A Study of Simple Partially-Recovered Sensor Data Imputation Methods

Christoph Sydora
Department of Computing Science
University of Alberta
Edmonton, AB, Canada
csydora@ualberta.ca

Johannes Jung
Department of Informatics
Technical University of Munich
Munich, Germany
johannes.jung@tum.de

Ioanis Nikolaidis
Department of Computing Science
University of Alberta
Edmonton, AB, Canada
nikolaidis@ualberta.ca

***Abstract*—We consider the problem of loss of continuous data feeds from sensor networks, due to transient failures. Because the failures are recoverable, part of the missing data may be, eventually, acquired. Even then, the limited resources of the nodes can result in an incomplete reconstruction of the missing data. In this paper we study a set of proposed data imputation methods, and their variations, on a real data set. We determine the tradeoffs involved in the proposed techniques. A common characteristic of the studied techniques is that they depend on the recent behavior of the data stream and do not make specific assumptions about the long-term stochastic behavior of the data. We consider also the case where simple, sub-sampling based, handling of accumulated missing data is implemented by the nodes.**

***Index Terms*—Data Imputation, Wireless Sensor Networks**

I. Introduction

Missing data is an inherent problem in Wireless Sensor Networks (WSNs) due to node or communication failures that occur for a number of reasons, such as wireless medium interference, jamming attempts, power failures, physical damage to the sensor nodes, unreliable or aged electronic components, etc. [1]. These issues can lead to short-term or longer-term, transient, sensor "blackouts" which in turn can result in gaps in the collected data. Since the deployment of WSNs is primarily performed to acquire data to inform or automate tasks, gaps in the data may inhibit the ability to carry out those tasks. A strategy is needed of how to handle missing data such that the intended application of the WSN continues to be fulfilled, correctly, and without undue difficulty [2].

Two general approaches exist: (a) to recover the missing data in some fashion, isolating the application from data blackouts of any number of nodes, or, (b), to handle such failures within the application. The first option may potentially compromise the correctness of the application, while the second requires that contingency plans are developed and implemented by the application for a (typically large and complex) variety of failure scenarios. The two approaches are not necessarily mutually exclusive. For example, a WSN can recover missing data to allow the normal operation of the application, but also provide some notification to the application signaling that it is now operating with recovered, and possibly less reliable, data. We adopt option (a) and study alternatives of how to determine the missing data, evaluated in terms of their root mean square error (RMSE). A notable characteristic of our work is that we do not assume any particular long-term stochastic behavior of the data streams. The evaluated schemes simply have a short-term "window" of the data over which they operate and whose gaps they attempt to fix.

Specifically, in this paper we consider techniques for recovering the missing data to allow the continuous operation of the WSN application. We are motivated by our own experience in collecting data from a multi-apartment residential building. Our evaluation is therefore making use of a specific sensor type that was capable of providing CO2, relative humidity (RH) and ambient temperature readings. Over a multi-year data collection, there were gaps in the collected data. We speculate that most of the failures were due to power failures that were, subsequently, recovered. We note that powering on/off the sensors was possible by the residents (either intentionally or accidentally). As we describe in the experiments sections, we found a long period of nearly pristine data, with few missing data gaps, which we subsequently used as ground truth data. It is on this pristine ground truth data that we are performing mutation experiments to evaluate the efficacy of various recovery techniques. The intention was to be better informed about which technique we should apply to the gaps in other parts of the data stream.

Overall, we follow the typical model for a WSN according to which data is collected to a single node, the *sink* [3]. The sink is the location where the data maintenance and analysis can be performed and where complete data can be relayed to the cloud or directly to the system actuators. Typically, the sink is connected to the wired infrastructure, and hence it is also assumed to have no energy limitations as it could be connected to a continuous power supply. If we exclude the sink, all the remaining, "regular," WSN nodes are assumed to have limited storage and computation resources, and hence require little energy to operate and small size to enable their ubiquitous deployment.

Another practical form of faults covered by our study are faults restricted *only* to the communication transceiver but not to the entire sensor node. Then, it is possible for sensing to proceed independently of whether the transceiver is powered or not. This is due to the elevated wireless transceiver energy

978-3-903176-24-9 © 2019 IFIP

demands, making them prime candidates for powering down when energy is scarce (in conditions of low battery charge – until batteries are replaced, or due to unavailability of harvested energy, etc.). Hence, node failures can be perceived as inability to communicate, but still being able to collect sensed data that are later sent when communication is restored. The limitation then becomes that of the storage available on the sensor node. We capture the limited storage resources of sensor nodes, by following a technique similar to that by Raza et al. [4] to handle the storage of data over prolonged failures. That is, the impact on the recovered data is both a side-effect of the failure intervals as well as of the storage management strategy. The data recovered after communication is re-established are used to improve past estimations by re-estimating the missing data.

In this research, a collection of sensor data imputation methods proposed in the literature, with some variations, are evaluated under different sensor failure circumstances, using a real dataset. We selected five methods for missing data estimation on the basis of how frequently they appeared in various articles and their ability to be reproduced. We consider the impact of various parameters that control an underlying Markovian model of node failures. Additionally, we adopt a partial sub-sampling strategy assuming the nodes that cannot communicate can still sense, albeit the available storage for their samples is limited and have to evict measurements if their storage buffer is full. We assume that the micro-controller used in the sensor is of meager capabilities, such as an 8051 or an MSP430. Therefore, no more than a handful of measurements can be kept in micro-controller RAM and external storage is prohibitive due to its own additional energy demands.

The rest of this paper is structured as follows: Section II presents a summary of related work. Section III introduces the missing data estimation methods evaluated in the remaining of the paper. Section IV outlines the models used for the node failures, and for the storage management of the sensor nodes. Section V presents information related to the evaluation process and the real datasets used. Section VI presents the evaluation results. Section VII provides a discussion of the results and provides evidence linking the observed performance to the "sparsity" of the underlying signals. Finally, section VIII concludes the paper with an overview of the results and directions for future work.

II. Related work

Extensive literature exists on the topic of statistical imputation techniques. Specifically for WSNs, the existing proposed methods can be categorized into methods that exploit temporal correlation of values produced by a sensor node, and those that exploit spatial correlations, i.e., correlations of sensor data produced simultaneously from sensors placed at multiple locations [2], [5], [6], or a combination of both spatial and temporal correlations [3], [7]–[15].

In [7], [8] the authors used a method called time-space correlation which uses a neural network. Jie et al. [9] used a Back Propagation (BP) Neural Network for spatial correlations and Linear Regression model for temporal regression and then weighted the two methods based on Pearson correlations of the sensor nodes. Li and Parker [16] used a Fuzzy Adaptive Resonance Theory (ART) Neural Network to perform classification of data events with missing data. Yan et al. [10] made use of piecewise cubic interpolation for their temporal correlation. They also used Data Estimation using Statistical Methods (DESM) for their spatial correlation estimation and used a weighted sum of the two based on the Pearson correlation coefficient.

Zhang and Yang [11] used multiple regression for both their Spatial Method (SM) and Temporal Method (TM). They then combined the two methods to create STM and DC methods which appeared to use the TM or SM method that had the smallest error. Zhen [12] also used linear regression together with a second exponential smoothing method.

Ren et al. [13] created a decision tree which, based on spatial and temporal Pearson correlation values, determined which method to use. The methods proposed were Temporal Correlation Algorithm Linear Interpolation (TCA-LI), Temporal Correlation Algorithm Multiple Regression (TCA-MR), Spatial Correlation Algorithm (SCA), or Spatial Temporal Correlation Algorithm (STCA).

Pan et al. [5] used the nearest neighbour estimates using the regression (R^2) values for weights for each. The same group also published results using Adaptive Multiple Regression (AMR) algorithm using spatial data [2] and in addition, used piecewise cubic interpolation for additional temporal correlations [14]. Lou et al. [6] also used a Nearest Neighbour search. Kumar et al. [3] proposed Prediction using a Spatial-Temporal Correlation (PSTC) algorithm which used the Pearson Correlation as a weight to surrounding nodes.

Finally, Gao et al. [15] presented Temporal and Spatial Correlation Algorithm (TSCA) which combined a weighted temporal rate of change estimate with the rate of change of spatially correlated nodes.

In summary, a large number of methods involved linear interpolation as a baseline [2], [3], [5], [13], cubic interpolation [10], [14], multiple regression spatially or temporally [2], [11]–[13], and often compared with k-Nearest Neighbour (kNN) which is often very similar to or referred to as Naive Spatial Estimation (NSE) [3], [5], [6]. This group of methods forms the basis for our comparisons.

III. Missing Data Imputation Methods

We consider that the system synchronously advances in discrete time steps, and that we are limited to a "horizon" of m recent time steps (m is subsequently refined/modified as needed). In our evaluation section, we consider multiple options for mapping the time step to natural time, but the mapping does not impact the definition of the methods we describe here. The data are sent once per time step/slot as long as the node is not in failed state.

We follow closely the notation conventions of [13], where many of the presented techniques were systematically presented. We can view the data in the following form:

$$\begin{bmatrix} V_{11} & V_{12} & V_{13} & \dots & V_{1n} \\ V_{21} & V_{22} & V_{23} & \dots & V_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ V_{m1} & V_{m2} & V_{m3} & \dots & V_{mn} \end{bmatrix}$$

where V_{ki} corresponds to the k^{th} data value for node a_i. For the sake of presentation, it is convenient to think of V_{ki} as scalars, but this need not be the case in general. The time T_k corresponds to the timestamp for which the data $V_{ki}(\forall i)$ occurred.

1) Linear Interpolation (LIN) or Temporal Correlation Algorithm - Linear Interpolation (TCA-LI): This is the most simplistic method as it only requires the available data before and after a gap of missing data, and performs linear interpolation to recover the missing data, or more formally:

$$\hat{V}_{ki} = V_{ui} + \frac{T_k - T_u}{T_v - T_u}(V_{vi} - V_{ui})$$

Where, $\hat{V}_{ki}$ is the estimated value and $T_u < T_k < T_v$. The data from T_u and T_v should be the closest available data to the missing data at time T_k. If we express the computation required for computing the missing data over the horizon of m time steps, the computational complexity of this method is $O(m)$ due to the search for the values for V_{vi} and V_{ui}.

2) Temporal Correlation Algorithm - Multiple Regression (TCA-MR): This method uses multiple (per-node) linear regression to find correlations in the temporal aspect of the data. The weights determine how strongly the recent data effects the next value versus the data more distant in the past.

The outlines for this method can be seen in [13]. TCA-MR has a time complexity of $O(l^3)$, where l is the number of recent data points that effect the next value.

3) Spatial Correlation Algorithm (SCA): An alternative is to describe the correlations in the spatial aspect of the data . Here, the weights determine how strongly the other node data is correlated to the data from the missing node.

As with TCA-MR, the outlines for this method can be seen in [13]. SCA has a time complexity of $O(r^3)$, where r is the number of nodes used to find the data of the missing node.

4) NSE or k-NN: Typically, the Nearest Neighbour (k-NN) approach is described as using an estimation based on the values of the k nearest nodes, where distance is determined by the Pearson correlation. Other methods, such as Naive Spatial Estimation (NSE), use a weighted summation of neighbouring node estimations where the weights are assigned by correlation coefficients over the total sum of all correlations. The method used here is similar to the latter with one small modification to account for the discrepancies in the node types. That is, we first normalize each of the nodes data sequences:

$$\hat{V}_{ki} = \sum_{q=1}^{r} \frac{\rho_{iq}}{C_i} * \frac{V_{kq} * \mu_i}{\mu_q}$$

Where: $\rho_{kq} = \dfrac{E[(V_k - \mu_k)(V_q - \mu_q)]}{\sigma_k \sigma_q}, C_i = \sum_{q=1}^{r} \rho_{iq}$

Here, μ_x and σ_x are the mean and standard deviation values respectively of node a_x over the temporal sequence length (i.e. the previous h data values). C_i is the sum of each of the correlation values of all nodes $\{a_1, a_2, \dots, a_r\}$ with node a_i.

As with SCA, a subset of all nodes is used which only include those nodes that contain a value at the time slot T_k in which node a_i has missing data. The complexity of this algorithm is $O(rh)$.

Notice that TCA-MR, SCA, and NSE share in common the requirement of a horizon of past values to a length of h time steps in order to capture the influence of recent history on the to-be-estimated data. The choice of h (and l for TCA-MR) is discussed in Section VI.

5) Cubic Interpolation: To address whether a more elaborate curve interpolation could provide benefits over linear interpolation, we include a cubic interpolation scheme. The cubic interpolation takes the two previous data points before the "gap" and the next two data points after the gap, and interpolates for the missing values. Since the matrix inversion performed in the interpolation is constant (fixed size matrices) the algorithm complexity here is $O(m)$, i.e., similar to that of linear interpolation.

IV. Models

A. A Markovian Node Failure Model

A Gilbert-Elliot (GE) Markovian model is considered representing the independent state of each node: $a_i = 1$, communicating or $a_i = 0$, non-communicating (failed). Transitions are used to represent changes from a slot in which the nodes is communicating to non-communicating in the next slot (and vice versa), while self-transitions express sojourn in the same state. Specifically:

$$\begin{cases} P(a_i(t+1) = 0 | a_i(t) = 0) = (1 - P_{[rec]}) \\ P(a_i(t+1) = 1 | a_i(t) = 0) = P_{[rec]} \\ P(a_i(t+1) = 0 | a_i(t) = 1) = P_{[fail]} \\ P(a_i(t+1) = 1 | a_i(t) = 1) = (1 - P_{[fail]}) \end{cases}$$

The collective state of the system S can be expressed as the Cartesian product of the states of individual nodes, i.e., by a total of 2^n states where n is the number of sensor nodes. The transition probabilities are used as parameters in the evaluation of the schemes. In order to alter the number of missing data sequences, the node failure probability, $P_{[fail]}$, is increased or decreased. Likewise, to adjust the duration for which the node remains in the failure state, the recovery probability, $P_{[rec]}$, can be changed. A higher $P_{[fail]}$ results in more frequent node failures and a lower $P_{[rec]}$ results in longer durations of node failures.

B. A Node Storage Model

If the node is able to continue sensing while unable to communicate, a form of storage management is needed if the outage of communication lasts for a long time. We proceed in a manner similar to that described by Raza et al. [4]. In some scenarios, small collections of data can be gathered and, rather than transmit each data value separately, send messages consisting of a set of <timestamp,value> pairs. After a failure is restored, the stored data can be sent, i.e., recovered, providing the potential to improve over previous estimations (due to the gaps) and providing a new basis for further estimations (if still needed).

To our best knowledge, previous works using data imputation methods for missing data estimation have not accounted for partial missing data recovery from the nodes. Here, we assume a small per-node storage capacity allows some part of the missing date to be recovered and we demonstrate that this can result in drastic improvement of the quality of the imputation. A first, basic, partial data recovery scheme is a First-In-First-Out (FIFO) queue, storing only the most recently sensed data, to be collected as soon as communication in restored. Under prolonged failure periods, the fact that FIFO drops the older data results in potentially poor performance. For this reason, we also consider a sub-sampling strategy within the same space restrictions as that of FIFO. This sub-sampling takes place incrementally, replacing two of the oldest samples with a single value, i.e., effectively freeing up one storage slot for the more recent data.

It is important to note that a node cannot determine in advance the length of time it will stay in failed state. Therefore, it is possible that the sub-sampling will have to take place a number of times, and that eventually we will sub-sample already sub-sampled data, and do this possibly repeatedly. The gist of why sub-sampling can be powerful is based on the observation that, with sub-sampling, the lower frequency components that underlie shifting trends in the data are retained across the time span of the failure duration.

V. Evaluation

A. Dataset

The data used for this study was collected from a multi-unit apartment complex consisting of eleven apartments across four floors. A sensor in each apartment was collecting temperature (°C), CO_2 levels (ppm), and relative humidity (%). Some pre-processing was necessary to adopt them to the discretized synchronous assumptions made here. We set as the minimum time step/slot to be equal to 60 seconds. All data collection intervals studied are multiples of 60 seconds. Due to the lack of synchronization across all nodes, the exact natural time when each node was sampling was not directly controlled. Over a 60 second period, each node would approximately sample 4 times. The value closest to the beginning of the next time slot was considered the measurement for that time slot. When no measurement was reported in that time slot, the node was considered failed (as in failed node in the source data).

B. Recovery for Re-estimation

We restricted the node data storage to five <timestamp,value> pairs; applying data sub-sampling once we exceed this capacity. The data estimation simulation runs through each data value in sequential order, starting at time T_0 from Value V_{00} until V_{0k} and then repeating for T_1 and so on. When a point is reached where V_{ki} is missing, the value is estimated using one of the methods in Section III. When a node reaches a point where the data is no longer missing but the previous l values were missing, it indicates that a node has recovered from a failed state. At that point, the simulation will place up to p (in our case five) exact values recovered from the node. When no buffering is simulated, $p = 0$. If FIFO buffering is simulated, then the $p = 5$ most recent sensed values at that node are placed leaving the $l - p$ earlier entries vacant. If the sub-sampling behavior is simulated, the p values are spread across the l vacancies as per the timestamps of those p remaining values. The remaining $l - p$ missing values will be re-estimated using the same method as before (before the failed node had recovered) but will use the p true values that were before assumed missing.

C. Metrics

The metric used to evaluate the estimation methods is the Root Mean Square Error (RMSE).

For the sake of presentation, any method with an average RMSE greater than 10 for the Temperature dataset was considered unreasonable and removed from plotting to ensure results from other methods could be easily compared.

VI. Results

The analysis of the estimation methods was done over 3 parameters; namely, the time step/slots, the probability that a node will recover in the next interval if currently in the failed state, and the data recovery sampling method and amount.

Initial testing was done to determine the values of the controlled parameters which include the failure probability, $P_{[fail]}$, the length of the time series for TCA-MR, SCA, and NSE methods (h), and the start and end times of our datasets.

For the controlled failure probability, it was important that the value was large enough that there were enough failure sequences to give meaningful errors, but also not too large such that the number of overlaps of node failures were not too frequent, resulting in estimations largely influencing other estimations. It was determined that $P_{[fail]} = 10^{-4}$ and $P_{[rec]} = 10^{-2}$ was sufficient in meeting these demands for time intervals of 600 seconds, and it formed a baseline for all comparisons. When sub-sampling was used, a total of five values could be stored at each node.

The length of the time series for estimating using the TCA-MR, SCA, and NSE methods determined how many intervals prior to the missing data value were used to perform the estimation. This is represented as the h value in the description of the methods. After some testing, using the value of 100 (and using a value of 10 for the l parameter in TCA-MR) fit within these constraints. It should be noted that, under different time intervals, h and l should ideally be calibrated.

The data set used was not complete during the entire lifetime of the data collection process, some work was carried out to determine a region of data where there were at least h continuous and complete data points, followed by a large number of data that contained no more than 30 minutes of consecutive failures of all nodes, ending with at least 2 consecutive time intervals with complete data for all nodes. We ended up using the data collected from 3:20AM May 25, 2016

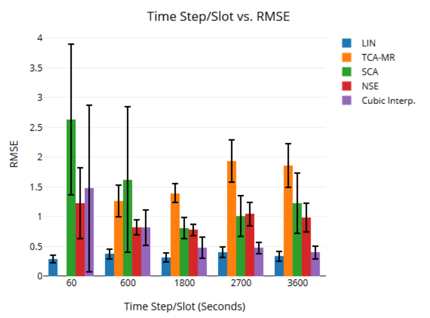

Fig. 1. Method comparison under different time steps/slots.

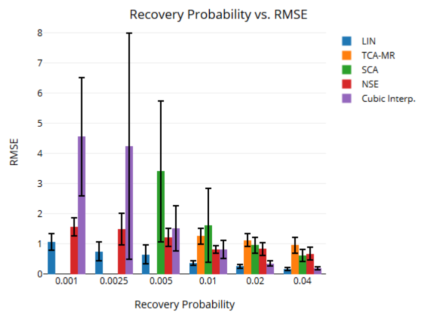

Fig. 2. Method comparison under different recovery probabilities ($P_{[rec]}$).

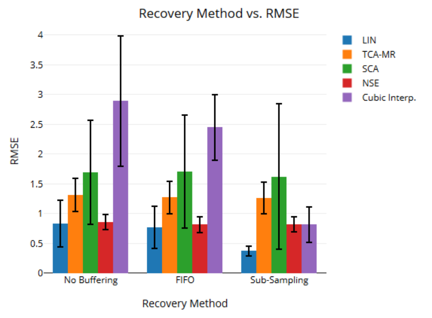

Fig. 3. Method comparison when (i) no data was recovered, (ii) only the most recent five data points were recovered (FIFO), and (iii) where five points were recovered and spaced using sub-sampling.

until 10:02AM August 17, 2016. At a 60 second granularity, it is 121362 minutes of data from each of the 11 nodes. Of the total 1334982 data values, just 14110 ($\sim 1\%$) are missing due to gaps in the data set.

A. Time Step/Slot

In order to only change the time steps/slots, leaving the average duration of the failure intervals the same, the failure and recovery probabilities are scaled by the same factor as the time steps. This means that at a time step of 60 seconds, the failure probability and recovery probability would be 10^{-5} and 10^{-3} respectively. But if the time step is increased 10-fold to 600 seconds, the new probabilities would be 10^{-4} and 10^{-2} respectively. For hour granularity or 3600 seconds, the fail and recovery probability are then $6 \cdot 10^{-4}$ and $6 \cdot 10^{-2}$.

The results of changing the time interval between subsequent data values can be seen in Figure 1. This shows that the only significant effect on any method is the decrease in error by the Cubic Interpolation method and the increase in the TCA-MR method's error. The decrease in the Cubic interpolation error could be attributed to the decrease in the lengths of the failure duration, while the increase in the TCA-MR could be a result of the more intense fluctuations in the previous h data points, resulting in a poorly fit linear equation. The static nature of the other methods is likely caused by the use of the recovered data which will be verified in the latter sections.

B. Recovery Probability

As seen in Figure 2, when the failure duration increases due to the decrease in recovery probability, TCA-MR is the first to deteriorate beyond a reasonable error, then SCA. Cubic interpolation can have relatively poor errors at recovery probabilities of 0.0025 and below. LIN errors remain low due to the five buffered, recovered, values which ensure that values do not deviate too far from the data trend. NSE also remains consistent, but with larger errors than LIN.

C. Node Recovery Amount and Method

Figure 3 shows the resulting error when the time interval is 600 seconds, $P_{[fail]} = 10^{-4}$ and $P_{[rec]} = 10^{-2}$. The first set of bars correspond to typical assumptions that nodes cannot store any data. The second set of points show the errors when a node is able to buffer five data points but only retains the most recent ones (FIFO scheme). The third set of points utilizes data sub-sampling of the five buffered data points, which appears to result in an increase of the information gained.

It is evident that recovering only the most recent data points results in no significant reduction in error. However, spreading the data out using sub-sampling decreases the error in the LIN method and cubic interpolation. This is because the additional recovered information ensures the LIN and Cubic methods follow the trend of the data more closely.

VII. Discussion

Our results suggest that "simple is beautiful." The LIN method has the lowest errors across all parameter sets. The strongest reason for this stems from the synergy of LIN and the use of recovered data which decreases the error significantly by guiding estimations to closely follow the true data trend. This is achievable with a surprisingly small number (in our simulations equal to five) of recovered, and possibly sub-sampled, values.

LIN and NSE appear to have the most consistent errors across all the time intervals and failure durations. Due to the nature of these two methods, it is not possible for the values to exceed the maximum and minimum values of the dataset and therefore the possible estimation values, and therefore errors, are bounded. Conversely, SCA, TCA-MR, and cubic interpolation can be susceptible to fixating on short term trends that may not accurately represent the broader trends in the data. This can result in the values increasing beyond the true range of the data and, with a node failure duration of significant length, can result in massive errors.

For the case of cubic interpolation, if there was a brief moment of large increase or decrease in the values before or after the node failure, the errors over that failure run would

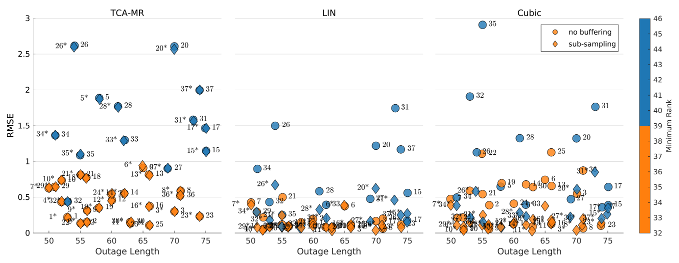

Fig. 4. Method comparison for recovery method with minimum rank in context. Each point plotted corresponds to an outage – outages are enumerated, circles are RMSE without buffering, diamonds (and star next to the number) are RMSE with buffering and sub-sampling; color indicates the range of the minimum rank.

likely be significantly high. Because cubic interpolation fits precisely to 4 points (2 prior to failure and 2 post failure), it is very sensitive to any slight deviations in their values, more prominent in the case of large intervals. Using five recovered data points helps by shortening a single interval into at most four smaller sub–intervals, reducing the sensitivity.

A. A Low-Rank Approximation Perspective

Next, we provide evidence linking the potential for an imputation technique to perform well based on whether, in the interval just prior to an outage, the reconstructed signal could be represented in a "sparse" form. We adopt a matrix rank view to express sparsity. Specifically, we notice the link between minimum matrix rank, and the RMSE. At the same time we keep the duration of the outage interval in context, because longer outages are expected to result in higher RMSE. Results assuming no buffering at the node and with buffering (and hence sub-sampling) at the nodes are shown.

We hypothesize that there exists a relation between the information content just prior to an outage and the RMSE of the imputation as applied to that outage's values. For an outage of length l between T_k and T_{k+l}, the information content is operationalized by the minimum rank of $\mathbf{M} = \mathcal{H}(V_{(k-t)i}, ..., V_{(k-2)i}, V_{(k-1)i})$, where $\mathcal{H}$ is the Hankel operator producing the matrix form from a time series of length t (the values prior to the outage). The minimum rank is obtained by using truncated singular value decomposition (SVD) w.r.t. a certain reconstruction error e as an approach for low-rank approximation. For $\mathbf{M} = \mathbf{USV^T}$ we look at the singular values diag$(\mathbf{S})$. A useful property of SVD is that the singular values coincide with the error norm for all possible low-rank approximations, $\mathbf{S}_{k,k}$ is equal to the norm of difference between the approximating matrix $\hat{\mathbf{M}}_k$ with rank k and the original data matrix $\mathbf{M}$ [17]. We minimize k subject to $\sum_{i=k+1}^{m} \mathbf{S}_{i,i} \leq e \sum_{j=1}^{m} \mathbf{S}_{j,j}$, where m corresponds to the number of rows in $\mathbf{S}$ and e is fixed. The approximating matrix $\hat{\mathbf{M}}_k$ is obtained by simply setting all singular values $\mathbf{S}_{k',k'}$ with $k' > k$ to zero.

Figure 4 shows the imputation RMSE for several outages under different conditions, each marker represents one outage. The outages that are present in the figure were obtained by simulations with the time interval of 600 seconds, the fail probability of 0.0001 and the recovery probability of 0.02 based on the temperature dataset. For the sake of clarity only outages with length between 50 and 75 are shown. The low rank approximation is produced for $t = 100$ and with $e = 0.1\%$. The resulting minimum rank k values are color coded depending on the range they belong to ("high" as blue, "low" as orange – both ranges are of equal length). Note that the minimum rank is independent of the imputation method.

In TCA-MR we clearly see a distinct low rank cluster and a high rank cluster of markers. Outages with corresponding lower minimum rank tend to have lower RMSE for the estimation of missing values. In the cases of LIN and Cubic methods we observe a similar distribution of the markers, though it is less distinct. Another factor which comes into play is the recovery method, *sub-sampling* versus *no buffering*. As shown in Figure 3, LIN and Cubic benefit a lot from sub-sampling, in contrast to TCA-MR which is hardly affected by a change of the recovery method as circles and diamond markers of the same outage coincide almost always. A closer look at the subplot for LIN reveals that the overlap of the lower rank cluster and the higher rank cluster is actually smaller within the markers for a specific recovery method. This is an interesting property to exhibit, because even though LIN is using only one value prior to the first missing value of an outage interval and the next (first after the outage interval) non-missing value, the resulting RMSE still appears influenced by the prior (pre-outage) minimum rank. The minimum rank is an indicator for the level of difficulty in estimating missing values independent of the chosen methods, at least for outage intervals of comparable length to the "window" of time t over which the low rank is determined.

VIII. Conclusion & Future Work

Wireless sensors can be unreliable due to many causes such as damage, tampering, etc. [1]. It is therefore imperative to have some form of data recovery to maintain real-time or retrospective analysis and actuation, while ensuring the correct and uninhibited operation of applications [2]. We evaluated a small collection of existing, simple, methods under a Markovian failure model, allowing the probabilities of the node outage and between-outage times to be controlled.

This research was conducted based on an assumption that nodes can store a small amount of data when unable to transmit to the sink node. The data recovery was used to re-estimate past predictions to reduce the error from previous estimations. A simple sub-sampling algorithm in limited storage was considered which uniformly spaces recovered data during an outage, resulting in improved imputation.

Although this study showed that, in most cases, LIN is a good choice, as long as we are dealing with short outages and sub-sampling based recovery, future work will examine more data sets as well as ways to simultaneously exploit the sparse representation of the sensed data, together with local decisions at each node as to which samples to retain and which ones to drop, beyond simple sub-sampling.

References

[1] L. Paradis and Q. Han. A survey of fault management in wireless sensor networks. *Journal of Network and Systems Management*, 15(2):171–190, 2007.

[2] L. Pan, H. Gao, H. Gao, and Y. Liu. A spatial correlation based adaptive missing data estimation algorithm in wireless sensor networks. *International Journal of Wireless Information Networks*, 21(4):280–289, 2014.

[3] R. Kumar, D. Chaurasia, N. Chuahan, and N. Chand. Predicting missing values in wireless sensor network using spatial-temporal correlation. *IRACST – International Journal of Computer Networks and Wireless Communications (IJCNWC)*, 7(3):20–25, 2017.

[4] U. Raza, A. Camerra, A. L. Murphy, T. Palpanas, and G. P. Picco. Practical data prediction for real-world wireless sensor networks. *IEEE Transactions on Knowledge and Data Engineering*, 27(8):2231–2244, 2015.

[5] L. Pan and J. Li. K-nearest neighbor based missing data estimation algorithm in wireless sensor networks. *Wireless Sensor Network*, 2:115–122, 2010.

[6] M. Luo, F. Wang, and X. Hu. Estimating missing values of wsn using modified frequent itemsets mining and nn search. pages 673–678. Proceedings of the 2015 4th International Conference on Computer, Mechatronics, Control and Electronic Engineering, 2015.

[7] C. Zhang, Y. Zhuang, and J. Li. A estimation model of missing value based on wireless sensor network. *DEStech Transactions on Engineering and Technology Research*, 2016.

[8] L. Z. Wong, H. Chen, S. Lin, and D. C. Chen. Imputing missing values in sensor networks using sparse data representations. pages 227–230. Proceedings of the 17th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems - MSWiM '14, 2014.

[9] J. Kan, R. Zhang, and L. Chen. A missing values imputation algorithm in wireless sensor networks. 2014 ASABE – CSBE/SCGAB Annual International Meeting Paper, 2014.

[10] N. Yan, M. Z. Zhou, and L. Tong. An estimation algorithm for missingdata in wireless sensor networks. *International Journal on Smart Sensing and Intelligent Systems*, 6(3):1032–1053, 2013.

[11] H. Zhang and L. Yang. An improved algorithm for missing data in wireless sensor networks. pages 346–350. International Conference on Software Intelligence Technologies and Applications & International Conference on Frontiers of Internet of Things 2014, 2014.

[12] Q. Zhen and T. Zhang. A missing data estimation algorithm in wireless sensor networks. *Boletín Técnico*, 55(3):212–217, 2017.

[13] X. Ren, H. Sug, and H. Lee. A new estimation model for wireless sensor networks based on the spatial-temporal correlation analysis. *Journal of Information and Communication Convergence Engineering*, 13(2):105–112, 2015.

[14] L. Pan, H. Gao, J. Li, H. Gao, and X. Guo. Ciam: An adaptive 2-in-1 missing data estimation algorithm in wireless sensor networks. 19th IEEE International Conference on Networks (ICON), 2013.

[15] Z. Gao, W. Cheng, X. Qiu, and L. Meng. A missing sensor data estimation algorithm based on temporal and spatial correlation. *International Journal of Distributed Sensor Networks*, 11(10), 2016.

[16] Y. Li and L. E. Parker. Classification with data in a wireless sensor network. pages 533–538. IEEE SoutheastCon 2008, 2008.

[17] I. Markovsky. *Low Rank Approximation: Algorithms, Implementation, Applications*. Springer Publishing Company, Incorporated, 2011.