2008 International Conference on Information Processing in Sensor Networks

Distributed Online Simultaneous Fault Detection for Multiple Sensors

Ram Rajagopal[1], XuanLong Nguyen[2], Sinem Coleri Ergen[3] and Pravin Varaiya[1]
[1] Electrical Engineering and Computer Sciences, University of California, Berkeley
[2] SAMSI and Dept. of Statistical Science, Duke University
[3] WSN Berkeley Lab, Pirelli and Telecom Italia
{ramr,varaiya}@eecs.berkeley.edu,xuanlong.nguyen@gmail.com,sinem.ergen@wsnlabberkeley.com

Abstract

Monitoring its health by detecting its failed sensors is essential to the reliable functioning of any sensor network. This paper presents a distributed, online, sequential algorithm for detecting multiple faults in a sensor network. The algorithm works by detecting change points in the correlation statistics of neighboring sensors, requiring only neighbors to exchange information. The algorithm provides guarantees on detection delay and false alarm probability. This appears to be the first work to offer such guarantees for a multiple sensor network. Based on the performance guarantees, we compute a tradeoff between sensor node density, detection delay and energy consumption. We also address synchronization, finite storage and data quantization. We validate our approach with some example applications.

1. Introduction

A randomly time-varying environment is monitored by a group of sensors. Each sensor has a fixed location where it periodically collects a noisy sample of the environment. A sensor may fail at any time, after which it reports incorrect measurements. Based on the sensor reports we wish to identify which sensors have failed and when the faults occurred.

If a failed sensor reports measurements with implausible values the fault can be correctly and quickly identified; but if it continues to report plausible values, fault detection is more difficult. We propose fault detection algorithms for this difficult case. The intuitive idea underlying the algorithms is for each sensor to detect a change in the correlation of time series of its own measurements with those of its neighbors' measurements. We call this *change point detection*.

In order for the idea to work, we make two assumptions. First, the measurements of functioning neighboring sensors must be correlated, while the measurements of a faulty sensor and a neighboring functioning sensor are not correlated. Second, since the environment being monitored is time-varying and the measurements are noisy, we require the average time between successive faults to be longer than the *event time scale*–the time between significant changes in the environment. The first assumption helps identification of a faulty sensor by comparing its measurements with its neighbors. Since the identification is made through statistical correlations, the probability of an incorrect fault identification (probability of false alarm) will be positive. The second assumption implies that a change in the environment can be distinguished from a change in the status of sensors, and also that there is sufficient time to reduce the false alarm probability at the cost of a delay in identifying when the fault occurred.

Research supported by California Department of Transportation and ARO-MURI UCSC-W911NF-05-1-0246-VA-09/05

As a concrete example consider the California freeway performance measurement system or PeMS, comprising a collection of 25,000 sensors, one per lane at 9,700 locations [23]. Every five minutes, a sensor reports the number of vehicles that crossed the sensor and the average occupancy or density (vehicles per meter) in the preceding five minutes. If no sensor has failed, these reports are directly used to generate a real-time traffic map on the PeMS website. On any day, however, upwards of 40 percent of the sensors have failed. PeMS uses statistical algorithms to identify the failed sensors and generate the traffic map without their measurements [2]. These algorithms rely on correlating each sensor's measurements with those of its neighbors but, unlike the approach here, they do not use temporal correlation. Also, PeMS algorithms are centralized, whereas ours are distributed as measurements are only communicated among neighbors.

We summarize our contribution. Section 2 reviews related work to our contribution. Section 3 proposes a change point distributed fault model for multiple faults, together with performance metrics to evaluate any sensor fault detection method.

Section 4 presents a distributed, online algorithm for simultaneously detecting multiple faults. The detection procedure relies on online message passing of detection results only among neighboring sensors.

Section 4 gives performance guarantees of the proposed algorithm in terms of the probability of false alarm (PFA) and the detection delay between the instant a fault occurs and the time when the algorithm detects the failure.

Sections 5 and 6 consider the selection of event time scales and propose efficient implementation schemes that minimize the amount of data transfer. Section 7 analyzes node density and fault detection tradeoffs.

2. Related Work

There is a sizable literature on detection in the context of sensor networks [3]. Fault detection of multiple sensors has received some attention [9]. An algorithm to increase

978-0-7695-3157-1/08 $25.00 © 2008 IEEE
DOI 10.1109/IPSN.2008.41

the reliability of a 'virtual' sensor by averaging values of many physical sensors in a fault tolerant manner is presented in [14]. The analysis assumes that each sensor measures the same physical variable with a certain uncertainty and fault specification. In [16], the authors develop a fault tolerant event detection procedure based on the assumption that time-varying failure probabilities of each node are known and a threshold test is used for detection. They also use geographical information to enhance spatial event detection. Decisions are made using only the current time observations, without accounting for trends in the data. [13] proposes a similar model. [6] describes a method for outlier detection based on Bayesian learning. The procedure learns a distribution for interval ranges of the measurements conditional on the neighbor's interval ranges and last observed range. Neighbor's information and past information are assumed conditionally independent when the current range is observed. The idea of detecting malfunctioning sensors based on correlation-type reliability scores among the neighboring sensors is considered in [10]. The model leads to a detection rule based on the posterior probability of the sensor failure given the observed scores at a certain time instance without looking at the time series of measurements. A model-based outlier detection method is developed in [21]. The method relies on estimating a regression model for each individual sensor, and estimating deviations from the predictions of the model. [7] proposes a systematic database approach for data cleansing. A time window primitive for outlier detection based on model estimation is proposed.

A related branch of work lies in the large literature on decentralized detection (see, e.g., [20], [22] for a survey). The main distinction between this line of work and ours is that the former tends to focus on aggregating measurements from multiple sensors to perform test a single hypothesis or conduct an estimation task, whereas our method deals with multiple dependent testing/estimation tasks from multiple sensors. The key technical ingredient in our analysis is drawn from the well-established sequential analysis and sequential change point detection literature [1], [11], but the departure from the traditional formulation of a single change point to a formulation involving multiple correlated change points is novel and theoretically challenging, as well as important in applications.

3. Problem statement

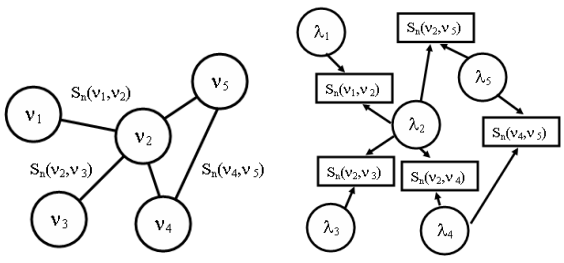

Figure 1. (a) Neighborhood graph of a sensor network and (b) corresponding statistical dependency graph.

3.1. Set-up and underlying assumptions

There are m sensors, labeled u_1 to u_m. Sensor u's measurements form the time series $\{X_t(u)\}$. We are interested in developing an online and distributed procedure for detecting faulty sensors based on the data $\{X_t(u_i) \mid i = 1, \ldots, m\}$. Our method relies on the following assumptions, elaborated further below:

- Neighboring functioning sensors have correlated sensor measurements, but a failed sensor's measurements are not correlated with its functioning neighbors. The neighborhood relationship is specified by the known *fault graph* $\mathcal{G}(V, E)$: V is the set of sensors or nodes and E is the set of undirected edges (Figure 1). The graph normally includes self loops. In practice, the neighborhood relationship is that of geographic proximity. In PeMS, for example, sensors at the same and adjacent locations are considered neighbors.
- Each sensor makes a periodic noisy p-dimensional measurement of its environment. $X_t(u)$ is the measurement at time t. The sensors need not be synchronized.
- Sensors fail independently and for notational simplicity we assume a stationary failure rate of d faults per period. The true failure rate need not be known, but we require a known lower bound. λ_u denotes the random geometrically distributed time node u fails.
- Instead of making a decision at each sampling time t, we choose to make decisions after a block of T samples has been observed. The time scale T is selected to be longer than that of an event. For instance, in PeMS, T corresponds to the number of samples for a day. We index blocks by k and n.

3.2. Performance metrics

A fault detection rule for sensor u is denoted ν_u. Based on the information available at time n, the rule sets $\nu_u = n$ if it decides that u has failed at time n. Thus the random variable ν_u is a stopping time [5]. In the change point literature, such a stopping time is evaluated according to two metrics: probability of false alarm and detection delay, see e.g., [19]:

*Definition 1 (**Probability of false alarm**):* The probability of false alarm of the procedure ν_u is

$$\text{PFA}^{\pi}(\nu) = \sum_{k=1}^{\infty} \pi(k) \mathbb{P}(\nu_u \leq \lambda_u \mid \lambda_u = k).$$

Here λ_u is the true time the change (failure) occurred, and π is the prior distribution of λ_u, $\pi(k) = e^{-dT(k-1)}(1 - e^{-dT})$.

*Definition 2 (**Detection delay**):* The mth moment of the delay of ν_u for change time $\lambda_u = k$ is

$$D_m^{(k)}(\nu_u) = \mathbb{E}_k \left[(\nu_u - k)^m \, | \nu_u \geq k \right]$$

In our Bayesian formulation with prior π, this moment is

$$D_m^{\pi}(\nu_u) = \mathbb{E}_{\lambda} \left[(\nu_u - \lambda_u)^m \, | \nu_u \geq \lambda_u \right] = \sum_{k=1}^{\infty} \pi(k) D_m^{(k)}(\nu_u).$$

A good procedure achieves small (even minimum) delay $D_m^\pi(\nu_u)$, while maintaining $\text{PFA}^\pi(\nu_u) \leq \alpha$, for a pre-specified PFA α.

The key distributed computation constraint requires sensor u's stopping time ν_u to be based only on the scores it shares with its own neighbors. We express this constraint symbolically as $\nu_u \in \mathcal{F}^n_{\mathcal{N}(u)}$.

3.3. Data Preprocessing and Fault Behavior Model

Denote by $\mathbf{X}_n(u)$ the nth observed sample block by sensor u, which has size $T \times p$. Let $\mathcal{H}_{u,n}$ denote data available up to block $n-1$. Each sensor computes a vector score at time n, determined by a transformation F:

$$\mathbf{S}_n(u,u) = F(\mathbf{X}_n(u), \mathcal{H}_{u,n}), \quad (1)$$

$$\mathbf{S}_n(u,u') = F(\mathbf{X}_n(u), \mathbf{X}_n(u'), \mathcal{H}_{u,n}, \mathcal{H}_{u',n}), u' \in \mathcal{N}_u, \quad (2)$$

$\mathcal{N}_u$ is the set of neighbors u' of u. We call $\mathbf{S}_n(u,u')$ the *link score* of the link $(u',u) \in E$. The transformation is symmetric, so $\mathbf{S}_n(u,u') = \mathbf{S}_n(u',u)$. The statistic F captures a notion of distance between two block samples. We focus on correlation statistics, defined in the next subsection. In time block units the random change time is $\frac{\lambda_u}{T}$, which is a geometric random variable with parameter dT.

Intuitively, our fault detection model posits that the score $\mathbf{S}_n(u,u')$ undergoes a change in distribution whenever either u or u' fails, i.e., at time $\min(\lambda_u, \lambda'_u)$. This model captures the notion that in a networked setting, failed sensor data cannot be used to detect faults in other sensors. Thus our model departs from the traditional single change point detection models [11], in that we are dealing with multiple *dependent* change points based on measurements from a collection of sensors. The standard theory for a single change point can no longer be applied in a straightforward manner.

We formally specify our change point model. Given a score function $\mathbf{S}_n(u,u')$ for each pair of neighbors (u,u'), it is assumed that $\mathbf{S}_n$ for different pairs of sensors are independent. Also given are distributions $f_0(\cdot|u,u')$ and $f_1(\cdot|u,u')$ such that

$$\mathbf{S}_n(u,u') \overset{i.i.d}{\sim} f_0(\cdot|u,u'), \quad n < \frac{1}{T}\min(\lambda_u, \lambda_{u'}),$$
$$\overset{i.i.d.}{\sim} f_1(\cdot|u,u'), \quad n \geq \frac{1}{T}\min(\lambda_u, \lambda_{u'}).$$

We require f_0 and f_1 to be different, that is the Kullback-Liebler divergence between the two densities $D(f_1\|f_0) > 0$,

$$D(f\|g) = \int f(x)\log\frac{f(x)}{g(x)}dx. \quad (3)$$

3.4. Correlation scores

Our choice of correlation score function is motivated by the observation that in many applications when a sensor fails the the correlation experiences an abrupt change (e.g. [10]). The choice of correlation statistics is also attractive because it can be used in non-stationary environments if the time scale is appropriately chosen. Without losing generality assume $p = 1$ so that $X_t(u)$ is a scalar and $\mathbf{X}_n(u)$ is a vector of size T.

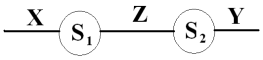

Figure 2: Focusing on two sensors .

The score is defined as

$$\mu_n(u) = \frac{1}{T}\sum_{t\in T_n} X_t(u), \quad (4)$$

$$s_n(u,u') = \frac{1}{T}\sum_{t\in T_n}(X_t(u) - \mu_n(u))(X_t(u') - \mu_n(u')),$$

$$\mathbf{S}_n(u,u') = \phi\left(\frac{s_n(u,u')}{\sqrt{s_n(u,u)s_n(u',u')}}\right),$$

$$T_n = [(n-1)T + 1, nT].$$

The actual score is a transformation of the empirical correlation estimate. The trivial choice is $\phi(x) = x$. To obtain desired statistical behavior, it is sometimes better to choose a combined Fisher and Box-Cox type transformation,

$$\phi_f(x,\gamma) = \frac{1}{2}\log\left(\frac{1+x^\gamma}{1-x^\gamma}\right). \quad (5)$$

We assume that the scores scaled by $\sqrt{T}$ converge to a normal distribution, and that the scores are pairwise independent. This assumption is not required and more complex covariance structures inferred from the data could be used. But our choice works well in practice, and simplifies exposition. Thus

$$\mathbf{S}_n(u,u') \sim N(\mu(u,u'), T^{-1}\,\sigma^2_{u,u'}), \quad n < \frac{1}{T}\min(\lambda_u, \lambda'_u),$$
$$\sim N(0, T^{-1}\,\sigma^2), \quad n \geq \frac{1}{T}\min(\lambda_u, \lambda'_u), \quad (6)$$

Before the change time, each computed score (in our case covariances) is approximately normal. The mean and variance parameters depend on the pairs of sensors. The variance scales as $1/T$ with respect to the window size T. Above we assumed mean and variance are time invariant, but this is not necessary. The assumption can be justified with a simple model. Suppose the blocks $\mathbf{X}_n(u)$ and $\mathbf{X}_n(u')$ are jointly Gaussian random variables, and the Fisher-Box transformation (Equation 5) with $\gamma = 1$ is used; it can then be shown [12] that asymptotic normality holds and

$$\sigma^2_{u,u'} = \begin{cases} \frac{(1-\mu_{u,u'})^2}{T}, & \text{for } \phi(x) = x \\ \frac{1}{T}, & \text{for } \phi(x) = \phi_f(x,1) \end{cases}.$$

The link information measure for (u,u') is [12]:

$$q_1(u,u') = D(f_1\|f_0) \quad (7)$$
$$= T\,\frac{\mu(u,u')^2}{2\sigma^2_{u,u'}} + \frac{1}{2}\left[\frac{\sigma^2}{\sigma^2_{u,u'}} + \log\left(\frac{\sigma^2_{u,u'}}{\sigma^2}\right) - 1\right].$$

The link information measure is minimized when $\sigma^2_{u,u'} = \sigma^2$.

4. Multiple Sensor Online Detection

To simplify the analysis of the solution proposed in this paper, let us first consider the two-sensor case before proceeding to the multiple sensor setting. In the two-sensor scenario

illustrated in Figure 2, the shared link score between the two sensors is Z, all the other links (if any) of sensor 1 are aggregated into a random variable X, and all other links of sensor 2 are aggregated into Y.

Let $\bar{\nu}_1$ be the decision rule for sensor 1 and $\bar{\nu}_2$ the rule for sensor 2. The distributed computation constraint requires $\bar{\nu}_1$ to depend only on X and Z, expressed as $\bar{\nu}_1 \in \mathcal{F}^n_{X,Z}$; similarly, $\bar{\nu}_2 \in \mathcal{F}^n_{Y,Z}$. Furthermore, denote the information distance for X, $q_1(X) = D(f_1(X)\|f_0(X))$, where $f_0(X)$ is the density before change, and $f_1(X)$ the density after change. Similarly define $q_1(Z)$ and $q_1(Y)$. All proofs in this section can be found in the Technical Report [17].

4.1. Background

Consider the single change point detection problem, which can be cast in our framework as a single sensor network with a self-loop graph. Shiryaev [18] showed that a threshold rule on the posterior probability is the optimal choice of stopping time to minimize the weighted sum of the expected delay and the probability of false alarm. The Shiryaev statistic and stopping time are

$$\Lambda_n(X) = \frac{\mathbb{P}^\pi(\lambda \le n|\mathcal{F}^n_X)}{\mathbb{P}^\pi(\lambda > n|\mathcal{F}^n_X)}, \tag{8}$$

$$\nu_S(X) = \inf\{n : \Lambda_n \ge B\}. \tag{9}$$

[19] showed that the Shiryaev rule with threshold $B_\alpha = \frac{1-\alpha}{\alpha}$, with α the false alarm probability bound, achieves the *optimal asymptotic delay* for the problem of minimizing the expected delay constrained to a given false alarm probability. The asymptotic mth moment of delay for the procedure is

$$\lim_{\alpha\to 0} D^\pi_m(\nu_S(X)) \doteq \left[\frac{|\log(\alpha)|}{q_1(X)+d}\right]^m \tag{10}$$

The single change point problem is considerably simpler than the multiple change problem, since once a change is detected, it is attributed to a unique fault, and there is no chance of *confusion* with other potentially failed sensors.

4.2. Detection without information exchange

The natural generalization of the Shiryaev rule for the multiple change point model is to use a threshold rule on the posterior probability of change for each sensor. In a decentralized setting, sensor 1 should use the posterior probability of random variable λ_1, conditional on the observed values of X and Z. Similarly, sensor 2 should use the posterior probability of random variable λ_2, conditional on observed values of Y and Z. This leads to the tests

$$\begin{aligned}
\Lambda_n(X,Z) &= \frac{\mathbb{P}_\pi(\lambda_1 \le n|\mathcal{F}^n_{X,Z})}{\mathbb{P}_\pi(\lambda_1 > n|\mathcal{F}^n_{X,Z})},\\
\Lambda_n(Y,Z) &= \frac{\mathbb{P}_\pi(\lambda_2 \le n|\mathcal{F}^n_{Y,Z})}{\mathbb{P}_\pi(\lambda_2 > n|\mathcal{F}^n_{Y,Z})},\\
\nu_1 &= \inf\{n : \Lambda_n(X,Z) \ge B_\alpha\},\\
\nu_2 &= \inf\{n : \Lambda_n(Y,Z) \ge B_\alpha\}.
\end{aligned} \tag{11}$$

Unfortunately this turns out not to be a good choice, as we can show that asymptotic delays are independent of the statistics of the random variable Z.

Theorem 4.1: The asymptotic delay of the stopping time rules based on posterior probabilities (Equation (11)) are

$$D^\pi_m(\nu_1) \doteq \left[\frac{|\log(\alpha)|}{q_1(X)+d}\right]^m,\ D^\pi_m(\nu_2) \doteq \left[\frac{|\log(\alpha)|}{q_1(Y)+d}\right]^m.$$

Thus in this extension, the common link information is not useful in determining which sensor has failed. The reason is that the information in either link pair (X,Z) or (Y,Z) by itself is not helpful in determining whether the change in Z is induced by a failure in sensor 1 or in sensor 2.

4.3. Detection with information exchange

We propose a new distributed procedure that benefits from the information contained in the shared link. Our procedure requires the definition of two stopping times for each sensor. Define:

$$\begin{aligned}
\nu_1 &= \inf\{n : \Lambda_n(X,Z) \ge B_\alpha\},\\
\nu_2 &= \inf\{n : \Lambda_n(Y,Z) \ge B_\alpha\},\\
\tilde{\nu}_1 &= \inf\{n : \Lambda_n(X) \ge B_\alpha\},\\
\tilde{\nu}_2 &= \inf\{n : \Lambda_n(Y) \ge B_\alpha\},
\end{aligned} \tag{12}$$

where $\Lambda_n(X,Z)$ is the Shiryaev statistic constructed under the assumption that only λ_1 can be finite (only sensor 1 may fail), and $\Lambda_n(Y,Z)$ is the Shiryaev statistic constructed under the assumption that only sensor 2 may fail. The statistics are given by

$$\Lambda_n(X,Z) = \frac{\sum_{k=0}^{n}\pi(k)\prod_{r=1}^{k}f_0(X_r)f_0(Z_r)\prod_{r=k+1}^{n}f_1(X_r)f_1(Z_r)}{\sum_{k=n+1}^{\infty}\pi(k)\prod_{r=1}^{n}f_0(X_r)f_0(Z_r)},$$

$$\Lambda_n(Y,Z) = \frac{\sum_{k=0}^{n}\pi(k)\prod_{r=1}^{k}f_0(Y_r)f_0(Z_r)\prod_{r=k+1}^{n}f_1(Y_r)f_1(Z_r)}{\sum_{k=n+1}^{\infty}\pi(k)\prod_{r=1}^{n}f_0(Y_r)f_0(Z_r)},$$

$$\Lambda_n(X) = \frac{\sum_{k=0}^{n}\pi(k)\prod_{r=1}^{k}f_0(X_r)\prod_{r=k+1}^{n}f_1(X_r)}{\sum_{k=n+1}^{\infty}\pi(k)\prod_{r=1}^{n}f_0(X_r)},$$

$$\Lambda_n(Y) = \frac{\sum_{k=0}^{n}\pi(k)\prod_{r=1}^{k}f_0(Y_r)\prod_{r=k+1}^{n}f_1(Y_r)}{\sum_{k=n+1}^{\infty}\pi(k)\prod_{r=1}^{n}f_0(Y_r)}. \tag{13}$$

We now define the stopping rules for the two sensors:

$$\begin{aligned}
\bar{\nu}_1 &= \nu_1\mathbb{I}(\nu_1 \le \nu_2) + \max(\tilde{\nu}_1, \nu_2)\,\mathbb{I}(\nu_1 > \nu_2),\\
\bar{\nu}_2 &= \nu_2\mathbb{I}(\nu_2 \le \nu_1) + \max(\tilde{\nu}_2, \nu_1)\,\mathbb{I}(\nu_2 > \nu_1).
\end{aligned} \tag{14}$$

The procedure works in an intuitive manner: Each sensor computes posteriors as if the other sensor is always working, until the time one of them declares itself as failed. Notice that both sensors at this point are using the information in the shared link. When one sensor is thought to have failed (e.g. $\nu_1 > \nu_2$) the other sensor stops using the shared link information, and recomputes the change point test using only the information of its own 'private' link. The $\max$ operator reflects the situation that information for one's own private link also dictates that its sensor hsa failed (e.g., $\tilde{\nu}_1 < \nu_2$), in which case one should stop immediately at the present time (ν_2).

Implementation of the procedure requires an extra single bit of information that is issued to neighbors when a sensor declares itself as failed. If this bit is received the neighboring sensors stop using the shared link with the failed sensor, and use a rule based on the remaining links.

The procedure as described requires each sensor to keep track of all the link variables, since when the shared link is dropped, the sensor has to recompute the score using only the remaining links. In the two-sensor case this is not an issue since all stopping times can be computed simultaneously. In a network setting this matters, since we have multiple possible link combinations. But we propose very efficient solutions for this in section 5.

4.4. Performance Analysis

The detection with information exchange algorithm is interesting if we are able to show that for a given false alarm rate $O(\alpha)$, it achieves expected delays smaller than if the common link information is not used.

First, we compute the PFA for the algorithm, focusing on sensor 1 at time n. We can break up the false alarm cases into two distinct situations: when neither change point has occurred by time n ($\lambda_1 > n$ and $\lambda_2 > n$) and when sensor 2 has already failed ($\lambda_2 \leq n$). In the first case, a false alarm happens in the same way it happens in a problem with a single change point, thus the probability is $O(\alpha)$ for the chosen threshold. The second situation is unique to our problem: there is a chance that sensor 1 is *confused* by link Z, behaving as if it is failed, when in reality it is working. We need to show that this *confusion probability* is small. We formally define this quantity.

Definition 3: The confusion probabilities of a set of procedures $(\bar{\nu}_1, \bar{\nu}_2)$ are

$$\xi^{\alpha}_{\lambda_1,\lambda_2}(\bar{\nu}_1) = \mathbb{P}_{\lambda_1,\lambda_2}(\bar{\nu}_1 \leq \bar{\nu}_2, \lambda_2 \leq \bar{\nu}_1 \leq \lambda_1) \quad (15)$$

$$\xi^{\alpha}_{\lambda_1,\lambda_2}(\bar{\nu}_2) = \mathbb{P}_{\lambda_1,\lambda_2}(\bar{\nu}_2 \leq \bar{\nu}_1, \lambda_1 \leq \bar{\nu}_2 \leq \lambda_2) \quad (16)$$

A fault detection procedure is **regular** if

$$\lim_{\alpha\to 0} \xi^{\alpha}_{\lambda_1,\lambda_2}(\bar{\nu}_1) = 0,$$
$$\lim_{\alpha\to 0} \xi^{\alpha}_{\lambda_1,\lambda_2}(\bar{\nu}_2) = 0.$$

We see the importance of regularity in the next theorem.

Theorem 4.2: The PFA of sensors 1 and 2 for the joint procedure with information exchange are bounded as

$$\text{PFA}^{\pi_1,\pi_2}(\bar{\nu}_1) \leq 3\alpha + \xi^{\alpha}_{\lambda_1,\lambda_2}(\bar{\nu}_1),$$
$$\text{PFA}^{\pi_1,\pi_2}(\bar{\nu}_2) \leq 3\alpha + \xi^{\alpha}_{\lambda_1,\lambda_2}(\bar{\nu}_2). \quad (17)$$

If a procedure is not regular we are unable to achieve arbitrarily low false alarm rates. But our procedure is regular.

Theorem 4.3: The procedure of Equation 14 is regular:

$$\lim_{\alpha\to 0} \xi^{\alpha}_{\lambda_1,\lambda_2}(\bar{\nu}_1) = 0,$$
$$\lim_{\alpha\to 0} \xi^{\alpha}_{\lambda_1,\lambda_2}(\bar{\nu}_2) = 0.$$

In section 8 we estimate numerically the confusion probability under a variety of settings and show that it is negligible as long as the variance of X and Y is small compared to that of Z. Given that we can achieve arbitrarily small false alarm rates, what can be said about the detection delay?

Theorem 4.4: The delays of the regular procedures $\bar{\nu}_1$ and $\bar{\nu}_2$ are

$$D^{\pi}_m(\bar{\nu}_1) \doteq D^{\pi}_m(\nu_1)(1-\delta_\alpha) + D^{\pi}_m(\tilde{\nu}_1)\,\delta_\alpha,$$
$$D^{\pi}_m(\bar{\nu}_2) \doteq D^{\pi}_m(\nu_2)\,\delta_\alpha + D^{\pi}_m(\tilde{\nu}_2)(1-\delta_\alpha),$$

as $\alpha \to 0$. Here

$$D^{\pi}_m(\nu_1) = \left[\frac{|\log\alpha|}{q_1(X)+q_1(Z)+d}\right]^m,$$
$$D^{\pi}_m(\tilde{\nu}_1) = \left[\frac{|\log\alpha|}{q_1(X)+d}\right]^m,$$
$$D^{\pi}_m(\nu_2) = \left[\frac{|\log\alpha|}{q_1(Y)+q_1(Z)+d}\right]^m,$$
$$D^{\pi}_m(\tilde{\nu}_2) = \left[\frac{|\log\alpha|}{q_1(Y)+d}\right]^m,$$
$$\delta_\alpha = \mathbb{P}_{\lambda_1,\lambda_2}(\nu_1 > \nu_2).$$

Notice that the asymptotic moments of the delay are a weighted combination (with weight $\delta_\alpha \in [0,1]$) of the optimal delays obtained in the scenario when only a single change point exists. Since $q_1(Z) > 0$, our procedure is always better than a procedure that never uses the shared link: $D^{\pi}_m(\bar{\nu}_1) \leq D^{\pi}_m(\tilde{\nu}_1)$ and $D^{\pi}_m(\bar{\nu}_2) \leq D^{\pi}_m(\tilde{\nu}_2)$. To our knowledge this is the first proposed procedure with provable guarantees.

4.5. General Networks

The shared information algorithm for the two-sensor network can be suitably modified for a general network. Table I shows the proposed procedure, following the same principle as the two-sensor case. In this algorithm, whenever a sensor declares itself failed, all its neighbors recompute their test statistic excluding links with the failed sensor. Section 5 discusses implementation details, including finite storage, and transmission efficient computation.

The analysis in Section 4.4 applies to the general network if the probability of sensors failing simultaneously is small, which will be the case if the fault rates are very small compared to the number of neighboring sensors. The analysis even with this simplification is quite involved, but a key quantity emerges—the confusion probability. If the confusion probability is small, the probability of false alarm is small.

The asymptotic delays depend crucially on the parameter δ_α. In this subsection we explore this further, for the case of independent identically distributed link distributions in a fully connected network.

Networked Sensor Fault Detection: Each sensor $u \in V$ initializes its current neighbors set with all neighboring sensors in the fault graph (including self loops), so $\mathcal{N}_W(u) = \mathcal{N}(u)$. Then each sensor updates its current estimate of its own change point test statistic at time n:

(a) *Data Dissemination:* Each sensor broadcasts its current block of T samples $\mathbf{X}_n(u)$ to sensors u' that are active neighbors in the fault graph (i.e. $u' \in \mathcal{N}_W(u)$). Transmitted block might be transformed or compressed (see Section 5).

(b) *Score Computation:* After collecting all data blocks, the sensor computes the current score for shared links according to some transformation F, for example the correlation (Equation 4):

$$S_n(u,u') = F(\mathbf{X}_n(u), \mathbf{X}_n(u')), \qquad u' \in \mathcal{N}_W(u). \quad (18)$$

(c) *Update Test Statistic:* Recursive update of test statistic using active links(Section 5):

$$O_n(u) = \sum_{u' \in \mathcal{N}_W(u)} \left\{ \frac{(S_n(u,u'))^2}{2\sigma^2} + \right.$$
$$\left. - \frac{(S_n(u,u') - \mu_{uu'})^2}{2\sigma^2_{uu'}} + \log\left(\frac{\sigma^2}{\sigma^2_{uu'}}\right) \right\}$$
$$\log(\Lambda_n(u)) = \log\left(\frac{\Lambda_{n-1}(u)}{1-\rho} + \frac{\rho}{1-\rho}\right) + O_n(u), \quad (19)$$
$$\Lambda_0(u) = \pi_0/(1-\pi_0), \;\; \rho = 1 - e^{-dT}$$

(d) *Fault check and inform:* If

$$\Lambda_n(u) \geq \frac{1-\alpha}{\alpha}, \quad (20)$$

sensor u is declared faulty, and broadcasts failed bit $\delta(u)$ to all sensors $u' \in \mathcal{N}_W(u)$.

(e) *Update Current Links*: For each $u' \in \mathcal{N}_W(u)$, if bit $\delta(u')$ is received:

$$\mathcal{N}_W(u) = \mathcal{N}_W(u) - u', \quad (21)$$

Recompute $\Lambda_n(u)$ with new $\mathcal{N}_W(u)$, using stored samples.

If $\mathcal{N}_W(u)$ is empty (no self loops in fault graph), then stop sensor u.

TABLE I. Description of the networked fault detection algorithm. In a centralized data collection model, the data dissemination stage has no cost.

In the two-sensor case, if X and Y have the same probability density, it is clear from symmetry that $\delta_\alpha = 1/2$. Focusing on sensor 1, we see that the delay in this case is

$$D_m^\pi(\tilde{\nu}_1) \;\; \doteq \;\; \frac{1}{2} D_m^\pi(\nu_1) + \frac{1}{2} D_m^\pi(\tilde{\nu}_1).$$

Furthermore, it is known that if we have $\lambda_2 = \infty$ fixed (sensor 2 never fails), then any detection procedure ν has a delay that satisfies [19]

$$D_m^\pi(\nu) \geq \left[\frac{|\log\alpha|}{q_1(X) + q_1(Z) + d}\right]^m = D_m^\pi(\nu_1).$$

In the case when $\lambda_2 = 0$ fixed (sensor 2 is always failed), link Z gives no information about the status of sensor 1, so any procedure for detecting a fault in sensor 1 satisfies

$$D_m^\pi(\nu) \geq \left[\frac{|\log\alpha|}{q_1(X) + d}\right]^m = D_m^\pi(\tilde{\nu}_1).$$

For any procedure

$$D_m^\pi(\nu) = D_m^\pi(\nu|\lambda_1 < \lambda_2)\mathbb{P}(\lambda_1 < \lambda_2) +$$
$$+ D_m^\pi(\nu|\lambda_1 \geq \lambda_2)\mathbb{P}(\lambda_1 \geq \lambda_2).$$

Since the priors are identical, $\mathbb{P}(\lambda_1 \geq \lambda_2) = 1/2$. The statistics of ν conditional on $\lambda_1 < \lambda_2$ are the same as when we set $\lambda_2 = \infty$. This result, shown in [17], can be understood intuitively since Z indicates the failure of sensor 1 in this case. So $D_m^\pi(\nu|\lambda_1 < \lambda_2) \geq D_m^\pi(\nu_1)$. Intuitively, when $\lambda_1 \geq \lambda_2$ link Z gives no information on the change point for sensor 1, so any procedure should only use link X in the limit of small false alarm probability. Heuristically we reason that $D_m^\pi(\nu|\lambda_1 \geq \lambda_2) \geq D_m^\pi(\tilde{\nu}_1)$. Putting it all together gives

$$D_m^\pi(\nu) \geq \frac{1}{2} D_m^\pi(\nu_1) + \frac{1}{2} D_m^\pi(\tilde{\nu}_1).$$

Thus, in a sense the proposed procedure achieves optimality, if the confusion probability is of $O(\alpha)$.

Consider now a fully connected network, with all links having i.i.d. link distributions before and after change. Denote the performance metric by q_1. Notice that everything is symmetric in this case. Each sensor has an equal chance of being the $(n-k)$th sensor to fail. If we take small false alarm probability ($\alpha \to 0$) and all pairwise confusion probabilities go to zero with the false alarm probability going to zero, it is clear that no false alarm occurs. In the limit, the kth sensor uses either $(k-1)$ sensors to make its decision (if there are no self loops in the graph) or k (if there are self loops). The delay is

$$D_m^\pi(\nu_k) \doteq \left[\frac{|\log(\alpha)|}{(k-1+\delta_s)q_1 + d}\right]^m, \quad (22)$$

where $\delta_s = 1$ if the fault graph has self loops. Since each sensor has an equal chance of failing as the k-th sensor, the average delay for each sensor is

$$D_m^\pi(\nu) \doteq \frac{1}{|V|}\sum_{k=1}^{|V|}\left[\frac{|\log(\alpha)|}{(k-1+\delta_s)q_1 + d}\right]^m. \quad (23)$$

5. Algorithm Implementation

We investigate several practical considerations in the implementation of the proposed detection algorithm.

5.1. Correlation Computation: Compression and Synchronization

Given blocks $\mathbf{X}_n(u)$ and $\mathbf{X}_n(u')$ from sensors u and u', direct correlation as in Equation 4 might not be the best choice, either because the clocks of the two sensors may be delayed relative to each other, or more importantly, there could be a propagation delay in the underlying physical environment that reduces the effective correlation score between both sensors. A simple solution to improve performance and overcome these difficulties is to use cross correlation instead of correlation [15]. Denote by $\mathbf{X}_n^k(u)$ the block of samples $X_t(u)$ for $t \in [(n-1)T+k, nT+k]$, that is the samples delayed by k units.

The maximum cross correlation can be used to 'synchronize' the samples:

$$[k^{opt}, l^{opt}] = \arg\max_{k,l \in [0,M], k \leq l} \frac{1}{P}\sum_{n \in [1,P]} F(X_n^k(u), X_n^l(u')).$$

Here M is the maximum allowed shift between the sensor samples, P is the number of blocks to evaluate the shift, and

F is the correlation score definition in Equation 4. The shift is adjusted so that the correlation between samples is maximized either once at initialization or periodically depending on the clock skew between the nodes. Once the shift is adjusted, correlations are computed with respect to the chosen shifts.

If the block size T is large enough, an alternative procedure, which saves energy by reducing the amount of data transfer, is to use a Discrete Cosine Transform (DCT) to evaluate the maximum cross correlation. The method relies on computing the DCT of each block $\mathbf{X}_n(u)$ appropriately zero-padded and using these coefficients to compute the maximal correlations with a simple scalar product. Additional savings can be obtained by using only a few coefficients of the DCT. Details of such a strategy can be found in [15]. If the underlying signal has a few dominant frequencies this method is very efficient. Alternative transforms such as wavelets could be used. In fact, this is the suggested approach even when synchronization is not required.

5.2. Quantization

Considerable savings can be obtained if the block vectors $\mathbf{X}_n(u)$ are quantized to some finite precision before the correlation is performed. Since we are working in a stochastic framework, dithered quantization is favored. A stylized version of quantizing a real number x in dithered quantization is to output $y = Q_b(x + \epsilon)$, where ϵ is a uniform random variable and Q_b is a function that outputs a b-bit quantized version of the input.

Denote by $\mathbf{S}_n^b(u, u')$ the correlation score computed from the quantized samples of block $\mathbf{X}_n(u)$. The following lemma gives the asymptotic behavior of the estimates, when the expected value of the score without quantization is $\mu_{u,u'}$.

Lemma 1: Let us assume that the quantizer is $B + 1$-bit with full scale $X_{\max}$ such that the quantization error is uniformly distributed in interval $[-\frac{X_{\max}}{2^b}, \frac{X_{\max}}{2^b}]$ and statistically independent of the system input. (This assumption is valid for subtractive dither quantization when the dither satisfies certain conditions, e.g. i.i.d uniform dither [15]). As $T \to \infty$,

$$\sqrt{T}(\mathbf{S}_n^b(u, u') - \mu_{u,u'}) \xrightarrow{d} N(0, T\,\bar{\sigma}^2_{u,u'})$$
$$\bar{\sigma}^2_{u,u'} = \frac{1}{T}\left(\sigma^2_{u',u} + 2\,X^2_{\max}\,\sigma_b^2 + \sigma_b^4\right);\ \sigma_b^2 = \frac{1}{12 \cdot 2^{2b}}$$

Proof: Once we replace x_i^b and y_i^b by $x_i + \epsilon^b_{x,i}$ and $y_i + \epsilon^b_{y,i}$ respectively, where $\epsilon^b_{x,i}$ and $\epsilon^b_{y,i}$ are the quantization errors for x_i^b and y_i^b respectively, x_i, $\epsilon^b_{x,i}$, y_i and $\epsilon^b_{y,i}$ are all independent of each other, and the result follows.

Quantization increases the variance of a Gaussian distribution by additional terms that are inversely proportional to 2^{2b}, so $b = O(-\log(\sigma_{u',u}/X_{\max}))$ gives a performance that is about the same with or without quantization.

5.3. Windowed iteration

Computational efficiency is important in practical applications. The information sharing procedure proposed in Section 4.3 relies on computing the Shiryaev statistic for each sensor (Equation 13). The statistic can be recursively computed as:

$$\begin{aligned}\log(\Lambda_n) &= \log\left(\frac{\Pi_{n-1}}{\Pi_n}\Lambda_{n-1} + \frac{\pi_n}{\Pi_n}\right) + \log\left(\frac{f_1(S_n)}{f_0(S_n)}\right), \\ &= \log\left(\frac{\Lambda_{n-1}}{1-\rho} + \frac{\rho}{1-\rho}\right) + \log\left(\frac{f_1(X_n)}{f_0(X_n)}\right), \quad (24)\end{aligned}$$

where $\rho = \frac{1}{dT}$, and for correlation computation

$$\log\left(\frac{f_1(X_n)}{f_0(X_n)}\right) = \log\left(\frac{\sigma^2}{\sigma^2_{uu'}}\right) + \frac{S_n^2}{2\sigma^2} - \frac{(S_n - \mu_{uu'})^2}{2\sigma^2_{uu'}}. \quad (25)$$

The $\log$ function is used for convenience and to increase numerical precision.

The procedure in Section 4.3 requires each sensor to keep a history of all observed link score samples, since whenever a sensor detects a failure, others sensors that share links with the failed sensor have to recompute the test statistic without the shared link score. There is a practical implementation of the algorithm that avoids this. Before a failure occurs, the test statistic is ideally expected to be zero. After the failure, the proposed procedure requires about $D_m^\pi(\nu)$ samples to detect a fault, so a procedure that remembers a constant multiple of this number of samples works well. Notice that as sensors fail sequentially we have to increase the number of stored samples. Denoting by $\mathcal{N}_W(u)$ the set of working neighbors at time n, the sample storage size $M_n(u)$ required at time n for u is

$$\begin{aligned}\tilde{q}_{1,n}(u) &= \max_{u' \in \mathcal{N}_W(u)} q_1(u, u'), \\ M_n(u) &= T\,\frac{C\,\log(\alpha)}{\sum_{u' \in \mathcal{N}_W(u)} q_1(u, u') - \tilde{q}_1(u) + d\,T}, \quad (26)\end{aligned}$$

in which C is a constant factor (a good choice is $C = 1.5$) and T is the window size. The memory estimate subtracts the most informative link at each stage since we don't know which sensor might fail requiring recomputation, and we always assume the most useful sensor (in terms of decreasing delay) might. Each time a sensor reports a failure, sensors that share fault links all recompute the Shiryaev statistic using the stored samples.

6. Time Scale Selection

We address the choice of time scale or block size T. We first show how performance for different T values can be compared. We then discuss how to choose T. Lastly, we show a practical problem using PeMS data.

6.1. Delay scaling

The fault model of Equation 6 might suggest that we could reduce detection delay arbitrarily, since by increasing T we can make the variance arbitrarily small. But to legitimately compare the mth moment of the delay for different T, we should consider the total number of samples rather than the

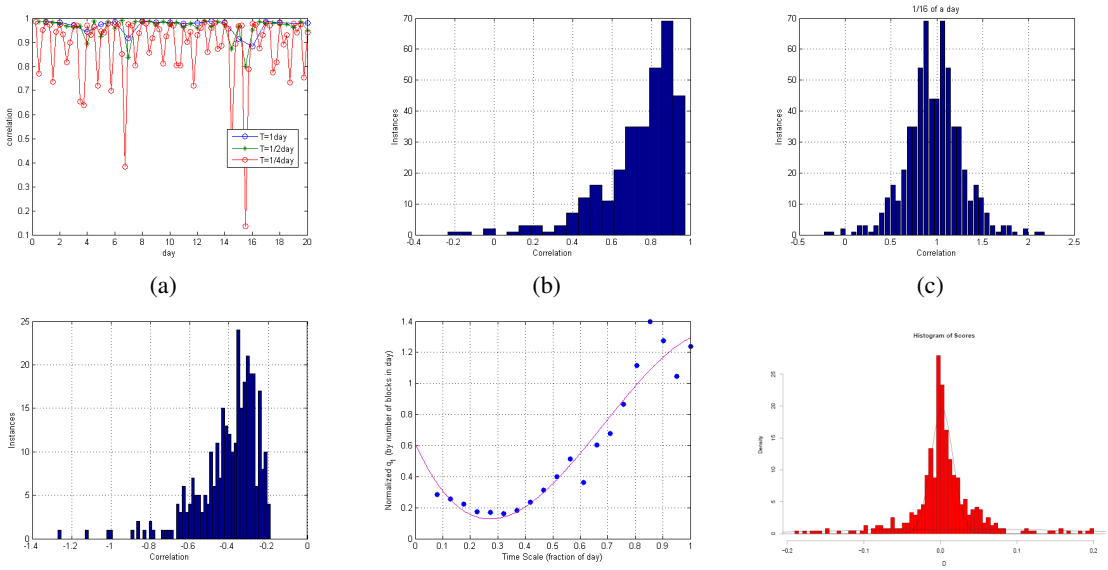

Figure 3. (a) Daily correlation values for different time scales, (b) Correlation distribution for $1/16$ of total daily samples, (c) Symmetrized version of (b), (d) Fisher transform with $\gamma = 1$, (e) Information parameter q_1 normalized by T and (f) Correlation distribution for broken sensors from [10].

number of blocks,

$$D_m^{\pi,T}(\nu_u) = T^m \times D_m^{\pi}(\nu_u),$$
$$= \left[T\,\frac{\log(\alpha)}{q_1 + d\,T}\right]^m = \left[\frac{\log(\alpha)}{\frac{q_1}{T} + d}\right]^m.$$

Here q_1 is a sum or average of the individual link quality metric, which by Equation 7 is given by

$$\frac{q_1(u,u')}{T} = \frac{\mu(u,u')^2}{2\sigma_{u,u'}^2} + \frac{1}{2\,T}\left[\frac{\sigma^2}{\sigma_{u,u'}^2} + \log\left(\frac{\sigma_{u,u'}^2}{\sigma^2}\right) - 1\right].$$

Thus merely by increasing T one cannot reduce the delay arbitrarily: If the variances are equal before and after a fault, the delay (in number of samples) is independent of T; and if the variances are different, there could even be a performance loss as $\frac{q_1(u,u')}{T}$ might decrease with T.

6.2. Events and faults time scale comparison

The choice of the time scale parameter must compare the time scale of faults–duration between successive faults–and the time scale of events–time between signification changes in the environment. In most sensing environments, one expect events to have a much smaller time scale than faults. That is, a change in sensor measurements caused by an event is expected to propagate to neighboring sensors at a speed that depends on the physical environment. On the other hand, sensor faults should not propagate to neighboring sensors and these faults are likely to persist longer.

Sensor failures frequently are intermittent: a sensor fails and after some time it spontaneously recovers. (PeMS sensors suffer from intermittent failures.) In such situations, if a large enough density of sensors is available, the detection delay can be made small enough to detect intermittent failures. In fact, once a sensor is detected as failed, the sequential procedure can continue with some modifications to detect when the measurements are reliable again. So the requirement for detection of intermittent failures is that the average length of time a sensor remains failed is of the same order as the detection delay.

Consider a simple model in which once an event occurs at the location of sensor u, its measurements become uncorrelated with those of its neighbor u'. Suppose events on average last τ samples. This could be either how long the event lasts, or the time to propagate the change caused by an event to neighboring sensors. During the time window τ, u samples an i.i.d. random variable with variance σ_e^2. At other times, the sensors sample i.i.d. values with a correlation of $\rho_{u,u'}$ and a variance of σ_S^2. If $\tau >> T$, we are unable to distinguish the event at sensor u from the sensor's failure. In fact, a simple computation reveals that the expectation of the empirical correlation with a window of size T (assuming an event at u occurs at the beginning of the time block) is

$$\hat{\rho}_{u,u'}(T,\tau) = \left\{\frac{(1-r)_+}{\sqrt{(1-r)_+ + r\,\psi_{e,S}}}\right\}\rho_{u,u'},$$
$$r = \frac{\tau}{T},\ \psi_{e,S} = \frac{\sigma_e^2}{\sigma_S^2}$$

As expected, when T is large relative to τ, the effect of the event is reduced (implying a correlation that is close to the case when the event is not present). Furthermore, if event uncertainties are large with respect to usual behavior uncertainties, a larger time scale helps even more. If event uncertainties are small, expected correlations are smaller, but the events do not significantly affect the system.

6.3. Example

To show how to select the time scale in a real application, we use 5-minute average density data from PeMS for Interstate 210-West in Los Angeles, which has 45 sensing stations, about 2 miles apart. Events such as accidents and demand-induced traffic congestion cause changes in the measured density, and we wish to distinguish the changes due to these events from changes due to sensor failures. We select two neighboring stations. Figure 3(a) shows the correlation over time for different time scales. Notice that for small time scales, we can observe large correlation drops, which correspond to events that have a low propagation speed. The implicit averaging proposed by our algorithm is essential in such situations.

Notice from Figure 3(b) that the correlation with the identity transformation function does not have a gaussian characteristic. The main reason for this is that our data set is limited. We propose two different approaches for handling such situations. Both are simple and fit within the methodology proposed here. The first approach uses a padded density estimate. Figure 3(c) shows the padded histogram for our sample set, in which we can clearly see a bell curve. From this curve we are able to estimate the parameters $\mu = 1$ (by definition) and $\sigma^2 = 0.0928$. But we also know that correlation values never exceed 1 (which is also the mean of our estimated distribution). Thus, we should use as a distribution for the score the distribution conditional on the fact that the score is less than the mean, which can be directly computed as

$$\mathbf{S}_n(u, u') \sim 2\, N(1, T^{-1}\, \sigma^2_{u,u'}), \quad n < \frac{1}{T} \min(\lambda_u, \lambda'_u).$$

After failure we don't see the cutoff effect [10], so the distribution remains as before (Equation 6). Notice that the algorithm is identical, except that the constant factor $(-|\mathcal{N}_W(u)| \log 2)$ should be added to the definition of $O_n(u)$ in Table I. The second approach is to use the Fisher type transformation in Equation 5. Figure 3(d) shows the result for the parameter value $\gamma = 1$. The distribution is more gaussian shaped. Figure 3(e) computes the scaled information metric $\bar{q}_1/T$ for several choices of the time scale parameter T. Observe that if the time is less than half a day, performance is the same. Some gains are observed as we increase the time scale.

7. Energy, delay and density tradeoff

We develop a tradeoff model to evaluate optimal choices of neighborhood size on an energy constrained network. We use delay results from previous sections to evaluate choices faced by a sensor under such constraints in a random placement setting.

7.1. Correlation decay

Many sensor networks monitor spatial and temporal changes in the environment. The correlation between measurements at different locations usually decays with distance. For example, in PeMS, the correlation of traffic measurements by adjacent sensors decays with the distance between them, since there are more points (ramps) where vehicles enter and exit. A simple way to capture this effect is an additive model

$$F(k+1) = F(k) + F_{in}(k+1) - F_{out}(k),$$

where k denotes the kth section of the highway, $F(k)$ denotes the flow in the kth section, $F_{in}(k+1)$ denotes the incoming flow to the kth section through an on-ramp, $F_{out}(k)$ denotes the outgoing flow in the previous section. Assume that the incoming flows are i.i.d. random variables with variance σ^2. If the outgoing flows are proportional to the input flows ($F_{out}(k) = -\beta F(k)$, for $0 < \beta < 1$) we have

$$\rho(k, \tilde{k}) = \frac{\sigma^2}{1-\beta^2} \beta^{|k-\tilde{k}|}.$$

The correlation decays with the distance between sensors, but the decay rates are different. The performance of the proposed fault detection algorithms depends crucially on the expected correlations between the sensors, as well as on the variance of this estimate, through the information parameter $q_1(u_i, u_j)$ of the link between sensors u_i and u_j. Under reasonable conditions, the variance of the correlation estimate increases as the correlation itself decreases. Under our normality assumptions, we showed that $q_1 = \rho^2/\sigma^2_\rho$. If we assume a power law decay with distance and $\sigma^2_\rho = O(1/\rho^{2p})$, we can state that

$$q_1(u_i, u_j) \propto T\, \beta^{\gamma \cdot \text{dist}(u_i, u_j)}, \tag{27}$$

in which the parameter $\gamma \geq 0$ controls the decay rate of the link informativeness as the distance between the sensors $\text{dist}(u_i, u_j)$ increases.

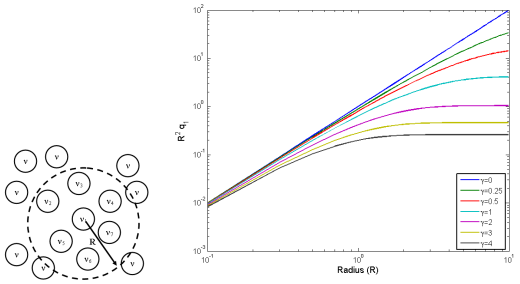

Figure 4. Informativeness models with respect to connectivity radius R .

7.2. Energy consumption

Some sensor networks have limited energy. If most energy is consumed in communication, it is important to minimize the data to be transferred. Suppose the energy consumed in transferring data between u_i and u_j is proportional to the square of the distance between them, $e_C(u_i, u_j) \propto \text{dist}(u_i, u_j)^2$. There might then be a maximum radius R of interest to realize fault detection for a single sensor with a limited power budget.

7.3. Tradeoff analysis

We adopt the viewpoint of a single sensor u_1, whose neighbors are randomly placed following a Poisson process on

a disk with center u_1 and mean (spatial) density η_F sensors per m^2 [8]. Assume these neighbors never fail. We use a mean field approximation to evaluate the tradeoffs between energy, detection delay and density.

The expected link informativeness in a disk with radius R, normalized by time scale, is

$$\begin{aligned}\bar{q}_1 &= \mathbb{E}[q_1(u_1,u_j)/T] \\ &= C\int_0^R \beta^{\gamma\,\mathrm{dist}(u_1,u_j)} d\mu(\mathrm{dist}(u_1,u_j)) \\ &= C\int_0^R \beta^{\gamma\,x}\frac{2}{R^2}x dx \\ &= \begin{cases} C & \gamma = 0 \\ \frac{2C}{(\log(\beta)\gamma R)^2}\left[1+\beta^{\gamma R}(\log(\beta)\gamma R-1)\right] & \gamma > 0 \end{cases}.\end{aligned}$$

For density η_F, the disk has on average $N = \eta_F \pi R^2$ sensors. Using the mean field approximation (valid for large N), the expected sample delay of the detection procedure is

$$\begin{aligned}\mathbb{E}[D_m^\pi(\nu_1)] &= \mathbb{E}\left[\frac{\log\alpha}{\sum_j q_1(u_1,u_j)+d\,T}\right] \approx \mathbb{E}\left[\frac{\log\alpha}{N\bar{q}_1\,T+d\,T}\right] \\ &= \frac{\log\alpha}{\eta_F\,\pi R^2\bar{q}_1\,T+d\,T}.\end{aligned}$$

The expected power consumption for each transmission round to each neighbor is

$$\begin{aligned}\mathbb{E}[e_C(u_1,u_j)] &= K\int_0^R \mathrm{dist}(u_1,u_j)^2\mu(\mathrm{dist}(u_1,u_j)), \\ &= K\int_0^R x^2\frac{2}{R^2}x dx = \frac{1}{2}K\,R^2.\end{aligned}$$

The average number of rounds of communication is $\bar{\lambda} + D_m^\pi(\nu_1)$, where $\bar{\lambda} = e^{d\,T}$ is the average failure time. Putting these together, using the mean field approximation to the delay in the first step, we obtain the total power consumed

$$\begin{aligned}\bar{P} &= \mathbb{E}[e_C(u_1,u_j)(\lambda + D_m^\pi(\nu_1))\,N], \\ &\approx \frac{1}{2}K\,R^2\,[\bar{\lambda}+\mathbb{E}[D_m^\pi(\nu_1)]]\,\eta_F\,\pi R^2.\end{aligned}$$

If $\bar{q}_1$ is small compared to d, the expected delay is dominated by $1/d$, which is smaller than $\bar{\lambda}$. If $\bar{q}_1$ is large, the delay is small. Thus essentially the total average power consumed by sensor u_1 is $O(\rho\,R^4)$. The expected sample delay is of order

$$\mathbb{E}[D_m^{\pi,T}(\nu_1)] = T\,\mathbb{E}[D_m^\pi(\nu_1)] = O\left(\frac{1}{\max\{\eta_F\,R^2\,\bar{q}_1, d\}}\right).$$

There are two ways to improve performance: (1) by increasing R for a fixed density, which corresponds to communicating with neighbors further away, and (2) by increasing the density as a function of R, requiring additional sensors. Which choice is better depends on the parameter γ of the underlying environment. For the model in Equation 27, $R^2\bar{q}_1$ increases with R^2 when $\gamma = 0$, and is order constant when $\gamma > 0$. Thus increasing R for a fixed density does not help reduce the delay arbitrarily when $\gamma > 0$. Figure 4 plots $\bar{q}_1$ as a function of R for the different models. In the order constant situations we need to increase the density as a function of $\eta_F(R) = R^p$ for some $p > 0$, which increases energy consumption from $O(R^4)$ to $O(R^{4+p})$. If performance is measured as total average power per unit detection delay, $\bar{P}/\mathbb{E}[D_m^\pi(\nu_1)] = O(R^2/\bar{q}_1)$, increasing density improves performance.

8. Examples

We evaluate the performance of our algorithm in simulations, which allows us to precisely define the moment of failure. We simulate three different situations: the two-node network and the fully connected network proposed in Section 4, and a toroidal grid network (see [4] for a definition). This is basically a four connected network that wraps around.

As a benchmark, we compute the expected delay of a naive fault detection strategy: direct thresholding of the correlation, assuming that the distributions are known. For a 5-node fully connected network, and a false alarm probability of 0.0001, approximate computations reveal that the expected delay is on the order of 172 blocks. By comparison, our approach yields a delay of 50 blocks for a false alarm probability of 10^{-20} (essentially zero), which it is much more efficient. The main reason is that we perform appropriate implicit averaging.

8.1. Two Sensor Network

We focus initially on the case in Figure 2. All variables are Gaussian. The mean parameters are $\mu_X = \mu_Y = \mu_Z = 1$ before change, and zero after change. Random variables X and Y are i.i.d. with variance σ_S^2. The common link Z has a fixed variance $\sigma_Z^2 = 1$. The prior failure rate is $d = -\log(0.01)$. Figure 5(a) shows a typical correlation sample path when $\sigma_S^2 = 0.2$. Notice that without time averaging it is very hard to say exactly when the change (failure) occurred.

In Section 4 we argued that the confusion probability should go to zero as the false alarm rate $\alpha \to 0$ for the procedure to be consistent, and we see this in Figure 5(b). Notice though that the rate depends on the uncertainty in the non-shared links σ_S^2. From Figure 5(c), if $\sigma_Z^2/\sigma_S^2 < 1.8$, the confusion probability is $O(\alpha^p)$ with $p < 1$, so the total false alarm rate of the procedure (Equation 17) grows slower than α. But for higher ratios, our procedure essentially has false alarm rate α, so it is indeed valuable to have additional sensors in a neighborhood.

Figure 5(d) shows the theoretical and experimental average delays obtained when the threshold is $\alpha = 10^{-7}$. There is disagreement between the curves, although the qualitative behavior is as expected. The disagreement is because our results are for $\alpha \to 0$. This issue is well known in sequential analysis [19]. In the next section we show the high accuracy of the approximation for small values of α. Figure 5(e) compares the behavior of our procedure using the common link Z and one that does not use it at all. There is a substantial reduction in delay using a shared link. Figure 5(f) is the corresponding theoretical prediction. There is a qualitative agreement between theory and simulation experiment.

8.2. General Networks

Now consider a fully connected network of sensors. Figure 6(a) shows the average detection delay for $\alpha = 0.12$ and

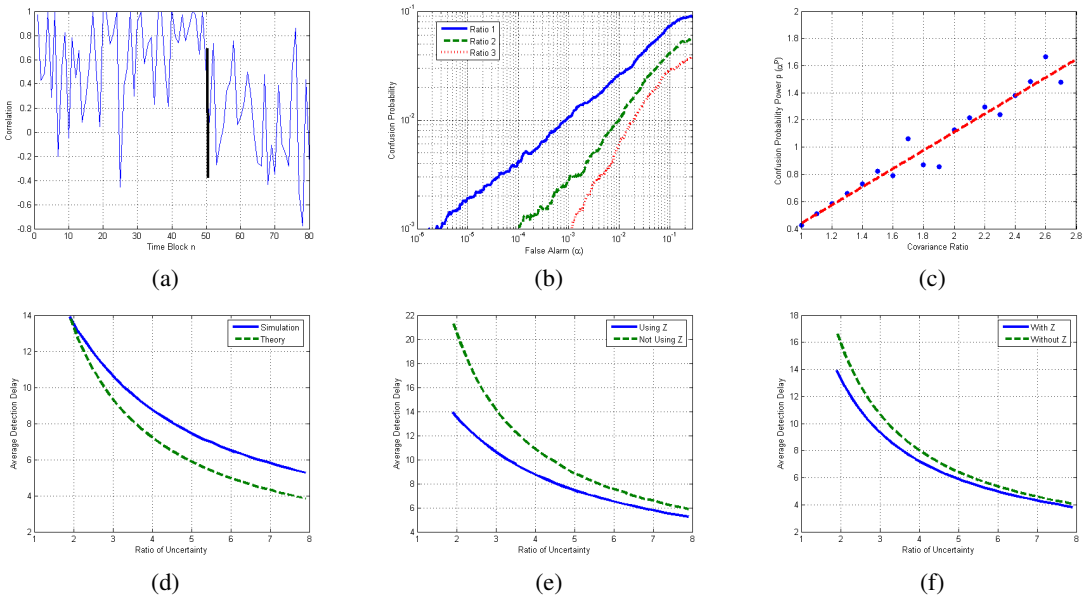

Figure 5. Two Sensor Network: (a)Sample path for correlation with change point at $n = 50$, (b) Confusion probability estimates for different variance ratios and (c) Confusion probability exponent estimates. Covariance ratio in these figures refers to the quantity σ_Z^2/σ_S^2.

Figure 6(c) for $\alpha = 10^{-20}$. As α becomes very small, our theoretical predictions agree better with experiment. Furthermore, the reduction in delay diminishes as the number of sensors increases beyond 20. Figure 6(b) shows the actual PFA observed for selected false alarm targets. As with the two-sensor case (in which the uncertainty ratio played the role of the number of nodes), beyond 10 sensors the false alarm probability is below the target level. Thus the confusion probability rate becomes large at that point. Figure 6(d) shows that with 20 nodes, the observed false alarm is always below the target level.

Lastly, we simulate a toroidal network, in which each sensor has four neighbors. The previous results lead us to believe that the average delay should remain the same independent of the number of sensors in the network, since the connectivity is fixed. Figure 6(e) shows this (except for when we move from 4 nodes–which is fully connected). Compare the delay level to the uncertainty ratio of 5 or a fully connected network with 4 sensors. The results are close. We can see also in Figure 6(f) that since the connectivity is still low, the false alarm is slightly higher than the target.

9. Discussion and Conclusions

In the paper we developed and evaluated an algorithm for distributed online detection of faulty sensors. We proposed a set of basic assumptions and a framework based on the notion of a *fault graph* together with fundamental metrics to evaluate the performance of any sequential fault detection procedure. Then we proceeded to analyze an efficient algorithm that achieves a good performance under the proposed metrics, and even an optimal performance under certain scenarios. As far as we know, this is the first paper to derive bounds on detection delay subject to false alarm constraints in a multiple fault or multiple change point setting. We validated the assumptions behind our algorithms with real data collected from a freeway monitoring application.

Our algorithm performs an implicit averaging which leverages the short term history of the samples reducing the detection delay for a fixed false alarm. Most of the proposed methods in the literature do not perform this averaging, and therefore are subject to much longer delays. Our algorithm and framework are general enough that even model based methods for computing scores, such as the one proposed in [21] or the primitive in [7], can benefit from the proposed procedure. That score method though might not be very efficient if the observed processes are non stationarity such as in freeway monitoring. Compared to procedures such as in [6] and in [16], our method benefits from implicit averaging, whereas those methods make sequential decisions based on only the current observation.

One important feature of the proposed procedure is that weak sources of evidence can be combined to give a reliable detection of failure. As long as the average correlation when a sensor is working is slightly larger then when it has failed detection can be performed reliably. Notice that very large uncertainties are tolerated, although detection delays increase. On the other hand, as more neighboring sensors are added, the shared information can be used to reduce delays. This means that in situations where fault periods are short can still be detected. Some straightforward adaptation of the algorithm also allows for detecting when a malfunctioning sensor might return to give reasonable readings in intermittent detection scenarios.

Although we focused on the case where the distribution of the correlations is approximately Gaussian, in case other score metrics are used, the proposed algorithm can be adapted for

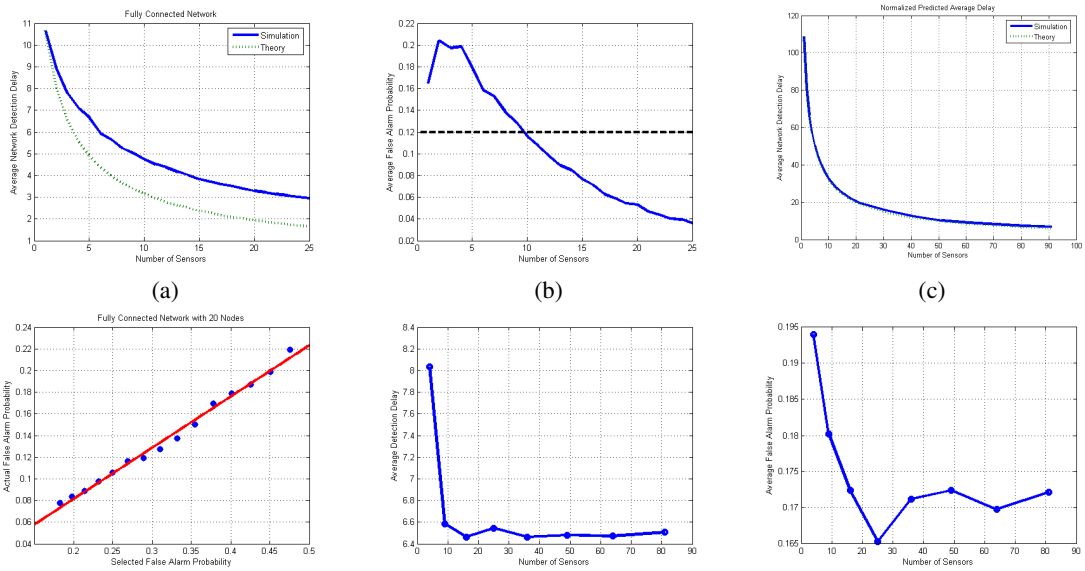

Figure 6. Fully Connected Network: (a)Detection Delay as a function of the number of sensors for $\alpha = 0.12$ and (b)Empirical average false alarm. (c)Detection Delay as a function of the number of sensors for $\alpha = 10^{-20}$ and (d) Selected false alarm rate and actual rate for network with 20 nodes. Grid Network: (e)Average Detection Delay as a function of number of sensors and (f) False alarm rate. Chosen false alarm rate $\alpha = 0.12$.

different statistical distributions. As avenues for future work we propose to investigate the estimation of the fault graph, currently based on geographic proximity, and generalizations of the methodology to applications such as event detection.

References

[1] B. E. Brodsky and B. S. Darkhovsky. *Nonparametric methods in change-point problems*. Kluwer Academic Pub, 1993.

[2] C. Chen, J. Kwon, J. Rice, A. Sakabardonis, and P. Varaiya. Detecting errors and imputing missing data for single loop surveillance systems. *Transportation Research Record*, (1855):160–167.

[3] C. Chong and S. P. Kumar. Sensor networks: Evolution, opportunities, and challenges. *Proceedings of the IEEE*, 91:1247–1256, 2003.

[4] A. G. Dimakis, A. D. Sarwate, and M. J. Wainwright. Geographic gossip: efficient aggregation for sensor networks. In *Information Processing in Sensor Networks (IPSN)*, pages 69–76, 2006.

[5] R. Durrett. *Probability: Theory and Examples*. Duxbury Press, New York, NY, 1995.

[6] E. Elnahrawy and B. Nath. Context-aware sensors. *Lecture Notes in Computer Science (LNCS)*, 2920:77–93, 2004.

[7] S. R. Jefferey, G. Alonso, M. J. Franklin, W. Hong, and J. Widom. A pipelined framework for online cleaning of sensor data streams. In *ICDE*, 2006.

[8] J.G.Proakis. *Digital Communications*. McGraw-Hill, New York,NY, 2000.

[9] F. Koushanfar, M. Potkonjak, and A. Sangiovanni-Vincentelli. Fault-tolerance in sensor networks. *Handbook of Sensor Networks*, 36, I. Mahgoub and M. Ilyas (eds.) 2004.

[10] J. Kwon, P. Bickel, and J. Rice. Web of evidence models: Detecting sensor malfunctions in correlated sensor networks. Technical report, University of California Berkeley, 2003.

[11] T. L. Lai. Sequential analysis: Some classical problems and new challenges (with discussion). *Statist. Sinica*, 11:303–408, 2001.

[12] E. Lehmann. *Elements of Large-Sample Theory*. Springer, 1999.

[13] X. Luo, M. Dong, and Y. Huang. On distributed fault-tolerant detection in wireless sensor networks. *IEEE Transactions on Computers*, 55:58–70, 2006.

[14] K. Marzullo. Tolerating failures of continuous-valued sensors. *ACM Transactions on Computer Systems*, 8:284–304, 1990.

[15] A.V. Oppenheim, R.W.Schafer, and J.R.Buck. *Discrete-time Signal Processing*. Prentice-Hall, Inc., New Jersey, 1999.

[16] E. Ould-Ahmed-Vall, G. F. Riley, and B. Heck. Distributed fault-tolerance for event detection using heterogeneous wireless sensor networks. Technical Report GIT-CERCS-06-09, Georgia Institute of Technology, 2007.

[17] R. Rajagopal, X. Nguyen, S. C. Ergen, and P. Varaiya. Distributed online fault detection with multiple sensors. Technical report, University of California Berkeley, 2007.

[18] A. N. Shirayev. *Optimal Stopping Rules*. Springer-Verlag, 1978.

[19] A.G. Tartakovsky and V.V. Veeravalli. General asymptotic bayesian theory of quickest change detection. *Theory of Probab. Appl.*, 49(3):458–497, 2005.

[20] J. N. Tsitsiklis. Decentralized detection. In *Advances in Statistical Signal Processing*, pages 297–344. JAI Press, 1993.

[21] D. Tulone and S. Madden. An energy-efficient querying framework in sensor networks for detecting node similarities. In *MSWiM*, pages 191–300, 2006.

[22] V. V. Veeravalli. Sequential decision fusion: theory and applications. *Journal of the Franklin Institute*, 336:301–322, 1999.

[23] PeMS website. http://pems.eecs.berkeley.edu.