectors: creation of the right number of clusters refers to the ability of the method to correctly identify the number of states the process explores. Likewise, an accurate aggregation of parameter vectors coming from the same state refers to the ability of correctly characterizing the operational state. As described in Section I, and to the best of our knowledge, no cognitive fault diagnosis systems able to characterize the fault dictionary during the operational life are available in the literature. As a consequence, to compare the performance of the FDS, we consider algorithms designed to group unlabeled data, a task commonly addressed by clustering methods. We consider both off-line clustering algorithms, such as the DBScan (DBS) [49], the Affinity Propagation (AP) [50], and the Evolving Clustering Algorithm (ECM) [24]. DBS and AP process the whole dataset and do not require a-priori information about the number of clusters to be created, hence representing a relevant reference for the proposed FDS. On the contrary, ECM manages clusters with evolving strategies; the drawback here is that it requires parameter $D_{thr}$, which is strictly related to the number of clusters the algorithm will create during the operational life (such information is obviously unknown in real applications).

To evaluate the performance of the suggested method, we consider the following figures of merit:

- $n_c$: the number of created clusters. It represents the number of states detected by the algorithm. When $n_c$ equalizes the correct number of states the algorithm operates well;
- $r$: the percentage of experiments where the algorithm creates the correct number of clusters. Large values of $r$ suggest that the fault diagnosis method is able to correctly characterize the number of process states;
- $a$: the accuracy in associating a parameter vector to the correct cluster. It represents the ability to correctly identify the state in which the process is operating;
- $p_o$: the percentage of outliers, i.e., the percentage of parameter vectors which cannot be associated to any state. Large values of $p_o$ imply that the algorithm is not able to associate parameter vectors to any cluster.

It is worth mentioning that the FDS requires an initial training phase. The FDS is trained on the training set and tested on a separate test set, while DBS, AP and ECM are applied to the whole training + test set (but their performance are evaluated only on the test set). Since ECM and AP do not generate outliers, $p_o$ is not provided for them.

We considered two different hierarchies of model family $\mathcal{M}(\theta)$:

- the auto-regressive with exogenous input $ARX(na, nb)$ linear model family, where $na$ and $nb$ are the autoregressive and exogenous orders, respectively. Here, the $p = na + nb$ dimensional parameter vector $\theta \in \mathbb{R}^p$ is:

$$\theta = (\theta_1 \ldots \theta_{na}\ \theta_{na+1} \ldots \theta_{na+nb});$$

- the *Reservoir Network* (RN) [51] model defined as:

$$x(t) = g\left(Wx(t-1) + W_{in}u(t)\right)$$
$$\hat{y}(t) = \theta x(t)$$

where $\hat{y}(t) \in \mathbb{R}$ is the prediction value at time $t \in \mathbb{N}$, $u(t) \in \mathbb{R}^m$ is the input observation vector at time $t$, $x(t) \in \mathbb{R}^p$ is the internal state of the network at time $t$, $W \in \mathbb{R}^{p\times p}$ is the internal weight matrix and $W_{in} \in \mathbb{R}^{p\times m}$ is an input weight matrix, both randomly chosen. $g : \mathbb{R}^p \to \mathbb{R}^p$ is an activation function (e.g., $g_i(\cdot) = \tanh(\cdot), i \in \{1, \ldots, p\}$) and $\theta \in \mathbb{R}^p$ is an output weight vector to be learned (model parameter vector).

We considered ARX and RN model families since they satisfy the hypotheses required by the theoretical framework described in Section II. The structural risk is the squared error; the Bayesian Information Criterion [52] was considered to identify model orders. In the following, batches of $N = 400$ not overlapping data are considered to estimate the parameters of the approximating models. The proposed FDS has been developed in `MATLAB` and can be freely downloaded from [53] and [54].

The performance of the proposed FDS system has been compared with those of DBS, AP and ECM methods applied to three different applications: a synthetic one, a simulation of the Barcelona Water Distribution Network (BWDN) and a real-world application related to rock collapse forecasting. On the aforementioned applications, faults are of abrupt type as requested by the proposed FDS. Three applications are detailed in the sequel

### A. General remarks on the FDS method

Key parameters describing the FDS algorithm are given in Tab. I. In particular,

- the *spatial confidence* $\alpha_s$ has been set to 0.03 (see Eq. 3). This parameter controls the rate of structural outliers generated by the FDS. Large values of $\alpha_s$ would create more compact clusters and be sensitive to new states, at the expenses of a larger outlier set. On the contrary, small values of $\alpha_s$ would reduce the number of outliers at the expenses of a reduced sensitivity in identifying new states;
- the *temporal threshold* $\eta_t$ has been set to 1, meaning that cluster statistics in Eq. 8 are updated when two consecutive parameter vectors are inserted in the same cluster. Larger values of $\eta_t$ update the cluster statistics with less restrictive conditions. $\eta_t = 1$ represents a conservative choice for this parameter;
- the *merging confidence* $\alpha_m$ has been set set to 0.05. It represents the confidence of the hypothesis test designed to assess whether two clusters need to be merged or not;
- the *cluster creation confidence* $\alpha_c$ has been set to 0.1. It represents the confidence of the KS hypothesis test, meant to assess if a new cluster must be created by looking at the distribution of the parameter vectors in the outlier set.

Since the FDS creates clusters with as low as $p + 1$ parameter vectors (required by the minimum covariant determinant procedure, see Sec. IV for details), we set the DBS parameter $minPts$ to $p + 1$ to have a fair comparison. Parameter $\varepsilon$ of DBS has been set by using the heuristics described in [49]. The ECM parameter $D_{thr}$ was set to 0.1, as suggested in [24].

| | |
|---|---|
| Spatial confidence | $\alpha_s = 0.03$ |
| Temporal threshold | $\eta_t = 1$ |
| Merging confidence | $\alpha_m = 0.05$ |
| KS-test confidence | $\alpha_c = 0.1$ |
| Spatial-temporal penalization | $\lambda = 0.5$ |
| Mountain method threshold | $\eta_i = 10^{-6}$ |

TABLE I
PARAMETERS OF THE PROPOSED FDS

### B. APP D1: Synthetic application

Synthetic data are generated according to model:

$$y(t) = \sin\left(a_1 y(t-1) + a_2 y(t-2) + b_1 u(t-1)\right) + d(t) \quad (16)$$

where $a_1 = 0.1$, $a_2 = 0.2$, $b_1 = -0.1$, $d(t) \sim \mathcal{N}(0, 10^{-4})$. The exogenous input follows the model:

$$u(t) = 0.4u(t-1) + \epsilon(t),$$

with $\epsilon(t) \sim \mathcal{N}(0, 1)$.

The length of each experiment is 60300 samples with the first 24120 ones used to train the FDS. Faults affecting the system have been modeled as abrupt changes in the parameters of Eq. 16. This models the situation where a fault affecting the system induces a change in the dynamics of the relationship between input and output. The first fault affects the system in sample interval $[24120, 36180]$, inducing an abrupt change which shifts the parameters from $\theta = (a_1\ a_2\ b_1)$ to $\theta_\delta = (1+\delta)\theta$, $\delta$ being a positive scalar controlling the intensity of the perturbation. Afterwards, the data-generating process returns to the nominal state. Then, another fault affects the system in sample interval $[48240, 60300]$, inducing a change in the parameters from $\theta$ to $\theta_\delta = (1-\delta)\theta$. As a consequence, the total number of states for this application is three (i.e., the nominal state and the two faulty ones). We considered different scenarios for this application by taking into account abrupt changes in the parameters with magnitude $\delta$ ranging from 0.01 to 0.3. For each scenario, we generated 200 experiments; averaged results are presented in Tables II and III.

In particular, Tab. II shows the number of created clusters $n_c$ for the considered algorithms, model hierarchies and fault magnitudes. As expected, the ability to create the correct number of clusters increases with the magnitude of $\delta$ (a strong fault is easy to be identified). Interestingly, the *FDS with ARX* is able to correctly create three clusters even with very low fault magnitudes (e.g., $\delta = 0.015$). *ECM* creates an excessive number of clusters making this algorithm not useful in this application. The reason of this behaviour resides in the incorrect setting of $D_{thr}$ [49]. Unfortunately, as explained above, it is hard to set this parameter for fault diagnosis purpose, since it is related to the number of clusters to be created, which is obviously unknown a-priori. Interestingly, both *DBS* and *AP with ARX* are able to create the correct number of clusters, for large $\delta$ magnitudes, i.e., $\delta \geq 0.05$ and $\delta \geq 0.3$, respectively. Despite the evolving approach,

| | | Fault | $n_c$ | | | |
|---|---|---|---|---|---|---|
| | | | FDS | ECM | DBS | AP |
| APP D1 (SYN) | ARX | $\delta = 0.010$ | 1.8(0.8) | 120.1(3.6) | 1.0(0.1) | 12.7(1.1) |
| | | $\delta = 0.015$ | 3.3(0.5) | 118.3(3.7) | 1.0(0.2) | 12.7(1.1) |
| | | $\delta = 0.020$ | 3.2(0.4) | 113.8(3.7) | 1.1(0.4) | 12.9(1.1) |
| | | $\delta = 0.025$ | 3.2(0.5) | 108.8(3.7) | 1.8(0.7) | 12.9(1.1) |
| | | $\delta = 0.050$ | 3.2(0.4) | 88.5(3.7) | 3.0(0.2) | 11.6(1.1) |
| | | $\delta = 0.100$ | 3.2(0.5) | 60.7(3.3) | 3.0(0.0) | 7.1(0.8) |
| | | $\delta = 0.200$ | 3.2(0.4) | 33.8(2.8) | 3.0(0.0) | 3.8(0.4) |
| | | $\delta = 0.300$ | 3.1(0.3) | 21.5(2.2) | 3.0(0.0) | 3.0(0.0) |
| | RN | $\delta = 0.010$ | 1.0(0.1) | 89.4(20.6) | 1.0(0.0) | 11.2(2.0) |
| | | $\delta = 0.015$ | 1.0(0.2) | 90.2(21.2) | 1.0(0.1) | 11.0(2.0) |
| | | $\delta = 0.020$ | 1.0(0.2) | 88.0(22.5) | 1.0(0.1) | 11.2(2.1) |
| | | $\delta = 0.025$ | 1.1(0.4) | 90.6(22.0) | 1.0(0.0) | 11.1(1.9) |
| | | $\delta = 0.050$ | 1.5(0.8) | 88.8(22.1) | 1.0(0.2) | 10.9(2.0) |
| | | $\delta = 0.100$ | 2.4(1.0) | 84.9(18.9) | 1.2(0.5) | 10.1(2.2) |
| | | $\delta = 0.200$ | 2.7(0.9) | 68.7(22.0) | 1.9(1.0) | 8.1(2.9) |
| | | $\delta = 0.300$ | 2.6(0.9) | 52.1(23.3) | 2.3(0.9) | 6.6(3.1) |
| APP D2 (BWDN) | ARX | $BW1$ | 2 | 38 | 1 | 4 |
| | | $BW2$ | 3 | 57 | 1 | 6 |
| | | $BW3$ | 4 | 47 | 2 | 4 |
| | | $BW4$ | 5 | 59 | 2 | 6 |
| APP D3 (RIALBA) | ARX | $R1$ | 2 | 48 | 1 | 10 |
| | | $R2$ | 3 | 39 | 1 | 8 |

TABLE II
NUMBER OF CLUSTERS CREATED FOR THE CONSIDERED APPLICATIONS. AVERAGE VALUES IS GIVEN; STANDARD DEVIATION IN BRACKETS.

the proposed *FDS* with *ARX* is more effective in creating the correct number of clusters once compared with non-evolving algorithms such as the DBS and the AP even for small $\delta$s. The rationale behind this refers to the fact that the FDS is able to simultaneously consider both temporal and spatial dependencies among parameter vectors.

RNs provide lower performance than ARX. The reason of this behaviour can be associated to the fact that the performance of RNs is highly influenced by the choice of the random network topology. In fact, training the network topology is entirely based on nominal state samples. This leads to a RN modeling the nominal state, but does not necessarily guarantee the ability to identify new states during the operational life. However, the ability to create the correct number of clusters increases with $\delta$ and the *FDS* with *RN* is able to identify the correct number of faults with magnitude $\delta \geq 0.2$.

Then, in order to evaluate the ability to correctly identify the states where the process operates over time, we focus only on those experiments for which the number of created clusters is correct. Tab. III shows $r$, $a$ and $p_0$ for ARX and RNs and fault magnitudes, when the number of clusters created by the algorithm is correct (i.e., $n_c = 3$ for APP D1).

As expected, the FDS improves its performance both in terms of percentage of experiments which identified the correct number of clusters $r$ and in terms of the classification accuracy $a$ as the magnitude of the fault increases. Interestingly, when $\delta < 0.015$ the *FDS* with *ARX* reduces its effectiveness in the clustering (i.e., $r = 12.0\%$ and $a = 51.7\%$) meaning that small fault magnitudes represent challenging situations for the proposed approach. In our opinion, this behaviour is due to the fact that the neighbourhood of probability $1 - \alpha_s$ induced by the covariance matrix $\Sigma_N$ includes also some faulty states when $\delta \leq 0.01$.

Furthermore, the analysis of $p_0$ allows us to evaluate the effect of the choice of the FDS parameters on performance. Specifically, as explained in the previous section, the parameter $\alpha_s$ controls the percentage of structural outliers of the FDS and this is particularly evident by the values of $p_o$ in Table III which are in line with what expected from the theory. On the contrary, the percentage of outliers of DBS decreases, when the fault magnitude increases. This is reasonable since the method does not contemplate a fixed percentage of structural outliers.

By inspecting accuracy $a$, we see that the *FDS* with *ARX* provides higher performance than the one with *RN* in the small perturbation case ($\delta \leq 0.025$). As the magnitude $\delta$ increases, there is no strong evidence for selecting a specific model family. Nevertheless, the standard deviation of the accuracy of ARX model is lower than the RN model one. This behaviour is in line with the difficulties in selecting the RN topology following the discussions given for Tab. II.

As expected, non evolving clustering algorithms like DBS or AP provide higher performance than FDS when $\delta = 0.3$. This is reasonable since these algorithms work in an off-line way, by analyzing the whole dataset at once. On the contrary, the proposed FDS provides better performance than DBS and AP with small values of $\delta$, making it suitable to manage subtle and not evident faults. Even in this case, the reason of this behaviour resides in the ability of the method to exploit temporal dependencies among parameter vectors during the operational life (while not evolving algorithms do not exploit time dependencies in the clustering phase). ECM was never able to correctly identify the number of clusters, in line with comments following Tab. II.

| | | Fault | FDS | | | ECM | | DBS | | | AP | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | $r$ | $a$ | $p_o$ | $r$ | $a$ | $r$ | $a$ | $p_o$ | $r$ | $a$ |
| APP D1 (SYN) | ARX | $\delta = 0.010$ | 12.0 | 51.7(9.3) | 3.4(3.8) | 0.0 | N.a. | 0.0 | N.a. | N.a. | 0.0 | N.a. |
| | | $\delta = 0.015$ | 72.5 | 84.5(9.6) | 2.7(2.8) | 0.0 | N.a. | 0.0 | N.a. | N.a. | 0.0 | N.a. |
| | | $\delta = 0.020$ | 82.0 | 95.4(3.5) | 3.1(3.3) | 0.0 | N.a. | 1.5 | 55.9(12.0) | 6.7(2.9) | 0.0 | N.a. |
| | | $\delta = 0.025$ | 83.5 | 96.5(2.8) | 3.0(2.8) | 0.0 | N.a. | 13.0 | 92.1(7.3) | 5.6(2.4) | 0.0 | N.a. |
| | | $\delta = 0.050$ | 82.0 | 96.2(3.4) | 3.5(3.4) | 0.0 | N.a. | 97.5 | 97.6(1.8) | 2.4(1.8) | 0.0 | N.a. |
| | | $\delta = 0.100$ | 81.5 | 96.7(3.1) | 2.6(3.1) | 0.0 | N.a. | 100.0 | 99.4(0.8) | 0.6(0.8) | 0.0 | N.a. |
| | | $\delta = 0.200$ | 86.5 | 96.5(2.9) | 2.6(2.8) | 0.0 | N.a. | 100.0 | 100.0(0.0) | 0.0(0.2) | 17.5 | 100.0(0.0) |
| | | $\delta = 0.300$ | 88.0 | 96.8(2.8) | 2.4(2.8) | 0.0 | N.a. | 100.0 | 100.0(0.0) | 0.0(0.0) | 100.0 | 100.0(0.0) |
| | RN | $\delta = 0.010$ | 0.0 | N.a. | N.a. | 0.0 | N.a. | 0.0 | N.a. | N.a. | 0.0 | N.a. |
| | | $\delta = 0.015$ | 0.0 | N.a. | N.a. | 0.0 | N.a. | 0.0 | N.a. | N.a. | 0.0 | N.a. |
| | | $\delta = 0.020$ | 0.5 | 66.7(0.0) | 1.1(0.0) | 0.0 | N.a. | 0.0 | N.a. | N.a. | 0.0 | N.a. |
| | | $\delta = 0.025$ | 3.5 | 63.0(23.0) | 3.0(3.0) | 0.0 | N.a. | 0.0 | N.a. | N.a. | 0.0 | N.a. |
| | | $\delta = 0.050$ | 16.5 | 80.9(16.8) | 3.7(3.4) | 0.0 | N.a. | 1.5 | 97.8(1.1) | 2.2(1.1) | 0.0 | N.a. |
| | | $\delta = 0.100$ | 45.0 | 90.9(10.5) | 3.9(4.4) | 0.0 | N.a. | 6.5 | 97.9(2.6) | 2.0(2.7) | 0.0 | N.a. |
| | | $\delta = 0.200$ | 60.5 | 94.3(7.0) | 3.4(4.6) | 0.0 | N.a. | 38.5 | 98.5(3.6) | 1.1(1.7) | 4.0 | 100.0(0.0) |
| | | $\delta = 0.300$ | 58.5 | 95.3(6.8) | 3.0(3.5) | 0.0 | N.a. | 63.5 | 99.3(1.5) | 0.7(1.5) | 25.5 | 100.0(0.0) |
| APP D2 (BWDN) | ARX | $BW1$ | 100 | 95.5 | 0.0 | 0 | N.a. | 0 | N.a. | N.a. | 0 | N.a. |
| | | $BW2$ | 100 | 81.8 | 0.0 | 0 | N.a. | 0 | N.a. | N.a. | 0 | N.a. |
| | | $BW3$ | 100 | 81.5 | 0.0 | 0 | N.a. | 100 | 56.9 | 4.0 | 0 | N.a. |
| | | $BW4$ | 100 | 80.5 | 3.4 | 0 | N.a. | 0 | N.a. | N.a. | 0 | N.a. |
| APP D3 (RIALBA) | ARX | $R1$ | 100 | 90.6 | 3.8 | 0 | N.a. | 0 | N.a. | N.a. | 0 | N.a. |
| | | $R2$ | 100 | 92.5 | 0.0 | 0 | N.a. | 0 | N.a. | N.a. | 0 | N.a. |

TABLE III
EXPERIMENTAL RESULTS FOR THE CONSIDERED APPLICATIONS. AVERAGE VALUE IS GIVEN; STANDARD DEVIATION IN BRACKETS.

We also performed a robustness analysis to evaluate the effects of variations of the main parameters of the FDS, i.e., $\alpha_s$, $\alpha_m$, $\eta_t$ and $\lambda$, on the considered figures of merit. In the considered scenario APP D1, which is characterized by a stationary process affected by abrupt changes, parameter $\alpha_s$ revealed to be the most sensitive one and its behaviour is deeply investigated in the sequel. Fig. 2a and 2b show how the figures of merit $a$, $p_0$, $r$ and $n_c$ range with $\alpha_s$ ranging in the interval $[2.5E{-}3; 2.5E{-}1]$. As expected, $p_0$ increases with $\alpha_s$ and this is quite obvious since we are creating clusters that are more and more compact. For the considered scenario, $\alpha_s = 0.025$ guarantees the highest value of $r$. Interestingly, lower values of $\alpha_s$ create a reduced number of clusters, while larger ones create an excessive number of clusters. This behaviour is evident by looking at the values of $n_c$ in Fig. 2b. The behaviour of the classification accuracy $a$ is particularly interesting: small values of $\alpha_s$ create very large clusters, hence possibly misclassifying estimated parameters that belong to a different state (e.g., a faulty one); on the contrary, large values of $\alpha_s$ create very small clusters, hence generating many outliers (and this is evident by looking at the behaviour of $p_0$ when $\alpha_s$ increases).

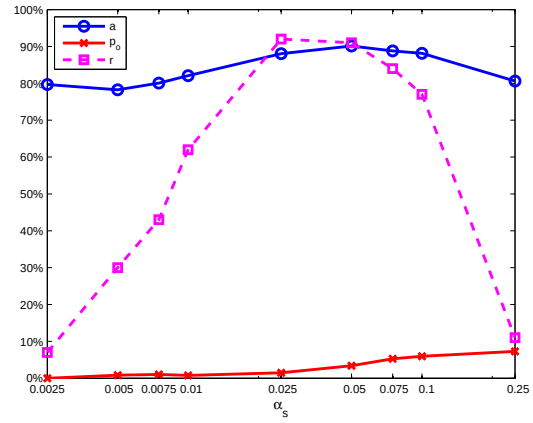
### C. APP D2: Barcelona Water Distribution Network

The second testbed refers to data generated from the Barcelona Water Distribution Network (BWDN) simulator [55]. By relying on a network of 17 tanks, 26 pumps, 35 valves, 9 external sources of the BWDN, this simulator allows to artificially inject faults in a specific flow sensor of the network (i.e., the `iOrioles` pump), by specifying the fault signature, the fault magnitude and the fault time-horizon. Four different scenarios have been considered:

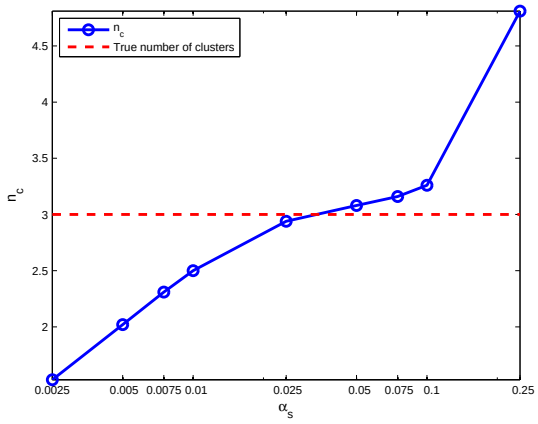
- BW1 An abrupt additive fault affecting the measurements of the `iOrioles` pump is injected in sample interval $[9546, 17472]$. The magnitude of the additive fault is $-20\%$ of the signal dynamic (i.e., the range between the maximum and minimum value of the signal). The length of the dataset is $17472$ samples;
- BW2 A sensor degradation fault is injected in sample interval $[18282, 26208]$. This fault consists in an additive Gaussian noise with zero mean and standard deviation equal to $30\%$ of the signal one. The length of the dataset is $26208$ samples. The first $17472$ samples are equal to the BW1 case;
- BW3 A stuck-at fault is injected in sample interval $[27018, 34944]$. The length of the dataset is $34944$ samples. The first $26208$ samples are equal to the BW2 case;
- BW4 An abrupt additive fault affecting the measurements of the `iOrioles` pump is injected in sample interval $[35754, 43680]$. The magnitude of the additive fault is $20\%$ of the range of the signal. The length of the dataset is $43680$ samples. The first $34944$ samples are equal to the BW3 case.

The FDS has been trained on the first $8736$ samples (representing one year of observations in the BWDN simulator) in all the four considered scenarios; as a reference model we consider the ARX.

Results given in Tab. II are particularly interesting and show how the proposed FDS is able to correctly identify the number of clusters in all the four considered scenarios. All other considered methods do not identify the correct number of clusters (with the exception of AP in the BW3 scenario). In line with the synthetic application experiments, AP and




(a) Average accuracy $a$, outlier percentage $p_o$ and percentage of experiments where the algorithm creates the correct number of clusters $r$ are reported for the experiments where $n_c = 3$.




(b) Average number of cluster $n_c$ created with different values of $\alpha_s$

Fig. 2. Robustness analysis result for $\alpha_s$

DBS usually detect a smaller (1-2) and larger (4-6) number of clusters than necessary, respectively, while ECM creates an excessive number of clusters, i.e., from 38 to 59. These results corroborate the ability of the proposed FDS method to correctly characterize the states explored by the process over time.

In Tab. III the value of $r$ is either 0 or 100, since here we are considering a single experiment. FDS accuracy decreases from $95.5\%$ in BW1 to $81.8\%$ in BW2, while there is no further significant reduction in accuracy in the other scenarios. In our opinion in scenario BW2, the injected degradation fault is particularly hard to be detect, since its effect on the estimated parameter vectors is not as evident as those induced by the other considered faults.

### *D. APP D3: A monitoring system for landslide forecasting*

In this application data are gathered from a monitoring system for landslide forecasting, [56] deployed at the Towers of Rialba site in Northern Italy. The dataset, available at [53], consisting in $35652$ samples, has been acquired in 2011, with a sampling period of 5 minutes. This dataset, which is shown in Figure 3, collects measurements coming from two clinometers.

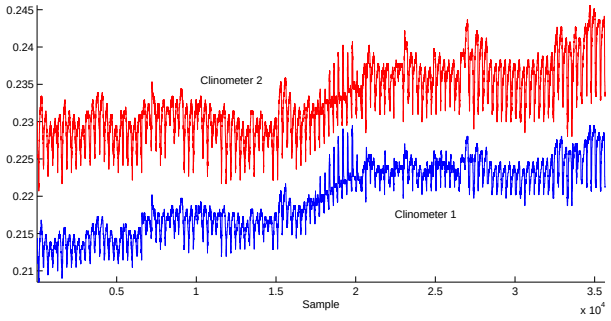



Fig. 3. APP D3: the measurements acquired from two clinometers of the monitoring system deployed at the Towers of Rialba.

Two different scenarios have been considered:

- R1 An abrupt additive fault affecting the measurements of the clinometer, regarded as output, is injected in sample interval $[17468, 24956]$. The magnitude of the additive fault is $-20\%$ of the signal dynamics;
- R2 The first $24956$ samples are equal to the R1 case. Then, a degradation fault is injected in the same clinometer in sample interval $[28164, 35652]$. The degradation fault consists in an additive Gaussian noise with zero mean and standard deviation equal to $30\%$ of the signal one;

We emphasize that, to ease the comparison, R1 and R2 in APP D3 correspond to BW1 and BW2 in APP D2, respectively. In this application, the first $14260$ samples have been used to train the FDS. The chosen model hierarchy was ARX.

Experimental results on this application are particularly interesting since data are coming from a real monitoring system.

By looking at Tab. II, we see that the number of states of the process is correctly recognized by the FDS in both scenarios whereas other methods are never able to create the correct number of clusters. These results are in line with APP D1-D2: AP and ECM are creating more clusters than necessary and DBS is creating a single cluster.

The FDS accuracy in R1 and R2, presented in Tab. III, are similar (i.e., $90.6\%$ in R1 and $92.5\%$ in R2), showing that we are able to deal effectively with multiple faults. With respect to the BWND application, here we do not have a decrease in performance as the degradation fault appears, suggesting that, in this application, the effect of the degradation fault is more easy to be perceived by the FDS.

## VI. Conclusion

The paper presents an evolving mechanism for cognitive fault diagnosis able to detect and cluster faults by characterizing the nominal state and the fault dictionary (initially empty) during the operational phase. The novelty of the proposed approach resides in the evolving mechanisms and the theoretically grounded framework that allows to work in the space of linear approximating models, even if the system under investigation is nonlinear. The cognitive approach allows us to characterize the faults during the operational phase by

introducing clusters in the parameter vector space and updating them in an evolving manner. The experimental section shows the effectiveness of the proposed solution once compared to existing clustering algorithm applied to both synthetic and real data. Results show the better ability of the proposed method over the ones present in the literature to correctly identify the states the process encounters over time.

## ACKNOWLEDGMENT

This research has been funded by the European Commission's 7th Framework Program, under grant Agreement INSFO-ICT-270428 (iSense).

## REFERENCES

[1] R. Isermann. *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer Verlag, 2006.

[2] J. Gertler. *Fault detection and diagnosis in engineering systems*. CRC, 1998.

[3] J.B. Comly, P.P. Bonissone, and M.E. Dausch. *Fuzzy logic for fault diagnosis*. GE Research & Development Center, 1990.

[4] V. Venkatasubramanian, R. Rengaswamy, S.N. Kavuri, and K. Yin. A review of process fault detection and diagnosis:: Part iii: Process history based methods. *Computers & chemical engineering*, 27(3):327–346, 2003.

[5] R. Isermann and P. Ballé. Trends in the application of model-based fault detection and diagnosis of technical processes. *Control Engineering Practice*, 5(5):709–719, 1997.

[6] M.A. Demetriou and M.M. Polycarpou. Incipient fault diagnosis of dynamical systems using online approximators. *Automatic Control, IEEE Transactions on*, 43(11):1612–1617, 1998.

[7] J. Farrell, T. Berger, and BD Appleby. Using learning techniques to accommodate unanticipated faults. *Control Systems, IEEE*, 13(3):40–49, 1993.

[8] M. Fochem, P. Wischnewski, and R. Hofmeier. Quality control systems on the production line of tape deck chassis using self organizing feature maps. In *European Symposium on Applications of Intelligent Technologies (ESIT 97)*, pages 9–13, 1997.

[9] DL Yu, JB Gomm, and D Williams. Sensor fault diagnosis in a chemical process via rbf neural networks. *Control Engineering Practice*, 7(1):49–55, 1999.

[10] A.B. Trunov and M.M. Polycarpou. Automated fault diagnosis in nonlinear multivariable systems using a learning methodology. *Neural Networks, IEEE Transactions on*, 11(1):91–101, 2000.

[11] Yann-Chang Huang. Evolving neural nets for fault diagnosis of power transformers. *Power Delivery, IEEE Transactions on*, 18(3):843–848, 2003.

[12] B.S. Yang, T. Han, and Y.S. Kim. Integration of art-kohonen neural network and case-based reasoning for intelligent fault diagnosis. *Expert Systems with Applications*, 26(3):387–395, 2004.

[13] J Zhang and AJ Morris. On-line process fault diagnosis using fuzzy neural networks. *Intelligent systems engineering*, 3(1):37–47, 1994.

[14] Xinsheng Lou and Kenneth A Loparo. Bearing fault diagnosis based on wavelet transform and fuzzy inference. *Mechanical systems and signal processing*, 18(5):1077–1095, 2004.

[15] Manjeevan Seera, Chee Peng Lim, Dahaman Ishak, and Harapajan Singh. Fault detection and diagnosis of induction motors using motor current signature analysis and a hybrid fmm–cart model. *Neural Networks and Learning Systems, IEEE Transactions on*, 23(1):97–108, 2012.

[16] Yann-Chang Huang, Hong-Tzer Yang, and Ching-Lien Huang. Developing a new transformer fault diagnosis system through evolutionary fuzzy logic. *Power Delivery, IEEE Transactions on*, 12(2):761–767, 1997.

[17] R. Naresh, V. Sharma, and M. Vashisth. An integrated neural fuzzy approach for fault diagnosis of transformers. *Power Delivery, IEEE Trans. on*, 23(4):2017–2024, 2008.

[18] H.C. Kuo and H.K. Chang. A new symbiotic evolution-based fuzzy-neural approach to fault diagnosis of marine propulsion systems. *Engineering Applications of Artificial Intelligence*, 17(8):919–930, 2004.

[19] A. Joentgen, L. Mikenina, R. Weber, A. Zeugner, and H.J. Zimmermann. Automatic fault detection in gearboxes by dynamic fuzzy data analysis. *Fuzzy Sets and Systems*, 105(1):123–132, 1999.

[20] Raj K Aggarwal, QY Xuan, Allan T Johns, Furong Li, and Allen Bennett. A novel approach to fault diagnosis in multicircuit transmission lines using fuzzy artmap neural networks. *Neural Networks, IEEE Transactions on*, 10(5):1214–1221, 1999.

[21] C. Alippi, M. Roveri, and F. Trovò. A learning from models cognitive fault diagnosis system. In *Artificial Neural Networks and Machine Learning–ICANN 2012*, pages 305–313. Springer, 2012.

[22] Rui Xu and Don Wunsch. *Clustering*, volume 10. Wiley-IEEE Press, 2008.

[23] O. Nasraoui and C. Rojas. Robust clustering for tracking noisy evolving data streams. In *Proc. 2006 SIAM Conf. on Data Mining (SDM 2006)*, pages 80–99, 2006.

[24] Q. Song and N. Kasabov. Ecm-a novel on-line, evolving clustering method and its applications. In *Proc. conference on artificial neural networks and expert systems (ANNES2001)*, pages 87–92. Citeseer, 2001.

[25] P.P. Angelov, P. Angelov, D.P. Filev, and N. Kasabov. *Evolving intelligent systems: methodology and applications*, volume 12. Wiley-IEEE Press, 2010.

[26] A. Rosich, R. Sarrate, V. Puig, and T. Escobet. Efficient optimal sensor placement for model-based fdi using an incremental algorithm. In *Decision and Control, 2007 46th IEEE Conference on*, pages 2590–2595. IEEE, 2007.

[27] M. Krysander and E. Frisk. Sensor placement for fault diagnosis. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 38(6):1398–1410, 2008.

[28] Y. Ding, P. Kim, D. Ceglarek, and J. Jin. Optimal sensor distribution for variation diagnosis in multistation assembly processes. *Robotics and Automation, IEEE Transactions on*, 19(4):543–556, 2003.

[29] A. Khan and D. Ceglarek. Sensor optimization for fault diagnosis in multi-fixture assembly systems with distributed sensing. *Transactions of American Society of Mechanical Engineers, Journal of Manufacturing Science and Engineering*, 122(1):215–226, 2000.

[30] C. Alippi, S. Ntalampiras, and M. Roveri. A cognitive fault diagnosis system for distributed sensor networks. *Neural Networks and Learning Systems, IEEE Transactions on*, 24(8):1213 – 1226, 2013.

[31] L. Ljung. *System identification*. Wiley Online Library, 1999.

[32] G.B. Huang, Q.Y. Zhu, and C.K. Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1):489–501, 2006.

[33] B. Schrauwen, D. Verstraeten, and J. Van Campenhout. An overview of reservoir computing: theory, applications and implementations. In *European Symposium on Artificial Neural Networks*. Citeseer, 2007.

[34] L. Ljung and P.E. Caines. Asymptotic normality of prediction error estimators for approximate system models. In *Decision and Control Symposium on Adaptive Processes*, volume 17, pages 927–932. IEEE, 1978.

[35] L. Ljung. Convergence analysis of parametric identification methods. *Automatic Control, IEEE Transactions on*, 23(5):770–783, 1978.

[36] R.A. Johnson and D.W. Wichern. *Applied multivariate statistical analysis*, volume 4. Prentice hall Upper Saddle River, NJ, 2002.

[37] Philippe Pébay. Formulas for robust, one-pass parallel computation of covariances and arbitrary-order statistical moments. *Sandia Report SAND2008-6212, Sandia National Laboratories*, 2008.

[38] R. Saunders and P. Laud. The multidimensional kolmogorov goodness-of-fit test. *Biometrika*, 67(1):237–237, 1980.

[39] A.N. Kolmogorov. Sulla determinazione empirica di una legge di distribuzione. *Giornale dellIstituto Italiano degli Attuari*, 4(1):83–91, 1933.

[40] N. Smirnov. Table for estimating the goodness of fit of empirical distributions. *The annals of mathematical statistics*, 19(2):279–281, 1948.

[41] Avrim Blum. Random projection, margins, kernels, and feature-selection. In *Subspace, Latent Structure and Feature Selection*, pages 52–68. Springer, 2006.

[42] I. Žliobaitė. Combining similarity in time and space for training set formation under concept drift. *Intelligent Data Analysis*, 15(4):589–611, 2011.

[43] S. Jaiyen, C. Lursinsap, and S. Phimoltares. A very fast neural learning for classification using only new incoming datum. *Neural Networks, IEEE Transactions on*, 21(3):381–392, 2010.

[44] R.R. Yager and D.P. Filev. Approximate clustering via the mountain method. *Systems, Man and Cybernetics, IEEE Transactions on*, 24(8):1279–1284, 1994.

[45] C.W. Wu, J.L. Chen, and J.H. Wang. Self-organizing mountain method for clustering. In *Systems, Man, and Cybernetics, 2001 IEEE International Conference on*, volume 4, pages 2434–2438. IEEE, 2001.

[46] M.S. Yang and K.L. Wu. A similarity-based robust clustering method. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(4):434–448, 2004.
[47] M.D. Penrose. Extremes for the minimal spanning tree on normally distributed points. *Advances in Applied Probability*, 30(3):628–639, 1998.
[48] J. Hardin and D.M. Rocke. Outlier detection in the multiple cluster setting using the minimum covariance determinant estimator. *Computational Statistics & Data Analysis*, 44(4):625–638, 2004.
[49] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of KDD*. KDD, 1996.
[50] Frey Brendan J. and Dueck Delbert. Clustering by passing messages between data points. *Science*, 315:972–976, 2007.
[51] H. Jaeger. *Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the" echo state network" approach*. GMD-Forschungszentrum Informationstechnik, 2002.
[52] Gideon Schwarz. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.
[53] Matlab fds toolbox and dataset: http://home.deib.polimi.it/roveri/software, June 2013.
[54] isense project website: http://www.i-sense.org/, June 2013.
[55] E. Caini, V. Puig Cayuela, G. Cembrano Gennari, et al. Development of a simulation environment for water drinking networks: Application to the validation of a centralized mpc controller for the barcelona case study. Technical report, Universitat Politcnica de Catalunya, 2010.
[56] C. Alippi, R. Camplani, C. Galperti, Marullo. A., and M. Roveri. A high frequency sampling monitoring system for environmental and structural applications. *ACM Transactions on Sensor Networks*, 9:41 – 32, 2013.

**Cesare Alippi** (S'92-M'97-SM'99-F'06) received the degree (cum laude) in electronic engineering and the Ph.D. degree from the Politecnico di Milano, Milan, Italy, in 1990 and 1995, respectively. He is currently a Full Professor of information processing systems with the Politecnico di Milano. He has been a Visiting Researcher with University College London, U.K., Massachusetts Institute of Technology, United States, ESPCI, France, and CASIA, China. He has authored or co-authored about 200 papers in international journals and conference proceedings, and holds five patents. His current research interests include adaptation and learning in nonstationary environments and intelligent embedded systems.

Dr. Alippi was a recipient of the IEEE Instrumentation and Measurement Society Young Engineer Award in 2004, the Knight of the Order of Merit of the Italian Republic in 2011. He is the Vice-President Education of the IEEE Computational Intelligence Society and an Associate Editor of the IEEE Computational Intelligence Magazine. He was an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS, the IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENTS from 2003 to 2009, and a member and the Chair of other IEEE committees.

**Manuel Roveri** received the Dr.Eng. degree in computer science engineering from Politecnico di Milano, Milan, Italy, the M.S. degree in computer science from the University of Illinois at Chicago, Chicago, and the Ph.D. degree in computer engineering from the Politecnico di Milano, Milan, in 2003, 2003, and 2007, respectively.

He is currently an Assistant Professor with the Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano. His current research interests include intelligent embedded systems, computational intelligence, and adaptive algorithms.

**Francesco Trovò** received the Dr.Eng. degree in computer science engineering from the Politecnico di Milano, Milan, Italy, in 2011, the M.S. degree in machine learning and data mining from the Aalto University, Helsinki, Finland, in 2011.

Currently, he is a Ph.D. candidate with the Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milan, Italy. His research interests include computational intelligence, cognitive fault diagnosis and machine learning.