Sparsity-based Online Missing Data Recovery Using Overcomplete Dictionary

Di Guo, Zicheng Liu, Xiaobo Qu, Lianfen Huang, Yan Yao, Ming-Ting Sun

***Abstract*—Estimating missing sample values is an inherent problem in sensor network applications. In wireless sensor networks, due to power outrage at a sensor node, hardware dysfunction, or bad environmental conditions, not all sensor samples can be successfully gathered at the sink. Additionally, in the context of data streams, some nodes may continually miss samples for a period of time. To address these issues, a sparsity-based online data recovery approach is proposed in this paper. First, we construct an overcomplete dictionary composed of past data frames and traditional fixed transform bases. Assuming the current frame can be sparsely represented using only a few elements of the dictionary, missing samples in each frame can be estimated by Basis Pursuit. If some delay is acceptable, the estimation of the current frame can be further improved by leveraging the observation from the next frames. Our method was tested on data from a real sensor network application: monitoring the temperatures of the disk drive racks at a data center. Simulations show that in terms of estimation accuracy and stability, the proposed approach outperforms existing average-based interpolation methods, and is more robust to burst missing along the time dimension.**

***Index Terms*—Sparsity, data recovery, sensor, dictionary.**

I. Introduction

WIRELESS Sensor Networks (WSN) are characterized by a dense deployment of sensor nodes that continuously observe a physical phenomenon, such as environmental sensing [1], habitat monitoring [2] and other emergency cases [3]. Here, we focus on the many-to-one architecture where distributed sensors collaboratively relay their data to a single sink (base station).

Some transmitted sensor data may be lost or corrupted due to many reasons, such as power outrage at a sensor node, hardware dysfunction, and bad environmental conditions. Many real-time applications, such as traffic and safety control [4], and healthcare [5] need to operate on continuous data streams. In this paper, we consider a 2-D (two dimensional) data stream scenario, and the missing data of each 2-D frame need to be estimated at the sink online with low time delay. Fig. 1 shows a sensor network with missing samples in time intervals n-1, n, and n+1, respectively.

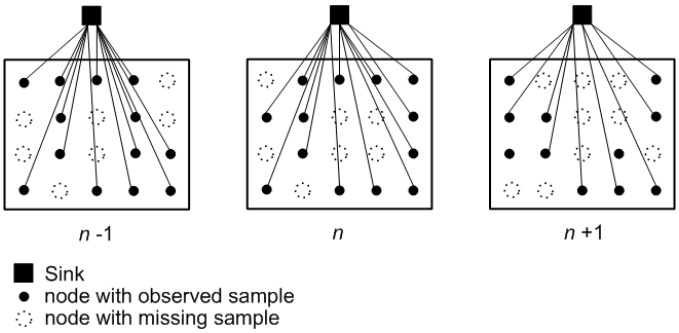


Fig. 1. A sensor network with missing samples.

To obtain the lost data, simply retransmitting these data could cause a long delay, consume power of the node [6], and take away valuable communication bandwidth. Furthermore, when there is a hardware failure at a sensor node, retransmitting is not an option. A more desirable approach is to perform all the necessary data recovery functions at the sink, since the sink usually has much relaxed power and computing constraints than the sensors have.

To estimate the missing data, typically, a model of the characteristics of the sensor data is needed, and the missing data are interpolated using spatial and temporal correlations among the sensor readings [7-9]. Examples of spatial interpolations include Inverse Distance Weighted Averaging (IDWA) [8] and Kriging [9]. However, they only consider data within a single frame, and do not take advantage of information in the sequential data frames. Others take temporal factors into consideration. For instance, the work in [10] is restricted to Markov models, where given the samples at time interval n, the samples at time interval n+1 are independent of those for any time earlier than n. This assumption is too restrictive.

Recently, as a new powerful tool for statistical signal

Manuscript received August 25, 2011. This work was supported in part by National Natural Science Foundation of China under grant 61172097, Tsinghua-Qualcomm Joint Research Program, and by the Fellowship of Postgraduates' Oversea Study Program for Building High-Level Universities from the China Scholarship Council.

D. Guo is with Department of Communication Engineering, Xiamen University, Xiamen 361005 CHINA, and also with Department of Electrical Engineering, University of Washington, Seattle, WA 98195 USA (e-mail: guodi@xmu.edu.cn or guodi@u.washington.edu).

Z. Liu is with Microsoft Research, One Microsoft Way, Redmond, WA 98052 USA (e-mail: zliu@microsoft.com).

X. Qu, is with Department of Communication Engineering, Xiamen University, Xiamen 361005 CHINA (e-mail: quxiaobo@xmu.edu.cn).

L. Huang is with Department of Communication Engineering, Xiamen University, Xiamen 361005 CHINA (corresponding author, phone: +86-592-2580142; fax: +86-592-2580038; e-mail: lfhuang@xmu.edu.cn).

Y. Yao is with Department of Communication Engineering, Tsinghua University, Beijing 100084 CHINA (e-mail: yaoy@tsinghua.edu.cn).

M. T. Sun is with Department of Electrical Engineering, University of Washington, Seattle, WA 98195 USA (e-mail: sun@ee.washington.edu).

modeling, sparse representation has been successfully used in medical imaging [11], face recognition [12], and compressive sensing for networked data [13]. Sparse representation approximates a signal with a linear combination of a small number of elementary signals called atoms. Often, the atoms are chosen from a so called overcomplete dictionary. Such dictionaries can offer a wider range of generating atoms, allowing more flexibility in the representation and adaptability to its content [11,14].

Guo *et al.* [15] adopted a 2-D Discrete Cosine Transform (DCT) basis as the sparsifying transform for realistic climate sensor data, but there is no guarantee that general transforms such as DCT, or Wavelet can sparsely represent the signal of interest [16]. Since we have the past data available at the sink, one possibility is to consider using the past data to construct the current frame. The investigation of exemplar-based sparsity for image processing can be found in literature [16,17]. Here, we propose to use an overcomplete dictionary composed of past-data and the 2-D DCT basis for the online data recovery. This dictionary is more effective than the fixed DCT basis, because with the temporal correlation, the current frame usually can be efficiently represented by a weighted linear combination of previous past frames. No off-line training phase is required. The recovery approach is simple enough to be implemented on the sink, with negligible delay compared to the sampling interval of the sensors.

After we recover the previous frame, it becomes one atom of the overcomplete dictionary for the current frame. If there exist some errors in the previous frame, the errors will propagate to affect the recovery of the current frame. In order to reduce the error propagation, we propose to leverage the available data in the current frame to correct the corresponding errors in the previous frame first, and then use this updated dictionary for recovering the missing data in the current frame. If some delay is allowed, future frames can be incorporated to further improve the estimation accuracy of the current frame.

Our contributions are summarized as following:

1) Propose a sparsity-based online data recovery method. An overcomplete dictionary composed of past data frames and the DCT basis is used to achieve higher sparsity.

2) Propose a scheme to reduce the error propagation by correcting the dictionary composed of recovered past frames.

3) Propose an approach that incorporates future frames to help the recovery of the current frame to improve the performance of the data recovery.

Our methods were tested on the data from a real sensor network application: monitoring the temperatures of the disk drive racks in a data center. Simulation shows that in terms of estimation accuracy and stability, the proposed approach outperforms existing average-based interpolation methods, and is more robust to burst missing along the time dimension.

II. Proposed method

A. Sparsity-based Recovery using Overcomplete Dictionary (SROD)

Considering a network with M sensor nodes, each node records the physical parameters of an environment at time intervals $1,2,\cdots,n,\cdots$. If all the sensors successfully collect these samples, samples of all the nodes can be arranged in a vector $\mathbf{f}_n = \left[f_1(n), f_2(n), \cdots, f_M(n) \right]^T$ to form the network data (the n^{th} frame). However, if some of the samples failed to be collected or transmitted to the sink due to hardware failure or environmental limitations, only a subset of $\mathbf{f}_n$ is observed. We can group the indices of the entries into two subsets: the first subset Λ_n consists of those indices of entries observed in $\mathbf{f}_n$; the second subset $\bar{\Lambda}_n$ consists of those indices of entries missed in $\mathbf{f}_n$. Correspondingly, $\mathbf{f}_n^{\Lambda_n}$ and $\mathbf{f}_n^{\bar{\Lambda}_n}$ are denoted as the available data and missing data in $\mathbf{f}_n$, respectively.

Using only general transforms such as the 2-D DCT basis may not sparsely represent the current frame. Since we have past frames available at the sink, and with the temporal correlation, the current frame often can be represented as a weighted linear combination of a few previous past frames, we propose to add some past frames to the dictionary besides the 2-D DCT basis, resulting in an overcomplete dictionary for online data recovery.

We assume the data in the frames are highly temporally correlated which is the case in most sensor network applications, since the sampling rate usually is controlled so that the sensor data do not change drastically within the sampling period. We verify this assumption by plotting the temporal correlation between frames in our temperature dataset [18] as shown in Fig.2. The temporal correlation is defined as:

$$R(\tau) = \frac{1}{\Delta n + 1} \sum_{n=n_0}^{n_0+\Delta n} \mathbf{z}_n^T \mathbf{z}_{n+\tau} \qquad (1)$$

where $\mathbf{z}_n$ is a normalized vector containing the sensor data in the n^{th} frame and $\Delta n + 1$ is the total number of frame pairs used in calculating $R(\tau)$. This temporal correlation averages the inner products for all the pairs of frames with time lag τ. Fig. 2(a) shows the temporal correlation in 200 frames. The high correlation of the frames indicates the feasibility of their sparse representation. We verify the sparse representation of frames in our case by Fig. 2(b), which shows the coefficients when representing the current frame using our proposed overcomplete dictionary. In the figure, only a few large coefficients exist in the representation, which means past frames can be used to sparsely represent the current frame.

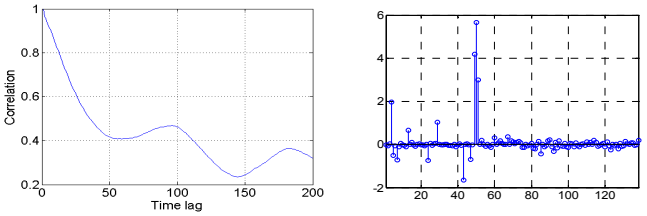


Fig. 2. (a) Data correlation between frames. (b) Coefficients when representing a frame using the proposed overcomplete dictionary.

To get started, we assume there is no missing data or the missing data have been recovered in the past L frames. Let $\mathbf{\Psi}$ denote the DCT basis, then the overcomplete dictionary for $\mathbf{f}_n$ is $\mathbf{\Phi}_n = [\mathbf{f}_{n-L} \cdots \mathbf{f}_{n-2}\ \mathbf{f}_{n-1}\ \mathbf{\Psi}]$, where $\mathbf{\Phi}_n$ is an $M \times (L+M)$ matrix with $\text{rank}(\mathbf{\Phi}_n) = M$, so that any signal can be represented by more than one combination of different atoms. We assume that the current frame can be represented as a sparse linear combination of the atoms in $\mathbf{\Phi}_n$, as shown in Fig. 3,

$$\mathbf{f}_n = \mathbf{\Phi}_n \boldsymbol{\alpha}_n \tag{2}$$

where $\boldsymbol{\alpha}_n \in \mathbb{R}^{(L+M)}$ is expected to be sparse, i.e. $\|\boldsymbol{\alpha}_n\|_0 \ll L+M$.

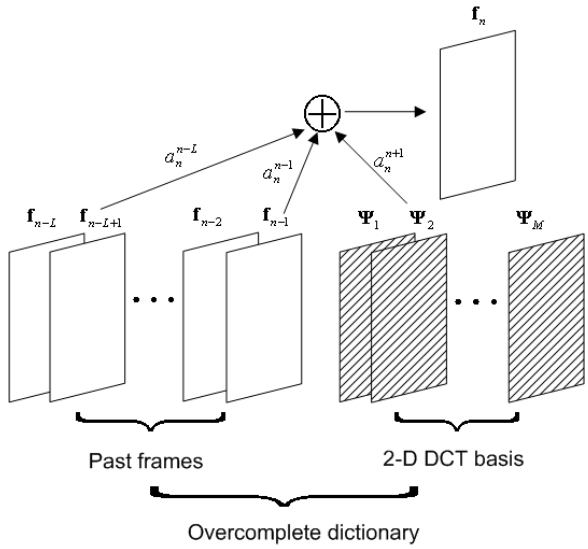


Fig. 3. The current frame is a sparse linear combination of atoms in an overcomplete dictionary. This overcomplete dictionary is composed of past L frames and the 2-D DCT basis.

With the notation of the available data $\mathbf{f}_n^{\Lambda_n}$ and the missing data $\mathbf{f}_n^{\bar{\Lambda}_n}$, the rows of $\mathbf{\Phi}_n$ can also be partitioned into two parts $\mathbf{\Phi}_n^{\Lambda_n}$ and $\mathbf{\Phi}_n^{\bar{\Lambda}_n}$ correspondingly. Thus, the current frame to be recovered in Eq. (1) can be rewritten as,

$$\begin{pmatrix} \mathbf{f}_n^{\Lambda_n} \\ \mathbf{f}_n^{\bar{\Lambda}_n} \end{pmatrix} = \begin{pmatrix} \mathbf{\Phi}_n^{\Lambda_n} \\ \mathbf{\Phi}_n^{\bar{\Lambda}_n} \end{pmatrix} \boldsymbol{\alpha}_n \tag{3}$$

Since $\mathbf{f}_n^{\bar{\Lambda}_n}$ is not known, we are unable to make any use of $\mathbf{f}_n^{\bar{\Lambda}_n} = \mathbf{\Phi}_n^{\bar{\Lambda}_n} \boldsymbol{\alpha}_n$. Our hope for finding $\boldsymbol{\alpha}_n$ relies on the equation corresponding to the available data,

$$\mathbf{f}_n^{\Lambda_n} = \mathbf{\Phi}_n^{\Lambda_n} \boldsymbol{\alpha}_n \tag{4}$$

The number of unknowns is more than the number of equations in Eq. (4), thus the system of equations is under-determined. Since we expect a priori that the presentation of the current frame will be sparse, $\boldsymbol{\alpha}_n$ can be estimated by solving the ℓ_1 norm optimization problem:

$$\arg\min_{\boldsymbol{\alpha}_n} \|\boldsymbol{\alpha}_n\|_1 \quad \text{s.t.} \quad \mathbf{f}_n^{\Lambda_n} = \mathbf{\Phi}_n^{\Lambda_n} \boldsymbol{\alpha}_n, \tag{5}$$

as long as $\mathbf{\Phi}_n^{\Lambda_n}$ satisfy the Restricted Isometry Property (RIP) [13, 16]. In other words, among all the solutions that satisfy the constraints, we select the one that has the smallest ℓ_1 norm, i.e., the sparsest solution. A relaxed version of this problem permits a small deviation in the representation, leading to the problem

$$\arg\min_{\boldsymbol{\alpha}_n} \|\boldsymbol{\alpha}_n\|_1 \quad \text{s.t.} \quad \left\|\mathbf{f}_n^{\Lambda_n} - \mathbf{\Phi}_n^{\Lambda_n} \boldsymbol{\alpha}_n\right\| \le \boldsymbol{\varepsilon} \tag{6}$$

where the constraint are for those components for which the data are not missing, and $\boldsymbol{\varepsilon}$ stands for the permissible deviation of the representation $\mathbf{\Phi}_n^{\Lambda_n} \boldsymbol{\alpha}_n$ from the original signal $\mathbf{f}_n^{\Lambda_n}$. One appealing method for solving Eq. (6) is Basis Pursuit Denoising (BPDN) [14],

$$\hat{\boldsymbol{\alpha}}_n = \arg\min_{\boldsymbol{\alpha}_n} \frac{1}{2}\left\|\mathbf{f}_n^{\Lambda_n} - \mathbf{\Phi}_n^{\Lambda_n} \boldsymbol{\alpha}_n\right\|_2^2 + \lambda \|\boldsymbol{\alpha}_n\|_1 \tag{7}$$

The solution to Eq. (7) is robust in the presence of noise, and also gives good performance even when the coefficient vector is not as sparse, which means $\mathbf{f}_n$ can be approximated with some error by truncating the small magnitude coefficients in $\boldsymbol{\alpha}_n$. The final recovered output $\mathbf{A}_n$ is

$$\mathbf{A}_n = \mathbf{\Phi}_n \hat{\boldsymbol{\alpha}}_n. \tag{8}$$

B. Recovery with Corrected Dictionary (RCD)

One problem of the basic scheme in Section A is the error propagation. Since the last frame $\mathbf{f}_{n-1}$ is used as a column of the dictionary, the error of the recovered $\mathbf{A}_{n-1}$ may propagate to the recovery of $\mathbf{f}_n$. In order to reduce the possible error propagation, we can leverage the available data $\mathbf{f}_n^{\Lambda_n}$ to correct $\mathbf{A}_{n-1}$ to some degree.

We consider each component r_n^m of $\mathbf{r}_n = \mathbf{f}_n - \mathbf{f}_{n-1}$ as identical and independent distributed (i.i.d) random samples drawn from a Gaussian function

$$p(r_n^m) = \frac{c_n}{\sqrt{2\pi}\sigma_n} e^{-\frac{(r_n^m - b_n)^2}{2\sigma_n^2}}, \tag{9}$$

where b_n is the mean, σ_n^2 is the variance, and $\frac{c_n}{\sqrt{2\pi}\sigma_n}$ is the height of the curve's peak. To verify that Gaussian distribution is a reasonable assumption, we select 200 frames from the real data and plot histograms of the difference $\mathbf{f}_n^m - \mathbf{f}_{n-1}^m$ for the m^{th} node, m = 1,2, …, M. Fig. 4 shows the histograms for two nodes. We can see that their histograms are close to a Gaussian distribution.

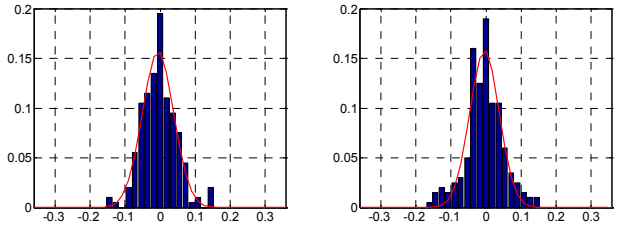
According to Eq. (9), the joint probability density function of $\{r_n^m\}_{m=1}^M$, which is also the likelihood of σ_n, becomes

$$\Gamma(\sigma_n|\mathbf{r}_n)=\prod_{m=1}^{M}\frac{c_n}{\sqrt{2\pi}\sigma_n}e^{-\frac{(r_n^m)^2}{2\sigma_n^2}}=(c_n)^M\left(2\pi\sigma_n^2\right)^{-\frac{M}{2}}e^{-\frac{\|\mathbf{r}_n\|_2^2}{2\sigma_n^2}}. \quad (10)$$

Thus, the negative log-likelihood is

$$-\log\Gamma(\sigma_n|\mathbf{r}_n)=\frac{\|\mathbf{r}_n\|_2^2}{2\sigma_n^2}+M\log\sqrt{2\pi}\sigma_n-M\log c_n. \quad (11)$$

Substituting $\mathbf{r}_n$ by $\mathbf{f}_n-\mathbf{f}_{n-1}$ and omitting the constant, we can maximize the loglikelihood of $\mathbf{r}_n$ by minimizing $\frac{1}{2\sigma_n^2}\|\mathbf{f}_n-\mathbf{f}_{n-1}\|_2^2$.



(a) (b)

Fig. 4 Plot histograms and then do Gaussian fitting of (a) 1st node, (b) 21st node.

Note that some of the components (nodes) in $\mathbf{f}_{n-1}$ are observed, while others are missing. The same is true for $\mathbf{f}_n$. We can group the indexes of the components into three subsets: Λ_{n-1}, consisting of those components that are observed in $\mathbf{f}_{n-1}$; $\bar{\Lambda}_{n-1}\bigcap\Lambda_n$, consisting of those components whose data are observed in $\mathbf{f}_n$ but not in $\mathbf{f}_{n-1}$; and $\bar{\Lambda}_{n-1}\bigcap\bar{\Lambda}_n$, consisting of those components whose data are missing in both $\mathbf{f}_{n-1}$ and $\mathbf{f}_n$.

In the original scheme where we recover $\mathbf{f}_{n-1}$ directly from $\mathbf{f}_{n-L-1},\mathbf{f}_{n-L},\cdots,\mathbf{f}_{n-2}$ (without leveraging $\mathbf{f}_n$), we essentially use $\mathbf{f}_{n-1}^{\Lambda_{n-1}}$ to solve for the coefficients $\boldsymbol{\alpha}_{n-1}$. Now, since $\mathbf{f}_n$ is available, we can leverage $\mathbf{f}_n^{\bar{\Lambda}_{n-1}\cap\Lambda_n}$ by adding a new term to Eq. (7). By minimizing $\frac{1}{\sigma_n^2}\left\|\mathbf{f}_n^{\bar{\Lambda}_{n-1}\cap\Lambda_n}-\mathbf{f}_{n-1}^{\bar{\Lambda}_{n-1}\cap\Lambda_n}\right\|_2^2$, and since $\mathbf{f}_{n-1}^{\bar{\Lambda}_{n-1}\cap\Lambda_t}=\boldsymbol{\Phi}_{n-1}^{\bar{\Lambda}_{n-1}\cap\Lambda_n}\boldsymbol{\alpha}_{n-1}$, we can replace the new term with $\frac{1}{\sigma_n^2}\left\|\mathbf{f}_n^{\bar{\Lambda}_{n-1}\cap\Lambda_n}-\boldsymbol{\Phi}_{n-1}^{\bar{\Lambda}_{n-1}\cap\Lambda_n}\boldsymbol{\alpha}_{n-1}\right\|_2^2$, and thus Eq. (7) becomes,

$$\begin{aligned}\hat{\boldsymbol{\alpha}}_{n-1}=\arg\min_{\boldsymbol{\alpha}_{n-1}}&\frac{1}{2}\left\|\mathbf{f}_{n-1}^{\Lambda_{n-1}}-\boldsymbol{\Phi}_{n-1}^{\Lambda_{n-1}}\boldsymbol{\alpha}_{n-1}\right\|_2^2+\lambda\|\boldsymbol{\alpha}_{n-1}\|_1\\&+\frac{\mu^2}{2\sigma_n^2}\left\|\mathbf{f}_n^{\bar{\Lambda}_{n-1}\cap\Lambda_n}-\boldsymbol{\Phi}_{n-1}^{\bar{\Lambda}_{n-1}\cap\Lambda_n}\boldsymbol{\alpha}_{n-1}\right\|_2^2\end{aligned} \quad (12)$$

where μ is a constant. This can also be written as,

$$\hat{\boldsymbol{\alpha}}_{n-1}=\arg\min_{\boldsymbol{\alpha}_{n-1}}\frac{1}{2}\left\|\mathbf{f}'-\boldsymbol{\Phi}'\boldsymbol{\alpha}_{n-1}\right\|_2^2+\lambda\|\boldsymbol{\alpha}_{n-1}\|_1 \quad (13)$$

where $\mathbf{f}'=\begin{bmatrix}\mathbf{f}_{n-1}^{\Lambda_{n-1}}\\ \frac{\mu}{\sigma_n}\mathbf{f}_n^{\bar{\Lambda}_{n-1}\cap\Lambda_n}\end{bmatrix}$, $\boldsymbol{\Phi}'=\begin{bmatrix}\boldsymbol{\Phi}_{n-1}^{\Lambda_{n-1}}\\ \frac{\mu}{\sigma_n}\boldsymbol{\Phi}_{n-1}^{\bar{\Lambda}_{n-1}\cap\Lambda_n}\end{bmatrix}$.

After we recover $\hat{\boldsymbol{\alpha}}_{n-1}$ in this way, we obtain

$$\mathbf{B}_{n-1}=\boldsymbol{\Phi}_{n-1}\hat{\boldsymbol{\alpha}}_{n-1}. \quad (14)$$

$\mathbf{B}_{n-1}$ would be a better recovered version of $\mathbf{f}_{n-1}$ than $\mathbf{A}_{n-1}$ which uses the approach described in Section A. We can think of the original formulation as a special case of the new formulation. If σ_n is infinity, which means there is no correlation between $\mathbf{f}_{n-1}$ and $\mathbf{f}_n$, then the new formulation becomes the same as the original formulation. When σ_n is not infinity, the new formulation leverages the correlation between the two neighboring frames in recovering the missing data in $\mathbf{f}_{n-1}$. However, the calculation of $\mathbf{B}_{n-1}$ requires the information from $\mathbf{f}_n$. Thus, for a causal system, $\mathbf{B}_{n-1}$ cannot be used as the output recovered frame for $\mathbf{f}_{n-1}$. To have a better recovery of $\mathbf{f}_n$, what we could do is to update the overcomplete dictionary $\boldsymbol{\Phi}_n$. That is, instead of using $\boldsymbol{\Phi}_n=[\mathbf{A}_{n-L}\ \cdots\ \mathbf{A}_{n-2}\ \mathbf{A}_{n-1}\ \boldsymbol{\Psi}]$, we use the new dictionary $\boldsymbol{\Phi}_n=[\mathbf{B}_{n-L}\ \cdots\ \mathbf{B}_{n-2}\ \mathbf{B}_{n-1}\ \boldsymbol{\Psi}]$ in the Eq. (7) to obtain a causal output $\mathbf{A}_n'$ as the recovered $\mathbf{f}_n$. Fig. 5 shows the process of this causal online data recovery system. For example, we assume that the first 100 frames are completely known a priori. When calculating 101st frame for the first time, represented as $\mathbf{A}_{101}$, 1st frame to 100th frame are used in the dictionary, as described in Section A. When 102nd frame arrives, we recalculate 101st frame, represented as $\mathbf{B}_{101}$, using 1st frame to 102nd frame, as described in Subsection B. Because $\mathbf{B}_{101}$ is a better version of $\mathbf{f}_{101}$ than $\mathbf{A}_{101}$, we update the dictionary with $\mathbf{B}_{101}$ and then compute $\mathbf{A}_{102}'$ using Eq. (7-8). The above process repeats until all the frames are recovered. Note that in evaluating the recovery error, we use $\mathbf{A}_n'$ instead of $\mathbf{B}_n$, so that the scheme is causal (i.e., the recovery of a frame does not depend on future frames).

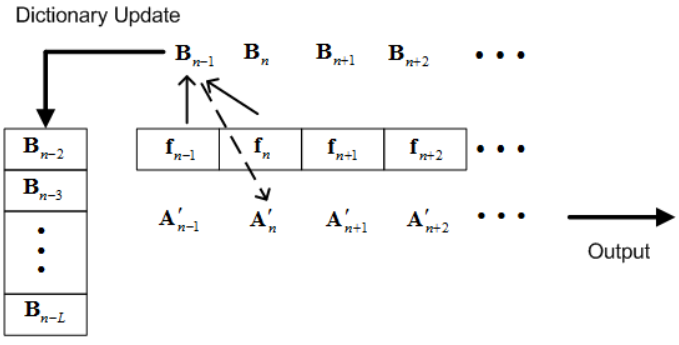


Fig. 5. Recovery with Corrected Dictionary (RCD). $\mathbf{A}'_n$ is the causal output of $\mathbf{f}_n$, n = 1,2,…. By leveraging the difference between the mutual components of two neighboring frames, $\mathbf{B}_{n-1}$ is a better version of $\mathbf{f}_{n-1}$ than $\mathbf{A}'_{n-1}$. For a causal system, $\mathbf{B}_{n-1}$ cannot be used as the recovered output of $\mathbf{f}_{n-1}$ since it uses the information from $\mathbf{f}_n$. However, it can be used to update the dictionary for computing $\mathbf{A}'_n$ for better performance.

C. Recovery with Future Frame Compensation (RFFC)

In Sections A and B, the data recovery only relies on the past and current frames. So these systems are causal and there is no delay for the output. However, for some practical systems, some delay is acceptable. For these systems, one can expect better performance if the future frames are incorporated to help the recovery of the current frame. We can use $\mathbf{B}_n$ as the output of recovered frame $\mathbf{f}_n$. More future frames (depending on the acceptable delay) can also be used to get better versions of $\mathbf{B}_n$. Let J denote the total number of future frames involved. Fig. 6 shows the process of data recovery with future frame compensation when $J = 2$.

Following the notations in Section B, there is a subset $\bar{\Lambda}_n \bigcap \Lambda_{n+j}$ which consists of those indices whose components are not observed for frames $\mathbf{f}_n$ but observed for the next j^{th} frame $\mathbf{f}_{n+j}$. The corresponding data are $\mathbf{f}_{n+j}^{\bar{\Lambda}_n \cap \Lambda_{n+j}}$. Note that subsets $\left\{\bar{\Lambda}_n \bigcap \Lambda_{n+j}\right\}_{j=1}^{J}$ can share common indices with each other, but not with $\mathbf{f}_n$.

Also assume each component of $\mathbf{f}_{n+j} - \mathbf{f}_n$ has a Gaussian distribution with mean zero and variance σ_{n+j}^2, the current frame can be recovered according to

$$\hat{\boldsymbol{\alpha}}_n = \arg\min_{\boldsymbol{\alpha}_n} \frac{1}{2}\left\|\mathbf{f}_n^{\Lambda_n} - \boldsymbol{\Phi}_n^{\Lambda_n}\boldsymbol{\alpha}_n\right\|_2^2 + \lambda\left\|\boldsymbol{\alpha}_n\right\|_1 + \sum_{j=1}^{J}\frac{\mu_{n+j}}{2\sigma_{n+j}^2}\left\|\mathbf{f}_{n+j}^{\bar{\Lambda}_n\cap\Lambda_{n+j}} - \boldsymbol{\Phi}_n^{\bar{\Lambda}_n\cap\Lambda_{n+j}}\boldsymbol{\alpha}_n\right\|_2^2 \tag{15}$$

The simplest case is $J = 1$, and $\boldsymbol{\alpha}_n$ can be computed as,

$$\hat{\boldsymbol{\alpha}}_n = \arg\min_{\boldsymbol{\alpha}_n} \frac{1}{2}\left\|\mathbf{f}_n^{\Lambda_n} - \boldsymbol{\Phi}_n^{\Lambda_n}\boldsymbol{\alpha}_n\right\|_2^2 + \lambda\left\|\boldsymbol{\alpha}_n\right\|_1 + \frac{\mu}{2\sigma_{n+1}^2}\left\|\mathbf{f}_{n+1}^{\bar{\Lambda}_n\cap\Lambda_{n+1}} - \boldsymbol{\Phi}_n^{\bar{\Lambda}_n\cap\Lambda_{n+1}}\boldsymbol{\alpha}_n\right\|_2^2 \tag{16}$$

and $\mathbf{B}_n = \boldsymbol{\Phi}_n\hat{\boldsymbol{\alpha}}_n$.

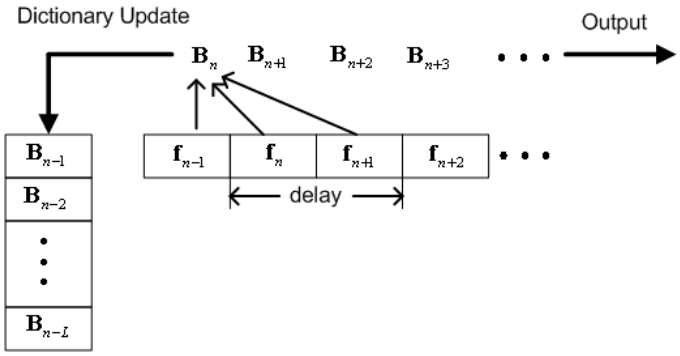


Fig. 6. Recovery with future frame compensation ($J = 2$). $\mathbf{B}_{n-1}$ is the noncausal output of $\mathbf{f}_n$, $n = 1, 2, \ldots$.

III. Simulations

A. Simulation Setup

Microsoft Dataset and Preprocessing

The dataset used in simulations comes from Microsoft Research Data Center Genome (DC Genome) system [18]. The goal of the project is to understand how the power is consumed in data centers and then to use this understanding to optimize and control the data center resources. To get an idea of the scenario, Fig. 7 shows that the temperature across the racks and across different heights of the same rack varies significantly [18].

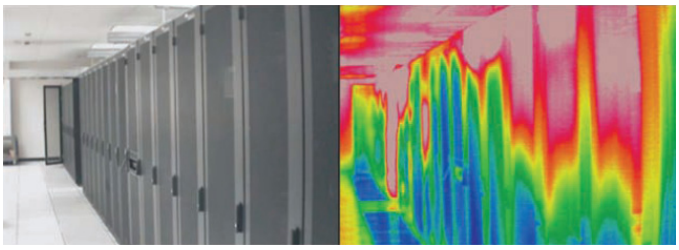


Fig. 7. The thermal image of an aisle in a data center. The infrared thermal image shows significant variations on intake air temperature across racks and at different height [18].

This dataset was recorded by sensors deployed in an 8×11 grid over a one-day period. Since about 10% of samples in the original dataset are missing, we use 2-D *K*-Nearest Neighbor (KNN) spatial interpolation algorithm [8], one of Inverse Distance Weighted Averaging (IDWA) interpolation methods, to fill in these missing samples before simulation, so that we have a complete data as the ground truth. Besides, we observed that the original data between 1st frame and 350th frame are during 0:00-11:40 a.m. when the temperatures in the data centre do not change much. The recovery for frames between 1st frame and 350th frame is comparable for the proposed methods and three-dimensional (3-D) *KNN* (will be shown in Table II later). Thus, we focus our proposed data recovery on data range from 351st frame to 700th frame which has a larger magnitude range so that the results are more meaningful. The completed dataset we used in the simulations are as shown in Fig. 8, and the sampling interval is two minutes. Then, we generate random and burst missing data patterns for 451st frame to 700th frame, and apply our proposed data recovery scheme to estimate the missing data.

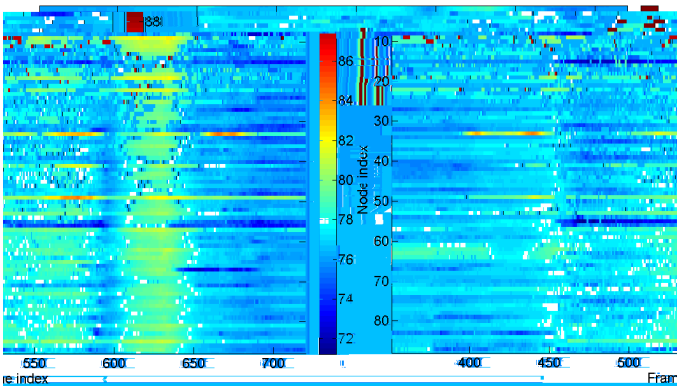


Fig. 8. Temperature data range from 351st frame to 720th frame. Its size is 8×11×370. Every two dimensional frame is arranged into a vector, so that the three-dimensional data are presented as a 2-D matrix. The color bar denotes the value of each sample.

Three Dimensional KNN

In the simulations, to extend 2-D *KNN* to 3-D *KNN* for our online data recovery scenario, the neighbors can only be chosen from the current frames and the past frames. Let (x, y) be the spatial location of a node, and n be the frame index. Adapting to anisotropic spatial and temporal correlation, we use a parameter η as the weighting of the temporal correlation relative to the spatial correlation, thus the distance is computed as,

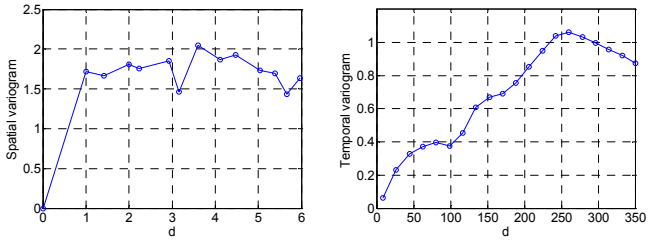
$$d = \sqrt{(x-x_0)^2 + (y-y_0)^2 + \eta(n-n_0)^2} \qquad (17)$$

where (x_0, y_0, n_0) and (x, y, n) are the coordinates of a node with missing sample and a neighboring node, respectively.

In order to pick an appropriate η, we investigate the spatial and temporal variograms. Assume $Z(x)$ represents a sample at location (x, y, n), then for a given distance d, the experimental variogram is defined as:

$$\gamma(d) = \frac{1}{2\#\Omega(d)} \sum_{\Omega(d)} \left[Z(x) - Z(x+d) \right]^2, \qquad (18)$$

where $\Omega(d)$ is the set of all data pairs with distance d, and $\#\Omega(d)$ is the number of data pairs with distance d. Fig. 9(a) shows the spatial variograms. The data values within $d = 1$ are highly correlated, while for other distances the process starts to look like an i.i.d process. Fig. 9(b) shows the temporal variogram. Comparing the spatial and temporal variograms, we pick the value for "η" as 0.01. For the 3-D *KNN* algorithm, the weight is inversely proportional to the distance d.



(a) (b)

Fig. 9. Variograms. (a) Spatial variogram, (b) Temporal variogram.

Evaluation Criterion

In order to evaluate the accuracy and stability of 3-D *KNN* and the proposed methods, we use Root Mean Squared Error (RMSE) and Maximal Absolute Error (MAE) calculated over all missing entries. Suppose $\mathbf{f}$ and $\hat{\mathbf{f}}$ are the original and recovered data vectors, respectively, and I is the total number of missing samples in $\mathbf{f}$. The RMSE is defined as,

$$\mathrm{RMSE}(\mathbf{f}, \hat{\mathbf{f}}) = \sqrt{\frac{\sum_{i=1}^{I} \left(f_i - \hat{f}_i \right)^2}{I}}, \quad i = 1, 2, \cdots, I, \qquad (19)$$

where f_i and $\hat{f}_i$ stand for ith missing entry of $\mathbf{f}$ and $\hat{\mathbf{f}}$, respectively. Note that all the observed entries remain the same in the data recovery process, and are excluded when calculating the recovery errors. Besides, the MAE of all missing entries is also used to assess the estimation stability. The MAE is defined as,

$$\mathrm{MAE}(\mathbf{f}, \hat{\mathbf{f}}) = \max \left[\left\{ \left| f_i - \hat{f}_i \right| \right\} \right] \text{ for } i = 1, 2, \cdots, I, \qquad (20)$$

where f_i and $\hat{f}_i$ are as defined above.

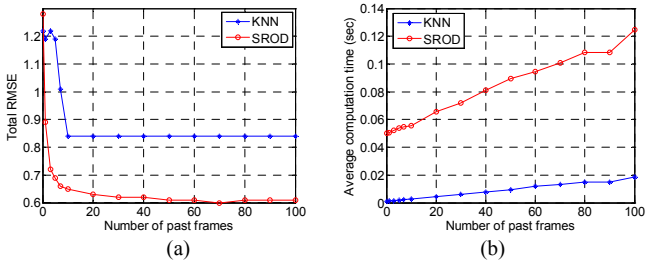
Specifically, we choose four evaluation metrics, including: (a) MAE of all nodes in each frame, (b) MAE of all frames in each node, (c) RMSE of all nodes in each frame, and (d) RMSE of all frames in each node. They evaluate the accuracy and stability of the methods node-by-node and frame-by-frame.

Parameter setting

To solve the ℓ_1 norm minimization in Eq. (8), (12) and (16), the "Sparselab 2.1" toolbox [19] is used. The parameters used in the proposed methods are $\lambda = 1 \times 10^{-3}$, $\mu = 0.1$, $\eta = 0.01$, and σ_n is automatically estimated directly from the past data frames. The number of nearest neighbors in 3-D *KNN* is $K = 9$. The size of each frame is 8×11, and the size of 2-D DCT basis is also 8×11. We generate burst missing patterns, i.e., the same node continuously missing samples along the temporal dimension, and the duration of time is defined as the *burst missing length*. In the simulations, we choose different missing rates (u) and burst missing lengths (v), and compare the above four evaluation metrics of 3-D *KNN* and the proposed approaches.

B. SROD vs. 3-D KNN

First, we discuss how the performance and complexity of *SROD* change with respect to the number of previous frames in the overcomplete dictionary. Compared with pure DCT dictionary, adding past frames into the dictionary can dramatically reduce the recovery error. This recovery error can be further reduced by increasing the number of past frames. However, further reduction of recovery error is not obvious when the number of past frame exceeds a threshold. As shown in Fig. 10(a), the recovery error decreases as the number of past frames increases. When the number of past frames is larger than 70, the recovery error no longer decreases significantly. On the other hand, increasing the number of past frames will increase the computation time [see Fig. 10(b)]. In our simulations, using 50 past frames is a reasonable choice to tradeoff the recovery error and the computation time. This is the default number of past frames in the following simulations.



(a) (b)

Fig. 10. Recovery error and computation time of *KNN* and *SROD*. $u = 20\%$, $v = 10$. (a) Recovery error with respect to the numbers of past frames in the dictionary. (b) Computation time with respect to the numbers of past frames in the dictionary.

Fig. 11 shows an overcomplete dictionary, which is composed of 50 past 8x11 2-D frames and an 8x11 2-D DCT basis.

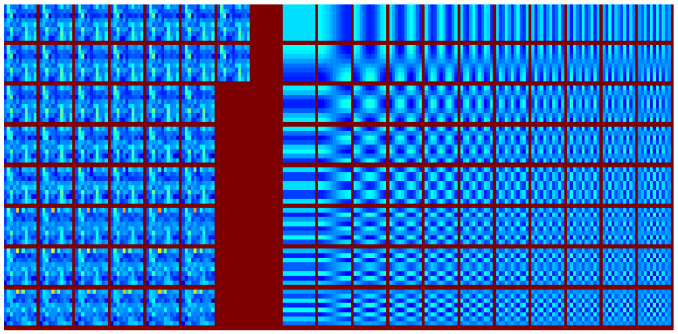


Fig. 11. The overcomplete dictionary for the 451st frame. Left half contains 50 past data frames which change over time, and each frame is with mean zero and normalized to 1. Right half is the fixed 8x11 2-D DCT basis.

With this dictionary, Table I shows the performance of the *SROD* and *KNN*, by summarizing the mean values of the above performance metrics for all the 451st ~700th recovered frames. u = 10% and 20%, v = 5 and 10. The proposed approach outperforms *KNN* in terms of these evaluation metrics with more than 20% improvement. The mean error values of *KNN* and *SROD* both increase as the missing rate and missing burst length go up, but the error of the proposed method is still much lower than that of *KNN*. The maximal error values demonstrate that the proposed approach is more robust to different missing rates and missing burst lengths.

We also compare *KNN* and *SROD* for 101st frame to 350th frame, and their performances are as shown in Table II. When temperature variation is small along the time dimension, the two methods can both reasonably recover the missing samples, and their performances are comparable.

TABLE I. PERFORMANCE COMPARISON OF *KNN* AND *SROD*

Methods / Mean	*KNN*				*SROD*			
	10%		20%		10%		20%	
	5	10	5	10	5	10	5	10
MAE_frame	1.31	1.40	1.54	1.88	0.88 (32.8%)	1.11 (20.7%)	1.19 (22.7%)	1.49 (20.7%)
MAE_node	1.48	1.48	1.75	1.80	1.06 (28.4%)	1.21 (18.2%)	1.39 (20.6%)	1.50 (16.7%)
RMSE_frame	0.66	0.69	0.66	0.78	0.43 (34.8%)	0.53 (19.7%)	0.47 (28.8%)	0.58 (25.6%)
RMSE_node	0.68	0.67	0.69	0.77	0.43 (36.8%)	0.52 (23.5%)	0.47 (31.9%)	0.56 (27.3%)
Total RMSE	**0.76**	**0.75**	**0.73**	**0.84**	**0.47 (38.2%)**	**0.57 (25.0%)**	**0.51 (30.1%)**	**0.63 (25.0%)**

From 451st to 700th frame. u = 10% and 20%, v = 5 and 10. The numbers between brackets are percentage improvement of the evaluation criteria of the *SROD* relative to that of *KNN*, at the same u and v.

TABLE II. PERFORMANCE COMPARISON OF *KNN* AND *SROD*

Methods / Mean	*KNN*				*SROD*			
	10%		20%		10%		20%	
	5	10	5	10	5	10	5	10
MAE_frame	0.43	0.51	0.57	0.55	0.41 (4.7%)	0.44 (13.7%)	0.55 (3.5%)	0.54 (1.8%)
MAE_node	0.49	0.47	0.56	0.58	0.47 (4.1%)	0.42 (10.6%)	0.55 (1.8%)	0.57 (1.7%)
RMSE_frame	0.21	0.23	0.22	0.22	0.19 (9.5%)	0.20 (13.0%)	0.20 (9.1%)	0.21 (4.5%)
RMSE_node	0.21	0.21	0.21	0.23	0.19 (9.5%)	0.18 (14.3%)	0.20 (4.8%)	0.21 (8.7%)
Total RMSE	**0.25**	**0.25**	**0.24**	**0.25**	**0.23 (8.0%)**	**0.21 (16.0%)**	**0.23 (4.2%)**	**0.24 (4.0%)**

From 101st to 350th frame. u = 10% and 20%, v = 5 and 10. The numbers between brackets are percentage improvement of the evaluation criteria of the *SROD* relative to that of *KNN*, at the same u and v.

Fig. 12 shows the MAE and RMSE of the two methods when u = 10% and v = 5, node by node, and frame by frame. On the majority of the nodes, *SROD* outperforms *KNN* in terms of both MAE and RMSE. On each frame, *SROD* shows larger improvement when the data change fast [see Fig.12(d) and Fig. 8].

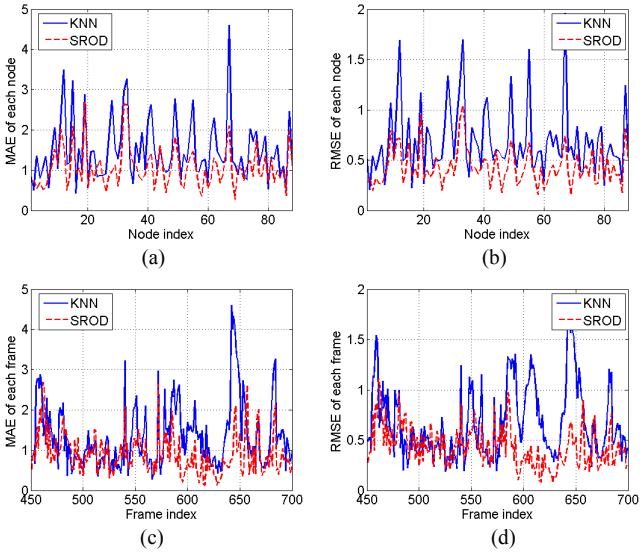


Fig. 12. Performance comparison for *KNN* and *SROD*. u = 10%, v = 5. (a) MAE of all frames in each node, (b) RMSE of all frames in each node, (c) MAE of all nodes in each frame, (d) RMSE of all nodes in each frame.

C. Performance of RCD and RFFC

In order to learn the performance of the two extended methods: recovery with corrected dictionary (*RCD),* and recovery with future one frame compensation *(RFFC-1),* we compare them with *SROD*.

For fair comparison, *KNN* is also developed into *KNN-CD* and *KNN-FFC-1*, respectively. Correspondingly, *KNN-CD* updates the last frame by utilizing the available data of the current frame before estimating the missing data of the current frame, while *KNN-FFC-1* estimates the missing data of the current frame by using the past frames and the next frame.

In Table III, we set u = 20% and v = 1. The proposed approaches all perform much better than *KNN*, for about 30% improvement. For the proposed approaches, *RCD* performs slightly better than *SROD*; and *RFFC-1* sees a larger improvement than *SROD*, by about 15%. This can be seen more clearly in Fig. 13, which compares the performances of *SROD* and *RFFC-1*, in a node-by-node and frame-by-frame way. *RFFC-1* can greatly improve the recovery quality of the *SROD* because the next frame introduces more new information.

In Table IV, u = 20% and v = 5. The proposed approaches also all perform much better than *KNN (by about* 20%). For the proposed approaches, the performance of *RCD* is the same as *SROD*, and *RFFC-1* has another 5% improvement over *SROD*. The improvements are not as big as that in Table III. The reason is that when the missing burst length become longer, the newly-introduced components becomes fewer, thus, it will not make much difference to correct one column of the dictionary.

TABLE III. PERFORMANCE COMPARISON OF SIX METHODS

Methods / Mean	*KNN*			*Proposed*		
	KNN	*KNN-CD*	*KNN-FFC-1*	*SROD*	*RCD*	*RFFC-1*
MAE_frame	1.55	1.54 (0.6%)	1.46 (5.8%)	0.97 (37.4%)	0.95 (38.7%)	0.79 (49.0%)
MAE_node	1.80	1.79 (0.6%)	1.73 (3.9%)	1.25 (30.6%)	1.24 (31.1%)	1.08 (40.0%)
RMSE_frame	0.66	0.66 (-)	0.62 (6.1%)	0.38 (42.4%)	0.37 (43.9%)	0.30 (54.5%)
RMSE_node	0.69	0.69 (-)	0.65 (5.8%)	0.39 (43.5%)	0.38 (44.9%)	0.32 (53.6%)
Total RMSE	**0.72**	**0.72 (-)**	**0.69 (4.2%)**	**0.43 (40.3%)**	**0.42 (41.7%)**	**0.36 (50.0%)**

$u = 20\%$, $v = 1$. The numbers between brackets are percentage improvement of the evaluation criteria of the other methods relative to that of *KNN*, at the same u and v.

TABLE IV. PERFORMANCE COMPARISON OF SIX METHODS

Methods / Mean	*KNN*			*Proposed*		
	KNN	*KNN-CD*	*KNN-FFC-1*	*SROD*	*RCD*	*RFFC-1*
MAE_frame	1.54	1.54 (-)	1.54 (-)	1.19 (22.7%)	1.16 (24.7%)	1.08 (29.9%)
MAE_node	1.75	1.75 (-)	1.75 (-)	1.39 (20.6%)	1.37 (21.7%)	1.29 (26.3%)
RMSE_frame	0.66	0.66 (-)	0.66 (-)	0.47 (28.8%)	0.46 (30.3%)	0.42 (36.4%)
RMSE_node	0.69	0.69 (-)	0.68 (1.4%)	0.47 (31.9%)	0.46 (33.3%)	0.43 (37.7%)
Total RMSE	**0.73**	**0.73 (-)**	**0.73 (-)**	**0.51 (30.1%)**	**0.50 (31.5%)**	**0.47 (35.6%)**

$u = 20\%$, $v = 5$. The numbers between brackets are percentage improvement of the evaluation criteria of the other methods relative to that of *KNN*, at the same u and v.

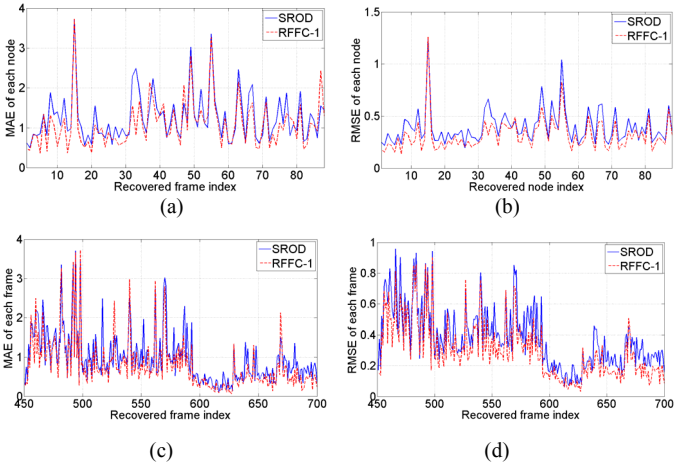


Fig. 13. Performance comparisons for the proposed approaches: *SROD* and *RFFC-1*. $u = 20\%$, $v = 1$. (a) MAE of all frames in each node, (b) RMSE of all frames in each node, (c) MAE of all nodes in each frame, (d) RMSE of all nodes in each frame.

To investigate the tradeoff between the performance and the delay in the *RFFC* scheme, we compare *RFFC-1* with *RFFC-2* and *RFFC-3*. Table V shows their performances when $u = 20\%$, $v = 1$ and 5. We can see that *RFFC-2* results in smaller errors than *RFFC-1*. When $v = 1$, *RFFC-3* achieves very similar performance to *RFFC-2*; when $v = 5$, *RFFC-3* can further reduce the recovery error. Using more future frames will not improve the performance, because the correlation between the future frame and the current frame becomes weaker as the future frame is further away.

From above comparisons, *KNN* interpolates the missing sample by averaging the values of K-nearest (temporally or spatially) neighbors, including both the past frames and the future frames, and they are not necessarily the most correlated samples. In contrast, the proposed methods find a small number of the most correlated frames (near or far) from a period of frames via ℓ_1 minimization to recover the current frame, thus the use of past frames and the future frames benefits the proposed methods significantly more than *KNN*.

TABLE V. PERFORMANCE COMPARISON OF *RFFC-1*, *RFFC-2* AND *RFFC-3*

Methods / Mean	1			5		
	RFFC-1	*RFFC-2*	*RFFC-3*	*RFFC-1*	*RFFC-2*	*RFFC-3*
MAE_frame	0.79	0.75 (5.1%)	0.75 (5.1%)	1.08	1.02 (5.6%)	1.00 (7.4%)
MAE_node	1.08	1.01 (6.5%)	1.01 (6.5%)	1.29	1.24 (3.9%)	1.22 (5.4%)
RMSE_frame	0.30	0.28 (6.7%)	0.28 (6.7%)	0.42	0.40 (4.8%)	0.39 (7.1%)
RMSE_node	0.32	0.29 (9.4%)	0.29 (9.4%)	0.43	0.41 (4.7%)	0.40 (7.0%)
Total RMSE	**0.36**	**0.33 (8.3%)**	**0.33 (8.3%)**	**0.47**	**0.44 (6.4%)**	**0.43 (8.5%)**

$u = 20\%$, $v = 1$ and 5. The numbers between brackets are percentage improvement of the evaluation criteria of the other methods relative to that of *RFFC-1*, at the same u and v.

We have also investigated the block-size effect to the proposed algorithm. Essentially, the whole frame can be considered as a large $P \times Q$ block of pixels. Intuitively, by breaking it into smaller $p \times q$ blocks, the data in each smaller block could have higher correlation and could find a better match from the entries in the dictionary. We have investigated using various smaller block-sizes. However, we found the performance is about the same as treating the whole frame as a large $P \times Q$ block with the data we used in our simulations. The reason is that this dataset has relatively low spatial resolution, which leads to weak homogeneity in local regions, and that limits the advantage of block partition.

D. Effect of noise

In reality, the data are usually corrupted with noise. We add white Gaussian noise to the available data, and the noisy measurement vector is written as

$$\mathbf{y} = \mathbf{f}_n^{\Lambda_n} + \boldsymbol{\varepsilon}, \tag{21}$$

where $\boldsymbol{\varepsilon}$ is the Gaussian noise, whose power is controlled by signal-to-noise ratio (SNR) defined as

$$\mathrm{SNR}_{\mathrm{dB}} = 10\log_{10}\frac{\left\|\mathbf{f}_n^{\Lambda_n}\right\|_2^2}{\left\|\boldsymbol{\varepsilon}\right\|_2^2}, \tag{22}$$

and then use BPDN algorithm

$$\hat{\boldsymbol{\alpha}}_n = \arg\min_{\boldsymbol{\alpha}_n}\frac{1}{2}\left\|\mathbf{y} - \boldsymbol{\Phi}_n^{\Lambda_n}\boldsymbol{\alpha}_n\right\|_2^2 + \lambda\left\|\boldsymbol{\alpha}_n\right\|_1 . \tag{23}$$

Fig. 14 shows the estimation error of *SROD* under different noise levels. As the noise increases, i.e., SNR drops, the curves of total RMSE basically remain the same, until SNR is as low as 10dB. This demonstrates that *SROD* is robust to low and medium noise.

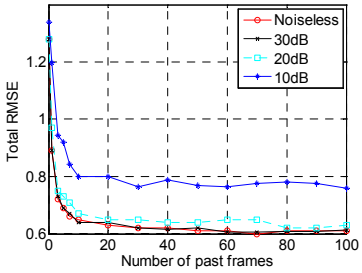


Fig. 14. Total RMSE of *SROD* for noisy data. $u = 20\%$, $v = 10$.

E. Effect of the payload size

In the type of sensor network applications we are addressing (e.g., temperature sensing), each node only reports its temperature reading. In order to achieve short delay, we assume that one packet contains only one reading number. Thus, the payload size should not be an issue. Each packet loss will result in one missing number in the frame.

For the more general case when the delay is not a concern, and no encoding is performed, one packet may contain S original reading numbers. Table VI shows performances of *KNN* and *SROD* under different payload sizes S = 1, 2, 5, 10. The performance of *KNN* remain almost the same when payload size is smaller than 5, but become significantly worse when payload size is 10. *SROD* performs gradually worse when the payload size grows. However, for the payload size we compared in Table V, *SROD* still outperforms the *KNN*. Thus, the proposed method is more robust with respect to the payload size.

Table VI. PERFORMANCE COMPARISON OF *KNN* AND *SROD* WITH DIFFERENT PAYLOAD SIZES

Methods / Mean	*KNN*				*SROD*			
	Payload size				Payload size			
	1	2	5	10	1	2	5	10
MAE_frame	1.56	1.56	1.55	1.77	1.02 (34.6%)	1.13 (27.6%)	1.14 (26.5%)	1.44 (18.6%)
MAE_node	1.82	1.72	1.73	1.84	1.30 (28.6%)	1.31 (23.8%)	1.33 (23.1%)	1.47 (20.1%)
RMSE_frame	0.66	0.67	0.66	0.78	0.40 (39.4%)	0.44 (34.3%)	0.49 (25.8%)	0.59 (24.4%)
RMSE_node	0.68	0.68	0.69	0.77	0.41 (39.7%)	0.44 (35.3%)	0.50 (27.5%)	0.57 (26.0%)
Total RMSE	**0.72**	**0.72**	**0.72**	**0.85**	**0.44 (38.9%)**	**0.48 (33.3%)**	**0.52 (27.8%)**	**0.63 (25.9%)**

u = 0.2. Payload sizes are 1, 2, 5 and 10. The numbers between brackets are percentage improvement of the evaluation criteria of *SROD* relative to that of *KNN*, at the same payload size.

IV. CONCLUSION

We presented new approaches for online data recovery in wireless sensor networks. A sparse linear relationship between a current frame and its past frames are modeled to estimate the missing data in the current frame. An overcomplete dictionary which is composed of the past data and the DCT basis is chosen to sparsely represent the current frame. Data recovery is achieved through ℓ_1 norm optimization. Correcting the last frame based on the current frame helps to prevent the error propagation. If some delay is allowed, the future frame can be used to further improve the performance. Simulation results on a real sensor data set demonstrate that the proposed approaches outperform the average-based interpolation methods in terms of both accuracy and robustness. Future work includes other data-based dictionaries for higher sparsity, and adaptive block partition methods.

ACKNOWLEDGMENT

The authors would like to thank Dr. Jie Liu in Microsoft providing the real sensor dataset. D. Guo thanks Qirong Ma for discussion. The authors would also like to thank the anonymous reviewers for their thorough reviews and comments.

REFERENCES

[1] T. Mitchell. (1999, March 27). "50" km resolution daily precipitation for the Pacific Northwest, 1949–94 [Online]. Available: http://www.jisao.washington.edu/data/widmann/

[2] R. Szewczyk, E. Osterweil, J. Polastre, M. Hamilton, A. Mainwaring, and D. Estrin, "Habitat monitoring with sensor networks," *Communications of the ACM*, vol. 47, no. 6, pp. 34–40, 2004.

[3] K. Lorincz, D. J. Malan, T. R. F. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, M. Welsh, and S. Moulton, "Sensor networks for emergency response: Challenges and opportunities," *Pervasive Comput.*, vol. 3, no. 4, pp. 16–23, 2004.

[4] A. Ledeczi, T. Hay, P. Volgyesi, D.R. Hay, A. Nadas, S. Jayaraman, "Wireless Acoustic Emission Sensor Network for Structural Monitoring," *IEEE Sensors J.*, vol. 9, no. 11, pp. 1370–1377, Nov. 2009.

[5] A. Gaddam, S.C. Mukhopadhyay, G.S. Gupta, "Elder Care Based on Cognitive Sensor Network," *IEEE Sensors J.*, vol. 11, no. 3, pp. 574–581, Mar. 2011.

[6] L. Gruenwald H. Chok and M. Aboukhamis, "Using data mining to estimate missing sensor data," in *IEEE 7th Int. Conf. Data Mining*, Oct. 2007, pp. 207–212.

[7] M. C. Vuran, Ö. B. Akan, and I. F. Akyildiz, "Spatio-temporal correlation: theory and applications for wireless sensor networks," *Computer Networks*, vol. 45, no.3, pp. 245–259, Jun. 2004.

[8] G. Y. Lu and D. W. Wong, "An adaptive inverse-distance weighting spatial interpolation technique," *Computers & Geosciences*, vol. 34, no.9, pp. 1044–1055, 2008.

[9] M. Umer, L. Kulik, and E. Tanin, "Kriging for Localized Spatial Interpolation in Sensor Networks," in *Scientific and Statistical Database Management*. vol. 5069, B. Ludäscher and N. Mamoulis, Eds. New York: Springer Berlin / Heidelberg, 2008, pp. 525-532.

[10] A. Deshpande, C. Guestrin, S. R. Madden, J. M. Hellerstein, and W. Hong, "Model-driven data acquisition in sensor networks," in *Proc. 30th int. conf. Very Large Data Bases* 2004, pp. 588-599.

[11] M. J. Fadili and J. L. Starck, "Sparse representation-based image deconvolution by iterative thresholding," in *Proc. Astronomical Data Analysis*, Marseille, France, Sept. 2006.

[12] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 210–227, 2008.

[13] J. Haupt, W. U. Bajwa, M. Rabbat, and R. Nowak, "Compressed sensing for networked data," *IEEE Signal Process. Mag.*, vol. 25, no.2, pp. 92–101, 2008.

[14] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Review*, vol. 43, no. 1, pp. 129-159, 2001.

[15] D. Guo, X. Qu, L. Huang, and Y. Yao, "Sparsity-Based Spatial Interpolation in Wireless Sensor Networks," *Sensors*, vol. 11, no.3, pp. 2385–2407, 2011.

[16] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. New York: Springer, 2010, pp. 227–228.

[17] K. S. Gurumoorthy, A. Rajwade, A. Banerjee, and A. Rangarajan, "A method for compact image representation using sparse matrix and tensor projections onto exemplar orthonormal bases," *IEEE Trans. Image Process.*, vol. 19, no.2, pp. 322–334, 2010.

[18] J. Liu, F. Zhao, J. O'Reilly, A. Souarez, M. Manos, C. J. M. Liang, and A. Tersiz. (2008, December). Project genome: Wireless sensor network for data center cooling. *The Architecture Journal* [Online], vol. 18, pp. 28–34. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=78813

[19] D. Donoho, Stodden, V., Tsaig, Y. (2007, May 26). Sparselab 2.1 [Online]. Available: http://sparselab.stanford.edu/