Real-time Synchrophasor Data Quality Analysis and Improvement

Final Project Report

S-71

Power Systems Engineering Research Center
Empowering Minds to Engineer
the Future Electric Energy System

Real-time Synchrophasor Data Quality Analysis and Improvement

Final Project Report

Project Team
Le Xie, Project Leader
P. R. Kumar
Texas A&M University

Mani V. Venkatasubramanian
Washington State University

Graduate Students
Meng Wu
Dongqi Wu
Texas A&M University

Yang Zheng
Mohammadreza Maddipour Farrokhifard
Washington State University

PSERC Publication 18-11

September 2018

For information about this project, contact:

Le Xie
Texas A&M University
Department of Electrical and Computer Engineering
301H Wisenbaker Engineering Building
College Station, TX 77843
Phone: 979-845-7563
Email: le.xie@tamu.edu

Power Systems Engineering Research Center

The Power Systems Engineering Research Center (PSERC) is a multi-university Center conducting research on challenges facing the electric power industry and educating the next generation of power engineers. More information about PSERC can be found at the Center's website: http://www.pserc.org.

For additional information, contact:

Power Systems Engineering Research Center
Arizona State University
527 Engineering Research Center
Tempe, Arizona 85287-5706
Phone: 480-965-1643
Fax: 480-727-2052

Notice Concerning Copyright Material

PSERC members are given permission to copy without fee all or part of this publication for internal use if appropriate attribution is given to this document as the source material. This report is available for downloading from the PSERC website.

© 2018 Texas A&M University. All rights reserved.

Acknowledgements

The work described in this report was sponsored by the Power Systems Engineering Research Center (PSERC).

The authors thank industry collaborators including Xiaoming Feng (ABB), Gurudatha Pai (GE), Vijay Sukhavasi (GE), Prashant Kansal (AEP), Giuseppe Stanciulescu (BC Hydro), Aftab Alam (California ISO), Alan Engelmann (ComEd), Floyd Galvan (Entergy), Jay Ramamurthy (Entergy), Mahendra Patel (EPRI), Evangelos Farantatos (EPRI), Chaitanya A. Baone (GE), Santosh Veda (GE), Frankie Zhang (ISO New England), Liang Min (LLNL), Yingchen Zhang (NREL), Naim Logic (SRP), Jay Caspary (SPP), and Harvey Scribner (SPP).

Executive Summary

This project is motivated by the need to monitor and mitigate data quality issues in phasor measurement unit (PMU) data streams. As utilities and vendors develop more and more PMU–based decision making tools, there is an urgent need to develop scalable, real-time methods to monitor and improve PMU data quality. Unlike traditional supervisory control and data acquisition (SCADA) measurements, PMU data has much higher sampling rate as well as accuracy requirement. Conventional bad data detection algorithms are therefore rendered ineffective.

Built upon the recent success of dimensionality reduction of PMU data together with the vast experience gained in handling real-time data quality problems for real-time application development, the project team proposes a two-thrust approach to monitoring and correcting PMU data quality. First, we develop an online data-driven algorithm that is well suited bad data detection during both normal and eventful conditions. Second, we investigate the root causes of the data quality issues by different mechanisms such as GPS clock issues and communication network delays, and develop mitigation strategies for "fixing" or "filling" the problems utilizing advanced signal processing tools and dynamic state estimation methods.

Part I: Online Detection of Low-Quality PMU Measurements: A Data-Driven Approach
Part I of the report presents a data-driven approach for online detection of low-quality PMU measurements. It leverages the spatio-temporal similarities among multi-time-instant PMU data, and applies density-based local outlier detection technique to detect low-quality PMU measurements. The major advantages of the proposed approach are summarized as follows. (1) This is a purely data-driven approach, without requiring any prior knowledge on system topology or model parameters, which eliminates the potential misdetections caused by inaccurate system information; (2) the proposed approach can operate without any converged state estimation results and is suitable for filtering out gross measurement errors for advanced power system analytics; (3) the proposed approach has fast computational speed, which could be beneficial for real-time applications; and (4) the algorithm is able to perform detections under both normal and fault-on operating conditions. The proposed detection algorithm differentiates high-quality PMU data recorded during system physical disturbances (faults) from the low-quality data, which avoids potential false alarms caused by physical disturbances. The proposed detection algorithm has been implemented as a desktop application to analyze offline data.

Part II: Multi-signal Analytical Guesstimation using Intelligent Collaboration (MAGIC) for synchrophasor data
Part II of the report presents algorithms for reconstruction of missing PMU data by using available measurements from neighboring locations. As noted in Part I of the report, PMU data may be unavailable or unusable for a variety of reasons such as from measurement errors or loss of data during communication. Traditional methods such as those based on linear interpolation are inaccurate and they do not preserve the dynamic features of the system response if any of the data is lost during system events. This report introduces a data-driven approach that uses available measurements from nearby signals and from data available from recent past as cross-references to obtain accurate reconstructions of the missing data. a) The proposed technique provides highly accurate reconstructions in preserving system modal information and the results match well with those from actual values in test studies. b) The proposed method is computationally fast and can

be implemented in real-time for correcting data that could be used in subsequent PMU applications. c) The proposed method can reconstruct data over long time-intervals as long as 20 seconds provided suitable measurements are available from its neighbors. d) The reconstructed signals can be used for improving the accuracy of PMU based applications such as dynamic state estimation, oscillation monitoring and model validation.

Project Publications:

[1] Meng Wu, and Le Xie, "Online Detection of Low-Quality Synchrophasor Measurements: A Data-Driven Approach," *IEEE Transactions on Power Systems*, vol. 32, no. 4, pp. 2817-2827, Jul. 2017.

[2] Meng Wu, and Le Xie, "Online Detection of False Data Injection Attacks to Synchrophasor Measurements: A Data-Driven Approach", *2017 50th Hawaii International Conference on System Sciences (HICSS)*, Hawaii, US, Jan. 2017.

[3] Meng Wu, and Le Xie, "Online Identification of Bad Synchrophasor Measurements via Spatio-Temporal Correlations", *2016 Power Systems Computation Conference (PSCC)*, Genoa, Italy, Jun. 2016.

[4] V. Venkatasubramanian, Y. Zheng and M. M. Farrokhifard, "Systems and Devices for Automatic Data Recovery in Synchrophasor Data Streams and Associated Methods", US patent application, May 2018.

Student Theses:

[1] Meng Wu. *Physics-Based and Data-Driven Analytics for Enhanced Planning and Operations in Power Systems with Deep Renewable Penetration*. PhD Dissertation, Texas A&M University, December 2017.

[2] Yang Zheng. *Cross-referencing Reconstruction of Missing PMU Data using Collaborative Filtering*, M.S. Thesis, July 2017.

Software Application:

[1] D. Wu, "Desktop Bad-Data Detection Application", https://github.tamu.edu/dqwu/LOF-Bad-Data-Detection, January 2018

Part I

Online Detection of Low-Quality PMU Measurements: A Data-Driven Approach

Le Xie
P. R. Kumar
Meng Wu, Graduate Student
Dongqi Wu, Graduate Student

Texas A&M University

For information about this project, contact

Le Xie
Texas A&M University
Department of Electrical and Computer Engineering
301H Wisenbaker Engineering Building
College Station, TX 77843
Phone: 979-845-7563
Email: le.xie@tamu.edu

Power Systems Engineering Research Center

The Power Systems Engineering Research Center (PSERC) is a multi-university Center conducting research on challenges facing the electric power industry and educating the next generation of power engineers. More information about PSERC can be found at the Center's website: http://www.pserc.org.

For additional information, contact:

Power Systems Engineering Research Center
Arizona State University
527 Engineering Research Center
Tempe, Arizona 85287-5706
Phone: 480-965-1643
Fax: 480-727-2052

Notice Concerning Copyright Material

PSERC members are given permission to copy without fee all or part of this publication for internal use if appropriate attribution is given to this document as the source material. This report is available for downloading from the PSERC website.

© 2018 Texas A&M University. All rights reserved

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Background

In recent years, there has been significant deployment of phasor measurement units (PMU) around the world. Compared with traditional metering units in supervisory control and data acquisition (SCADA) systems, PMUs provide measurements with much higher sampling rates. The high-resolution PMU measurements contain rich information on system dynamics, which stimulates the development of advanced analytics, such as dynamic state estimation [1], PMU-based model validation [2], and wide-area control and protection [3,4]. However, as a large amount of data is streaming into the control center, the PMU data quality problem becomes one of the major challenges for system operators. Generally speaking, low-quality PMU data represents data that cannot accurately reflect the underlying system behavior. The inaccuracy can be caused by various problems such as sensing noises, data loss, and the global positioning system (GPS) time errors. As an example, the ratio of low-quality PMU data, reported by California Independent System Operator (ISO) in 2011, ranged from 10% to 17% [5]. In 2013, the ratio of low-quality PMU data in China was reported to range from 20% to 30% [6]. The online data quality monitoring of PMUs becomes a major barrier for any advanced PMU-based analytics.

1.2 Literature Survey

In order to improve data quality of PMU systems, various methods have been proposed. In [7], a PMU-based state estimator is introduced to detect phasor angle bias and current magnitude scaling problems. In [8], the Kalman filtering technique is applied to detect low-quality PMU data. Both state estimator and Kalman filter-based approaches require prior knowledge on system topology and model parameters for detecting low-quality data. Therefore, the detection accuracy of the above approaches may be affected when gross errors are presented in system topology or parameters. Furthermore, these methods cannot operate successfully when state estimation diverges because of gross measurement errors, system physical disturbances, or stressful operating conditions. In [9,10], several logic-based low-quality data detection schemes are presented. These approaches compare PMU data with certain thresholds, apply high-noise filters to raw PMU measurements, and perform cross-checking on PMU measurements obtained in nearby physical locations, in order to detect abnormal PMU measurements. However, these pre-defined logics may be rendered ineffective when large disturbances occur in the studied power grid. In [11], clustering algorithms are applied to extract information from power system time-varying data. These clustering techniques could potentially be applied to detect system anomalies such as low-quality PMU data or system physical disturbances. Reference [12,13] pioneered a purely data-driven method to improve PMU data quality. This method applies low-rank matrix factorization techniques to detect and repair low-quality PMU data. It has satisfactory performance under both normal and fault-on operating conditions. However, since the matrix factorization techniques bear high computational burden such as nonlinear optimizations, it becomes a challenge when applied for real-time applications.

1.3 Scope of Work

In view of the current efforts on PMU data quality improvement, this report presents a data-driven approach for online detection of low-quality PMU measurements. It leverages the spatio-temporal similarities among multi-time-instant PMU data, and applies density-based local outlier detection technique to detect low-quality PMU measurements. The major advantages of the proposed approach are summarized as follows. (1) This is a purely data-driven approach, without requiring any prior knowledge on system topology or model parameters, which eliminates the potential misdetections caused by inaccurate system information; (2) the proposed approach can operate without any converged state estimation results and is suitable for filtering out gross measurement errors for advanced power system analytics; (3) the proposed approach has fast computational speed, which could be beneficial for real-time applications; and (4) the algorithm is able to perform detections under both normal and fault-on operating conditions. The proposed detection algorithm differentiates high-quality PMU data recorded during system physical disturbances (faults) from the low-quality data, which avoids potential false alarms caused by physical disturbances.

1.4 Report Organization

The rest of the report is organized as follows. Section 2 presents the problem formulation of the low-quality PMU data detection issue; Section 3 discusses the proposed data-driven approach for low-quality PMU data detection; Section 4 presents case study results to verify the proposed approach; Section 5 introduces a software tool developed using the proposed method; Section 6 provides concluding remarks to this report.

2. Problem Formulation

This section presents the key features differentiating low-quality PMU measurements from the high-quality ones. Based on these features, low-quality PMU measurements are formulated as *spatio-temporal outliers* among high-quality measurements in the power grid. Accordingly, the low-quality PMU data detection problem is formulated to be a spatio-temporal outlier detection problem.

2.1 Problem Formulation

Let $m \times n$ matrix M denote a set of PMU measurements collected from n PMU channels of the same type (i.e., all of them are voltage/current/power channels), within m time instants. This measurement matrix can be decomposed into the following two matrices:

$$M = L + D \tag{2.1}$$

where the k^{th} column of matrix L represents the accurate measurements corresponding to the k^{th} PMU channel in M, and D denotes the matrix containing inaccurate information caused by data quality problems. Each nonzero entry D_{ij} represents a measurement error of the j^{th} PMU channel at time instant i. Here, a PMU channel represents one of the following electrical quantities obtained by a PMU: voltage magnitude, voltage phasor angle, current magnitude, current phasor angle, real power, and reactive power. Therefore, M_{ij} is a real number instead of a complex number.

Definition 1. M_{ij} is defined to be low-quality PMU data if its corresponding $|D_{ij}| > \tau$, where τ is a positive threshold to determine low-quality data.

It has been shown in [12,13], when low-quality PMU data is presented in certain power system, the rank of matrix M would be higher than the rank of matrix L, due to the nonzero entries in matrix D. This phenomenon indicates the linear dependency (similarity) among PMU measurements would be weakened by data quality problems.

In order to demonstrate the above property of low-quality PMU measurements, Figure 2.1 shows voltage magnitude curves measured by two PMUs with nearby physical locations. Both curves were recorded at the same time period, when a line-tripping fault was presented in the system (from 3s to 5s). The upper curve contains low-quality data at around 1s. By observing only the upper curve, it is difficult to confirm whether the data spikes are caused by physical disturbance or data-quality problem, since all the data spikes have outlier behavior compared with their temporal neighbors. However, by comparing multiple PMU curves obtained in different locations of the system, it would be possible to differentiate spikes caused by data-quality problems and those caused by disturbances, since spikes caused by data-quality problems are outliers compared with their spatial neighbors, while spikes caused by disturbances appear in curves recorded by multiple PMUs and therefore cannot be considered as outliers compared with their spatial neighbors.

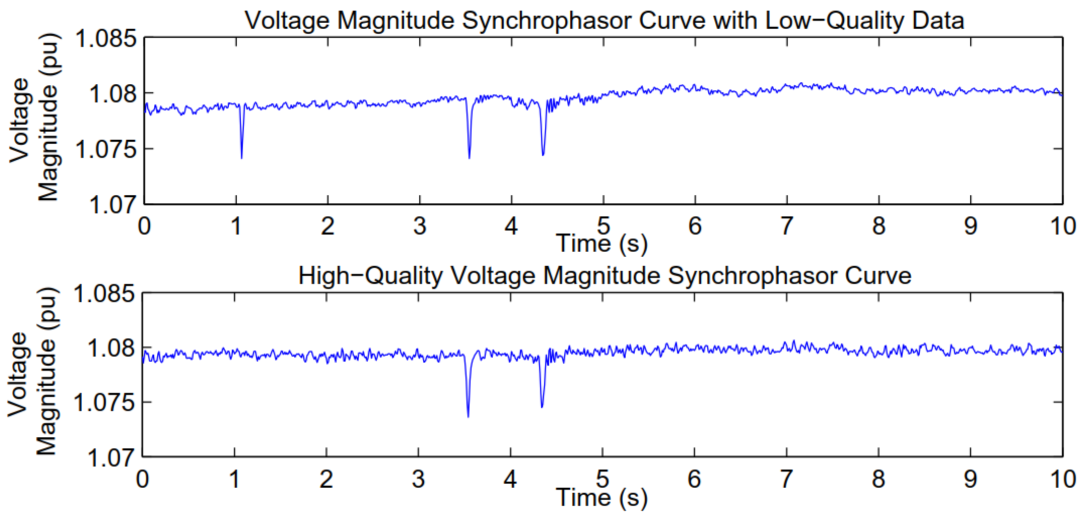


Figure 2.1 Comparison between PMU curves with and without low-quality data.

The above observations can be summarized as the following key features of low-quality and high-quality PMU data under normal/fault-on operating conditions:

Feature 1. Both low-quality PMU measurements and fault-on PMU measurements exhibit weak temporal similarities with the measurements obtained at the neighboring time periods, while high-quality PMU measurements obtained during normal operating conditions exhibit strong temporal similarities with the measurements obtained at the neighboring time periods.

Feature 2. Low-quality PMU measurements exhibit weak spatial similarities with the measurements obtained by the neighboring PMUs at the same time period, while fault-on PMU measurements exhibit strong spatial similarities with the measurements obtained by the neighboring PMUs at the same time period.

It should be noted that strong electrical connections among neighboring PMUs are required in order for the above features to be valid. Therefore, higher PMU measurement redundancy would lead to better accuracy in low-quality data detection, and lack of measurement redundancy could cause miss detections for the proposed algorithm. As more and more PMUs are being installed in power grids around the world, the measurement redundancy would be enhanced, and therefore the detection accuracy of the proposed algorithm would be improved.

2.2 Formulation of Low-Quality PMU Data as Spatio-Temporal Outliers

According to the discussions in the previous section, low-quality PMU measurements have weaker spatio-temporal similarities with their high-quality neighbors, under both normal and fault-on operating conditions. Therefore, these low-quality measurements can be formulated as spatio-temporal outliers among all the PMU measurements in the system. With a proper definition of similarity metrics for PMU curves, the degree of similarity between two PMU curves can be quantified, and data-mining techniques can be applied to detect the spatio-temporal outliers whose degrees of similarity are significantly different from other PMU curves.

For a measurement matrix M obtained within a certain period of time, general steps to formulate the detection problem are described as follows:

Step 1: Define a proper similarity metric (distance function) $f(M_i,M_j)$ which quantifies the degree of similarity between the i^{th} and j^{th} column of \$M\$.

Step 2: Map each column of M (a data curve obtained from certain PMU channel) to the space S where the distance function $f(M_i,M_j)$ is defined. Each column of M can be represented as a point in S.

Step 3: Examine the outlier behavior of the points in S, according to distance function $f(M_i,M_j)$. Points lying far from the majority are more likely to be outliers with low-quality data.

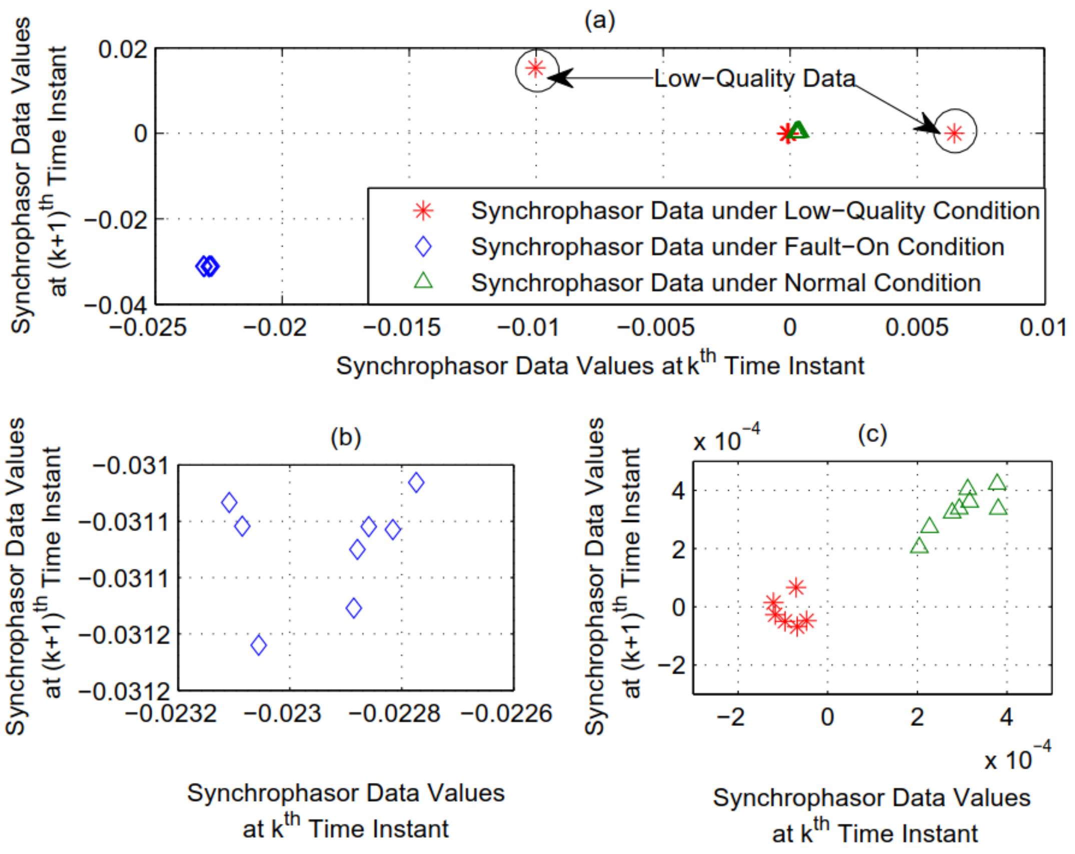


Figure 2.2 2D points representing PMU curves under normal/fault-on/low-quality conditions. (a) Overall figure with all the 2D points under normal/fault-on/low-quality conditions. (b) Zoomed-in figure with all the 2D points under fault-on condition. (c) Zoomed-in figure with all the 2D points under normal condition and high-quality 2D points under low-quality condition. [16]

Figure 2.2 demonstrates the above formulation through a simple example. Three 2×8 measurement matrices $M(1)$, $M(2)$, $M(3)$ are sampled from the same set of PMU channels at three different time periods. Each matrix contains 8 PMU curves within 2 consecutive time instants. $M(1)$ contains 6 high-quality PMU curves and 2 low-quality PMU curves obtained under normal operating condition. $M(2)$ and $M(3)$ contain 8 high-quality PMU curves obtained under fault-on and normal operating conditions, respectively. The Euclidean distance is used as the similarity metric (distance function), and each PMU curve in the three matrices is projected to the 2D Euclidean space shown in Figure 2.2. The x and y coordinates of each point are the data values at the first and second time instant of the corresponding PMU curve, respectively.

The following observations can be drawn from Figure 2.2: (1) The cluster of fault-on PMU data (fault-on cluster) lies far from the clusters of high-quality PMU data under normal operating condition (normal-condition cluster), indicating weak temporal similarity between the two clusters; (2) all the points within the fault-on cluster lie close to each other, indicating strong spatial similarities among points within the fault-on cluster; and (3) the two points representing low-quality PMU curves lie far from the normal-condition cluster, as well as the majority of points in the low-quality cluster, indicating weak spatial and temporal similarities with their neighboring points. Therefore, the low-quality data points can be defined as spatio-temporal outliers under this formulation.

3. Online Detection of Low-Quality PMU Data

Based on the previous discussion, we propose a density-based local outlier factor (LOF) analysis to detect low-quality PMU data. In [14], similar LOF-based techniques are introduced for the detection of high sensing noises and false data injections in PMU data. This section improves the similarity metrics for PMU curves, which lead to more robust performance on detecting various types of data quality problems, including not only sensing noises and false data injections, but also data spikes and un-updated data problems.

3.1 Similarity Metrics Between PMU Curves

In this subsection, two similarity metrics are proposed for detecting low-quality PMU data whose variance is significantly higher or lower than its spatio-temporal neighborhoods.

Definition 2. Let *M*(*k*) denote the PMU measurement matrix obtained at the k^{th} time period. The length of each time period equals to the length of the moving data window of the proposed algorithm. Let $M_i(k)$ and $M_j(k)$ denote the i^{th} and j^{th} columns of *M*(*k*).

Let $\sigma_i(k)$ denote the standard deviation of $M_i(k)$, Let *C* denote the data set of all the PMU measurements identified to be clean (without data quality problems) by the proposed algorithm. The normalized standard deviation for PMU data obtained from the i^{th} channel at the k^th time period is defined as follows:

$$\sigma_i^{Norm}(k) = \frac{\sigma_i(k)}{\frac{\Sigma_{t=1}^{t=k-1}\sigma_i(t)X_C\left(M_i(t)\right)}{\Sigma_{t=1}^{t=k-1}X_C\left(M_i(t)\right)}} \tag{3.1}$$

where

$$X_C\left(M_i(t)\right) = \begin{cases} 1 & (M_i(t) \in C) \\ 0 & (M_i(t) \notin C) \end{cases} \tag{3.2}$$

The normalized deviation $\sigma_i^{Norm}(k)$ represents the standard deviation of data curve obtained from the i^{th} PMU channel at the k^{th} time period, normalized by the average standard deviation of the historical clean measurements obtained from the same PMU channel. Considering $\sigma_i(k)$ as a indicator of the strength of system dynamic response recorded by i^{th} PMU channel at the k^{th} time period, $\sigma_i^{Norm}(k)$ is a normalized indicator which compares the current strength of system dynamic response with the average historical strength recorded by the same sensing channel. This normalization process removes the influence of PMU physical locations on the dynamic strength of the PMU curves.

3.1.1 Similarity Metric for Low-Quality PMU Data with High Variance

The similarity metric (distance function) $f_H(i,j)$ between $M_i(k)$ and $M_j(k)$ is defined as follows:

$$f_H(i,j) = \left|\sigma_i^{Norm} - \sigma_j^{Norm}\right| \tag{3.3}$$

3.1.2 Similarity Metric for Low-Quality PMU Data with Low Variance

The similarity metric (distance function) $f_L(i,j)$ between $M_i(k)$ and $M_j(k)$ is defined as follows:

$$f_L(i,j) = max\left(\left|\frac{\sigma_i^{Norm}}{\sigma_j^{Norm}}\right|, \left|\frac{\sigma_j^{Norm}}{\sigma_i^{Norm}}\right|\right) \tag{3.4}$$

The above two similarity metrics measure the difference between dynamic strength of data curves $M_i(k)$ and $M_j(k)$. Since during the same time period k, clean PMU curves across the system tend to have similar dynamic strength (similarly low/high strength under normal/fault-on operating condition), $f_H(i,j)$ and $f_L(i,j)$ values tend to be small for clean measurements. However, the dynamic strength of low-quality PMU curves tend to be different from that of the clean curves, since dynamics of low-quality PMU curves are mainly driven by the dynamics of the data quality problems, rather than the true system dynamics. Therefore, $f_H(i,j)$ and $f_L(i,j)$ values tend to be large for low-quality PMU measurements.

Although both similarity metrics could reflect the outlier behavior of both low-quality data with high variance (such as sensing noises, data spikes, etc.) and low variance (such as un-updated data), $f_H(i,j)$ tends to be more sensitive to high-variance data problems and $f_L(i,j)$ tends to be more sensitive to low-variance data problems. Under normal operating conditions, the performance of $f_H(i,j)$ in detecting low-variance data problems (such as un-updated data) could be unsatisfactory. This is because under normal operating conditions, the normalized standard deviations for clean measurements tend to be close to one, while the normalized standard deviations for low-variance data (such as un-updated data) tend to be close to zero. Therefore, under normal operating conditions, $f_H(i,j)$ between clean data and un-updated data would remain close to one, while $f_L(i,j)$ between clean data and un-updated data would be a very large number. However, under normal operating conditions, $f_H(i,j)$ between two clean data sets would be a small positive number (close to zero), and $f_L(i,j)$ between two clean data sets would lie around one. Therefore, the $f_L(i,j)$ value between un-updated data and clean data tends to be much larger than $f_L(i,j)$ value between two clean data sets, leading to a better detection performance. This performance difference is further demonstrated through case studies.

3.2 Density-Based Outlier Detections for PMU Data

Built upon the above similarity metrics, LOF analysis, which is a density-based outlier detection technique, is applied to solve the low-quality data detection problem. In this subsection, procedures for calculating LOFs are briefly discussed. The mathematical definition of ``density" is presented below. Details of LOF analysis can be found in [15].

3.2.1 Calculation of K-Distance(P)

Let the measurement matrix M be a database consisting of PMU measurements. Let p, q, o be some objects in *M*, each object represents a column in *M*. Let *k* be a positive integer. The distance between p and q, denoted by *d*(*p*,*q*), is defined by $f_H(p,q)$ or $f_L(p,q)$.

For any positive integer k, the *k-distance* of object p, denoted by *k-distance*(*p*), is defined as the distance *d*(*p*,*o*) between p and an object $o \in M$ such that:

a) for at least *k* objects $o' \in M\backslash\{p\}$ it holds that $d(p,o') \leq d(p,o)$, and

b) for at most k-1 objects $o' \in M\backslash\{p\}$ it holds that $d(p,o') < d(p,o)$.

In the above definition, $o' \in M\backslash\{p\}$ denotes $\{o' : o' \in M, o' \notin \{p\}\}$.

Intuitively, *k-distance*(*p*) represents the distance between object *p* and the k^{th} nearest neighbor of *p*. The value of *k-distance*(*p*) provides a measure on the density around the object *p*. For the same number of *k*, smaller *k-distance*(*p*) indicates higher density around *p*.

3.2.2 Identification of K-Distance Neighborhood of P

Given *k-distance*(*p*), the *k-distance neighborhood of p* contains every object whose distance from *p* is not greater than the *k-distance*. This concept is defined in (3.5).

$$N_{k-distance(p)}(p) = \{q \in M\backslash\{p\} | d(p,q) \leq k - distance(p)\} \quad (3.5)$$

These objects q are called the *k-nearest neighbors of p*.

3.2.3 Calculation of Reachability Distance of Object P from Object O

The reachability distance of object *p* with respect to object *o* is defined in (3.6).

$$reach - dist_k(p,o) = max\{k - distance(o), d(p,o)\} \quad (3.6)$$

Intuitively, if object *p* is far away from object *o*, then the reachability distance between *p* and *o* is simply their actual distance *d*(*p*,*o*). However, if they are "sufficiently" close to each other, the actual distance *d*(*p*,*o*) is replaced by the *k-distance*(*o*). The reason is that in doing so, the statistical fluctuations of *d*(*p*,*o*) for all the *p*'s close to *o* can be significantly reduced. The strength of this smoothing effect can be controlled by the parameter *k*. The higher the value of *k*, the more similar the reachability distances for objects within the same neighborhood.

Figure 3.1 illustrates the relationship among true distance $d(p_3,o)$, *k-distance*(*o*), $reach - dist_k(p_1,o)$, and $reach - dist_k(p_5,o)$. In this example, *k*=3, and true distance $d(\cdot)$ is the Euclidean distance. According to the above definitions, *k-distance*(*o*) represents the distance between object o and the k^{th} nearest neighbor of *o*. Therefore, when *k*=3, *k-distance*(*o*) = $d(p_3,o)$, where p_3 is the third nearest neighbor of *o*. The radius of the circle in Figure 3.1

represents *k-distance*(o). Since true distance $d(p_1, o) <$ *k-distance*(o), and true distance $d(p_5, o) >$ *k-distance*(o), the reachability distance between p_1 and o is $reach - dist_k(p_1, o) = k - distance(o)$, while the reachability distance between p_5 and o is $reach - dist_k(p_5, o) = d(p_5, o)$. These reachability distances $reach - dist_k(\cdot)$, developed through the comparison between the true distances $d(\cdot)$ and the *k-distance*(o), will then be used to formulate the local outlier factor.

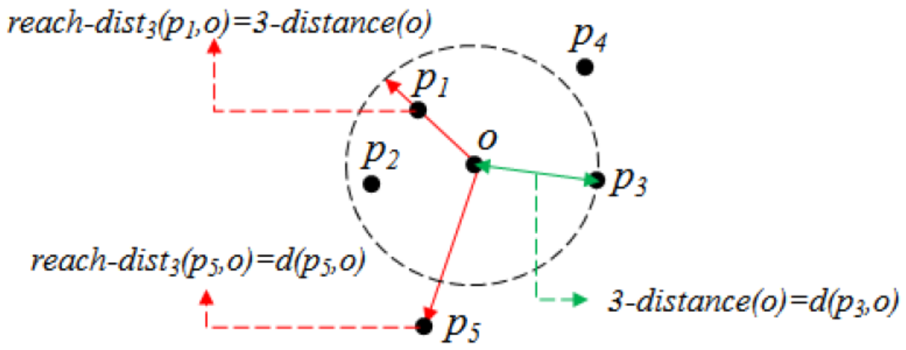


Figure 3.1 *k-distance*(o), *reach-dist*$_k(p_1,o)$, and *reach-dist*$_k(p_5,o)$ when k=3 [16]

3.2.4 Calculation of Local Reachability Density of P

The *local reachability density of p* is defined as

$$lrd_{MinPts}(p) = \frac{1}{\frac{\sum_{o \in N_{MinPts}(p)} reach\text{-}dist_{MinPts}(p,o)}{|N_{MinPts}(p)|}} \tag{3.7}$$

where $N_{MinPts}(p) = N_{MinPts-distance(p)}(p)$, and $MinPts$ is a positive integer.

Intuitively, the local reachability density of an object p is the inverse of the average reachability distance based on the *MinPts*-nearest neighbors of p.

3.2.5 Calculation of LOF of P

The local outlier factor of p is defined as

$$LOF_{MinPts}(p) = \frac{\sum_{o \in N_{MinPts}(p)} \left(\frac{lrd_{MinPts}(o)}{lrd_{MinPts}(p)}\right)}{|N_{MinPts}(p)|} \tag{3.8}$$

The local outlier factor of object p captures the degree to which p is an local outlier. It is the average of the ratio of the local reachability density of p and those of p's *MinPts*-nearest neighbors. It is easy to see that the lower p's local reachability density is, and the higher the local reachability densities of p's *MinPts*-nearest neighbors are, the higher the LOF value of p is.

3.3 Robust Detection Criterion and Parameter Selections

In order to improve the robustness of the proposed approach, the following detection criterion and parameter selection procedure are applied to the algorithm.

3.3.1 Robust Detection Criterion

Due to the propagation delay of electro-magnetic waves, PMUs installed at different locations of a large-scale power system may respond to physical disturbances at the time instants slightly asynchronous with each other. If a short moving data window is chosen for the algorithm, this slight time shift may cause false alarms under fault-on operating conditions. In order to avoid the false alarms without introducing too much computational burden, PMU measurements within the current moving data window are identified to contain low-quality data only if there are already l consecutive moving data windows prior to this current window, whose LOF values exceed the threshold value. l is an integer slightly less than the length of the moving data window. This criterion would introduce a small detection delay to the proposed algorithm. However, since the length of the moving data window is set to be short for the purpose of online application, the delay would be an insignificant value.

3.3.2 Parameter Selections

Three parameters need to be determined for the proposed algorithm: number of nearest neighbors (*MinPts*) of each object, length of the moving data window, and LOF thresholds for various similarity metrics. These parameters can be determined through off-line training using historical data. In order to reduce the detection delay, the length of moving data window should remain short. The *MinPts* value can be selected to be around half of the total number of PMU channels, by assuming the total number of low-quality curves at each time window should be less than the total number of high-quality PMU curves.

According to the previous discussions, the overall flowchart of the proposed algorithm is shown in Figure 3.2. Key steps for implementing this low-quality data detection approach are as follows.

Step 1: Create the current moving data window by reading in PMU measurements at the latest time instant.

Step 2: Compute $f_H(\cdot)$ and $f_L(\cdot)$ values for each pair of PMU curves.

Step 3: Compute LOF value of each PMU curve, based on $f_H(\cdot)$ and $f_L(\cdot)$. For each PMU curve, the LOF value can be calculated following the equations in the previous subsection.

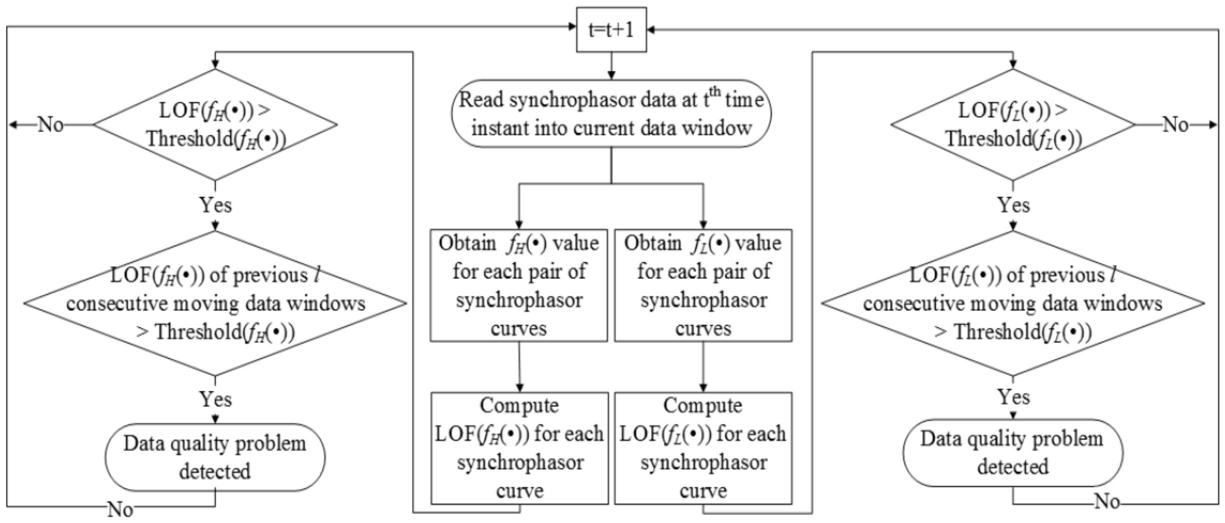
Step 4: If the LOF value corresponding to $f_H(\cdot)$ or $f_L(\cdot)$ of the i^{th} PMU curve exceeds the threshold, go to Step 5; otherwise, go to Step 7.

Step 5: If the previous l consecutive LOF values corresponding to $f_H(\cdot)$ or $f_L(\cdot)$ of the i^{th} PMU curve exceed the threshold, go to Step 6; otherwise, go to Step 7.

Step 6: The i^{th} PMU curve is detected to contain low-quality data at current time window.

Step 7: Move the data window to the next time instant, and go back to Step 1.

Although the above calculation procedure involves looping process for the LOF calculation of each PMU curve, there is no time-consuming computation (such as matrix inversion, decomposition, etc.) involved in the above procedure. All the operations within the looping process request light computational efforts. The computational burden of the entire process is not significant. The computational performance of the proposed algorithm is demonstrated through the case studies.



t=t+1
Read synchrophasor data at t^th time instant into current data window
Obtain $f_H(\bullet)$ value for each pair of synchrophasor curves
Obtain $f_L(\bullet)$ value for each pair of synchrophasor curves
Compute LOF($f_H(\bullet)$) for each synchrophasor curve
Compute LOF($f_L(\bullet)$) for each synchrophasor curve
LOF($f_H(\bullet)$) > Threshold($f_H(\bullet)$)
No
Yes
LOF($f_H(\bullet)$) of previous l consecutive moving data windows > Threshold($f_H(\bullet)$)
Yes
Data quality problem detected
No
LOF($f_L(\bullet)$) > Threshold($f_L(\bullet)$)
No
Yes
LOF($f_L(\bullet)$) of previous l consecutive moving data windows > Threshold($f_L(\bullet)$)
Yes
Data quality problem detected
No

Figure 3.2 Overall flowchart of the proposed approach [16].

4. Case Studies

The proposed approach is tested using both synthetic and real-world PMU data. Low-quality measurements caused by various reasons are used to verify the effectiveness of the approach. In all the following test cases, a unique set of algorithm parameters are used: moving data window length = 20 data points; LOF threshold corresponding to $f_H(\cdot)$ = 10; LOF threshold corresponding to $f_L(\cdot)$ = 100; Number of neighboring data for LOF algorithm = 0.5 × number of PMU curves. In order to demonstrate the proposed method is capable to detect low-quality data under fault-on operating conditions, a system physical disturbance (fault) is recorded by the PMU data in each test case.

4.1 Case Study with Synthetic Data

The synthetic PMU measurements are sampled from the simulation results of a standard IEEE-14 test system, with a sampling rate of 50Hz. A three-phase line-to-ground fault is presented while running the simulation. In each test case, one type of low-quality data is randomly inserted into a subset of the test data.

4.1.1 Synthetic Data with High Sensing Noise

This test data set contains 14 synthetic voltage magnitude measurement curves, where 3 of them (No. 1, 5, 14) contain Gaussian noises lasting from 6s to 6.4s, with a signal-to-noise ratio (SNR) of 40dB. Figure 4.1 shows the 3 curves with data quality problems.

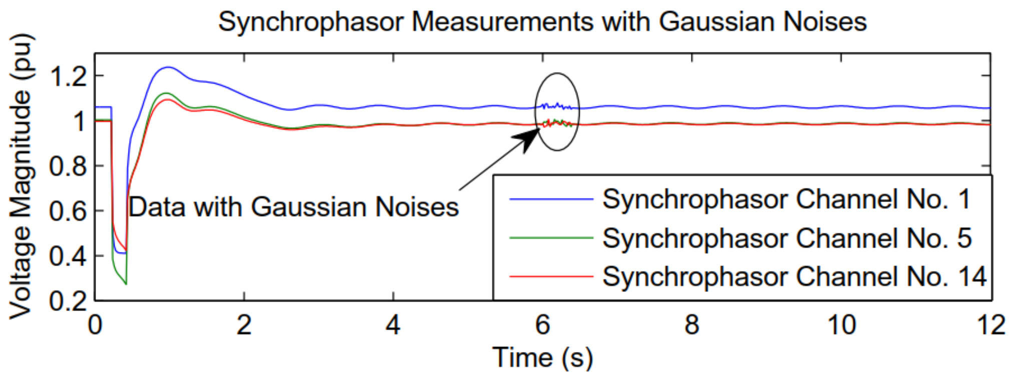


Figure 4.1 Synthetic PMU measurements with high sensing noise [16].

Table 4.1 presents the detection results. It shows that all the 3 noisy data segments are successfully detected, without introducing any false alarm by the physical disturbance. A small detection delay (less than 0.38s) is introduced, due to the length of the moving data window. The average computing time for each moving data window is 0.0161s. Figure 4.2 presents the LOF values of all the PMU curves, when data quality problem or physical disturbance is presented. This comparison shows that the LOF values exceed the threshold when low-quality data is presented, while remain below the threshold when physical disturbance is presented. The results indicate the proposed method is able to detect low-quality PMU data while avoiding false alarms caused by system physical disturbances.

Table 4.1 Detection Results for Synthetic PMU Data with High Sensing Noise 1

Index of PMU with High Noise	**Starting time of Noisy Segment**	**Ending Time of Noisy Segment**
1	6.22s (LOF = 620.5)	6.78s (LOF = 31.9)
5	6.34s (LOF = 429.1)	6.78s (LOF = 73.3)
14	6.34s (LOF = 418.6)	6.76s (LOF = 48.2)

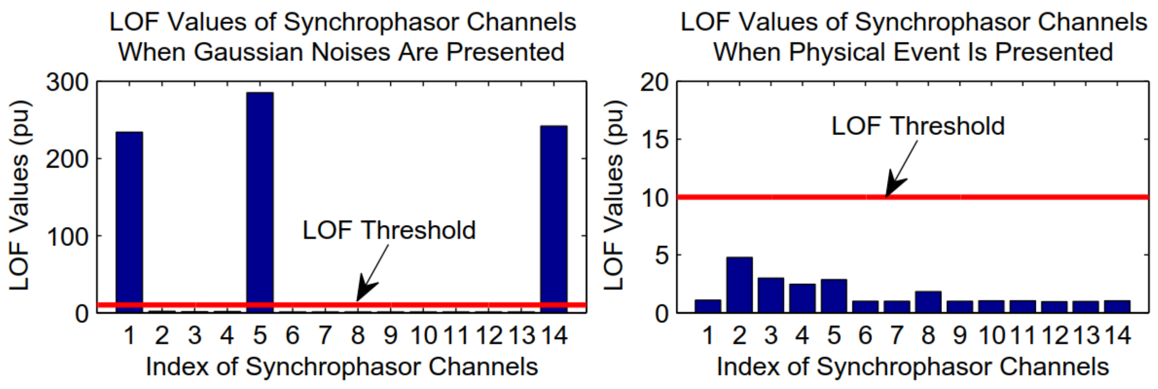


Figure 4.2 LOF values of synthetic PMU channels when physical disturbance (right) or high sensing noise (left) is presented [16].

4.1.2 Synthetic Data with Spikes

This test data set contains 47 synthetic real power measurement curves, where 4 of them (No. 3, 6, 30, 45) contain data spikes lasting from 6.3s to 6.4s. These spikes can be caused by problems such as data loss or time skew of GPS clock [10]. Figure 4.3 shows the 4 curves with data quality problems.

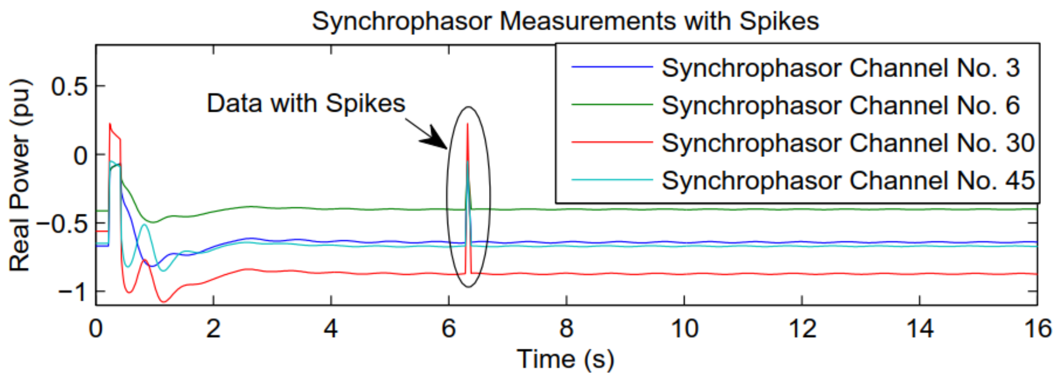


Figure 4.3 Synthetic PMU measurements with data spikes [16].

The detection results are shown in Table 4.2. All the 4 spikes are detected and no false alarm is introduced by physical disturbance. The detection delay introduced by the length of the moving data window is less than 0.36s. The average computing time for each moving data window is 0.0627s. Figure 4.4 presents the LOF values of all the PMU curves, when data quality problem or physical disturbance is presented. It is clear that low-quality data would cause the LOF values to exceed the threshold, while system physical disturbances would not cause a significant increment in LOF values.

Table 4.2 Detection Results for Synthetic PMU Data with Spikes [16]

Index of PMU with Data Spike	Starting time of Spike Segment	Ending Time of Spike Segment
3	6.46s (LOF = 107.7)	6.76s (LOF = 85.2)
6	6.46s (LOF = 113.9)	6.76s (LOF = 90.4)
30	6.48s (LOF = 102.3)	6.76s (LOF = 71.5)
45	6.44s (LOF = 270.3)	6.76s (LOF = 58.7)

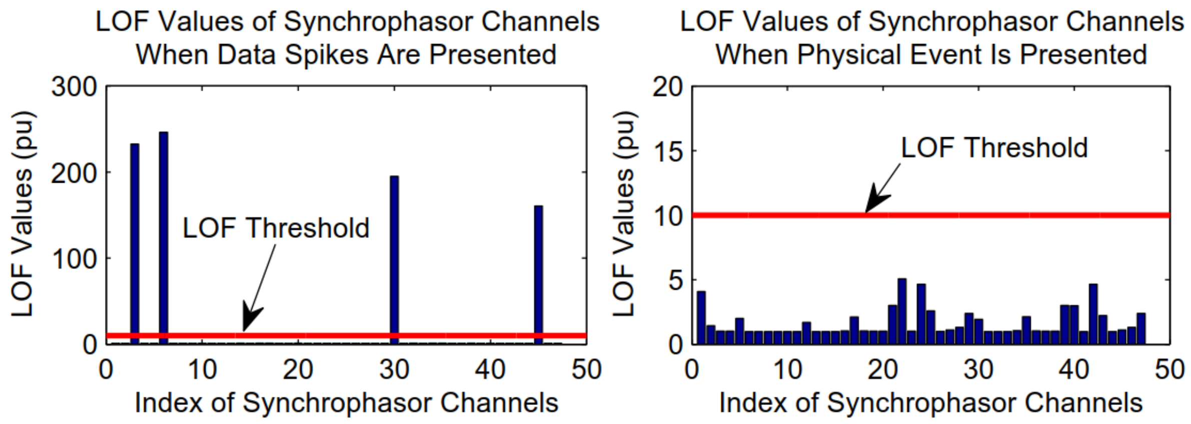


Figure 4.4 LOF values of synthetic PMU channels when physical disturbance (right) or data spike (left) is presented [16].

4.1.3 Synthetic Data with Un-Updated Data

This test data set contains 14 synthetic voltage magnitude measurement curves, where 3 of them (No. 6, 12, 13) contain un-updated data lasting from 6s to 6.4s. Figure 4.5 shows the 3 curves with data quality problems.

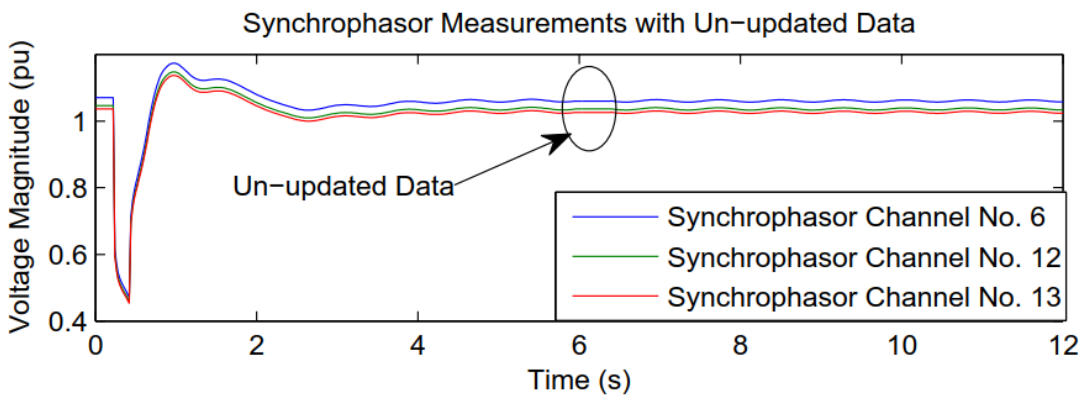


Figure 4.5 Synthetic PMU measurements with un-updated data [16].

Table 4.3 presents the detection results. The 3 un-updated data segments are detected, while the presence of physical disturbance does not cause any false alarm. The detection delay introduced by the length of the moving data window is less than 0.36s, and the average computation time for each moving time window is 0.0128s.

Table 4.3 Detection Results for Synthetic PMU Data with Un-updated Data [16]

Index of PMU with Un-updated Data	**Starting time of Un-updated Segment**	**Ending Time of Un-updated Segment**
6	6.36s (LOF = 3423.6)	6.40s (LOF = 3519.5)
12	6.36s (LOF = 3423.6)	6.40s (LOF = 3519.5)
13	6.36s (LOF = 3423.6)	6.40s (LOF = 3519.5)

4.1.4 Synthetic Data with False Data Injection

This test data set contains 47 synthetic real power measurement curves, where 4 of them (No. 15, 21, 29, 42) contain false data injections lasting from 6s to 6.4s. Figure 4.6 shows the 4 curves with data quality problems.

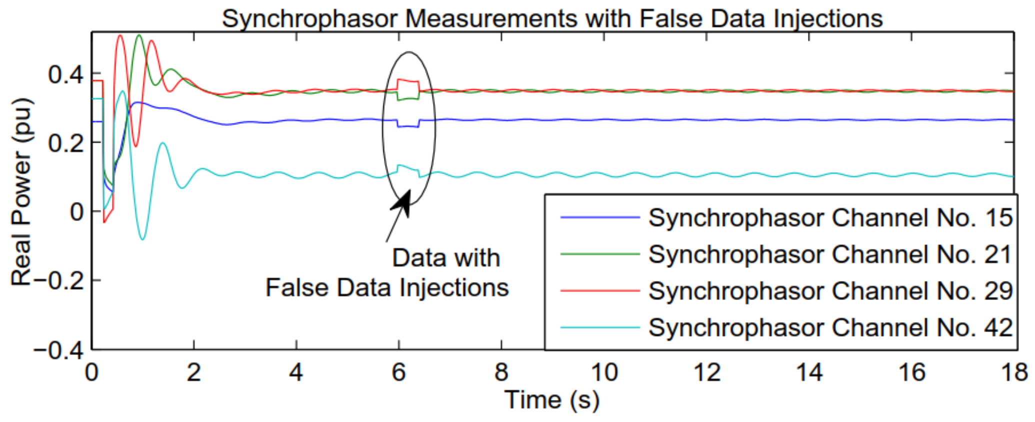


Figure 4.6 Synthetic PMU measurements with false data injection [16].

The detection results are shown in Table 4.4. Although physical disturbance is presented, all the 4 false data injections are correctly detected and no false alarm is introduced. The detection delay caused by the length of the moving data window is less than 0.38s. The average computing time for each moving data window is 0.0627s.

In all the above case studies using synthetic PMU measurements, the maximum detection delay is less than 0.4s, and the maximum computing time for each moving data window is less than 0.1s. It is summarized in [17] that the data latency requirements for online quasi-steady-state applications (state estimation, small signal stability analysis, oscillation analysis, voltage stability analysis, etc.) range from 1s to 5s. It is clear that both the detection delay and the computing time of the proposed method satisfy the latency requirements for PMU-based online quasi-steady-state applications. Therefore, the proposed method is suitable for online detection of low-quality PMU measurements, in order to improve the accuracy of these PMU-based applications.

Table 4.4 Detection Results for Synthetic PMU Data with Un-updated Data [16]

Index of PMU with False Data Injection	Starting time of Injected Data Segment	Ending Time of Injected Data Segment
15	6.32s (LOF = 39.7)	6.78s (LOF = 30.9)
21	6.32s (LOF = 25.7)	6.78s (LOF = 19.9)
29	6.32s (LOF = 14.1)	6.78s (LOF = 10.7)
42	6.34s (LOF = 10.8)	6.72s (LOF = 10.9)

4.2 Case Study with Real-World Data

High-quality PMU measurements obtained from a real-world power grid are used to test the proposed approach. The sampling rate of the data is 100Hz. A line-tripping fault is recorded by the data. In each test case, one type of low-quality data is manually inserted to a randomly-chosen subset of the test data, so that the ground truth of the existence of low-quality data is known for sure.

4.2.1 Real-World Data with High Sensing Noise

This test data set contains 39 real-world voltage magnitude measurement curves, where 4 of them (No. 10, 15, 23, 29) contain Gaussian noises lasting from 1s to 1.2s, with a SNR of 40dB. The SNR of the original clean data set is tested to be well below 40dB. Figure 4.7 shows the 4 curves with data quality problems.

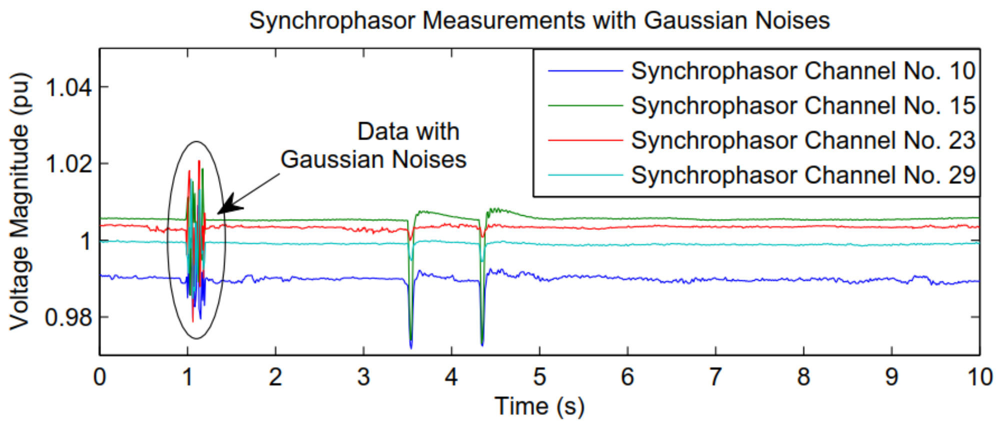


Figure 4.7 Real-world PMU measurements with high sensing noise [16].

Table 4.5 presents the detection results. It shows that all the 4 noisy data segments are successfully detected, without introducing any false alarm by the physical disturbance. A small detection delay (less than 0.19s) is introduced, due to the length of the moving data window. The average computing time for each moving data window is 0.0376s. Figure 4.8 presents the LOF values of all the PMU curves, when data quality problem or physical disturbance is presented. This comparison shows that the LOF values exceed the threshold when low-quality data is presented, while remain below the threshold when physical disturbance is presented. The results indicate the proposed method is able to detect low-quality PMU data while avoiding false alarms caused by physical disturbances.

Table 4.5 Detection Results for Real-World PMU Data with High Sensing Noise [16]

Index of PMU with High Noise	Starting time of Noisy Segment	Ending Time of Noisy Segment
10	1.17s (LOF = 286.9)	1.39s (LOF = 30.0)
15	1.16s (LOF = 577.8)	1.39s (LOF = 43.3)
23	1.16s (LOF = 206.3)	1.39s (LOF = 12.3)
29	1.16s (LOF = 328.2)	1.39s (LOF = 35.1)

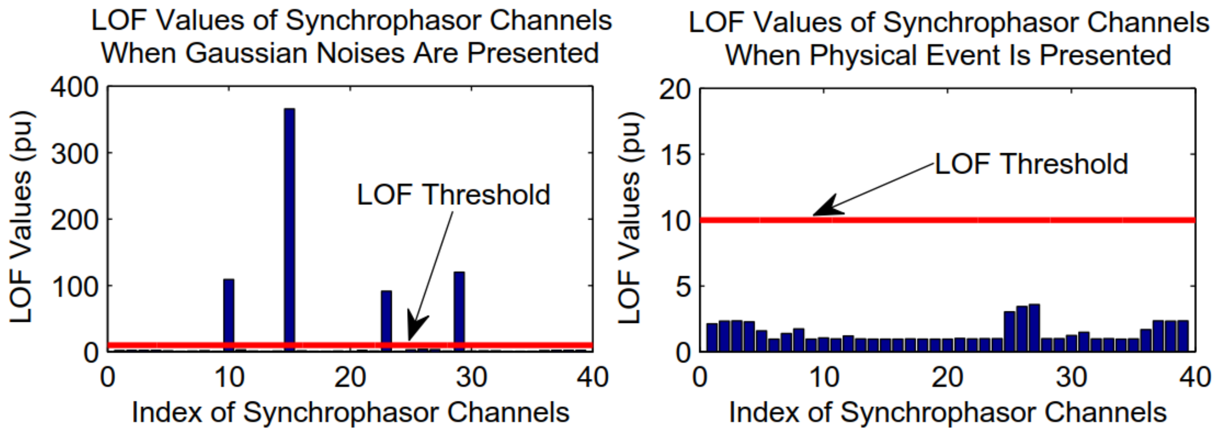


Figure 4.8 LOF values of real-world PMU channels when physical disturbance (right) or high sensing noise (left) is presented [16].

4.2.2 Real-World Data with Spikes

This test data set contains 22 real-world real power measurement curves, where 4 of them (No. 3, 6, 20, 21) contain data spikes at the time instant of 1.06s. In this test case, the length of each data spike is one sample. This test scenario is created in order to test the performance of the algorithm in detecting single data dropout. Figure 4.9 shows the 4 curves with data quality problems.

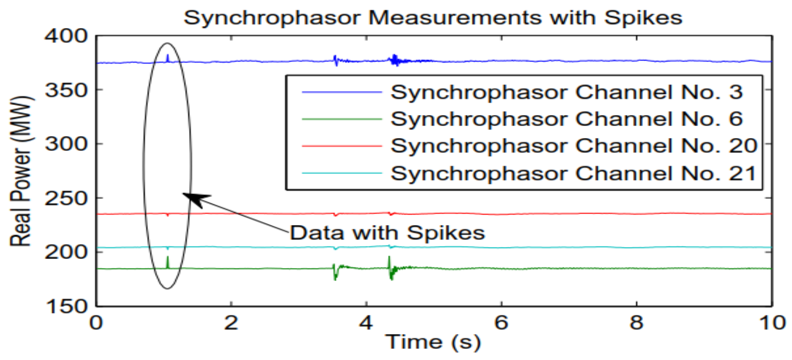


Figure 4.9 Real-world PMU measurements with data spikes [16].

The detection results are shown in Table 4.6. All the 4 spikes are detected and no false alarm is introduced by physical disturbance. The detection delay introduced by the length of the moving data window is less than 0.19s. The average computing time for each moving data window is 0.0150s. Figure 4.10 presents the LOF values of all the PMU curves, when data quality problem or physical disturbance is presented. It is clear that low-quality data would cause the LOF values to exceed the threshold, while system physical disturbances would not cause a significant increment in LOF.

Table 4.6 Detection Results for Real-World PMU Data with Spikes [16]

Index of PMU with Data Spike	**Starting time of Spike Segment**	**Ending Time of Spike Segment**
3	1.22s (LOF = 52.0)	1.25s (LOF = 28.2)
6	1.22s (LOF = 124.8)	1.25s (LOF = 69.2)
20	1.22s (LOF = 50.5)	1.25s (LOF = 27.2)
21	1.22s (LOF = 71.7)	1.25s (LOF = 39.5)

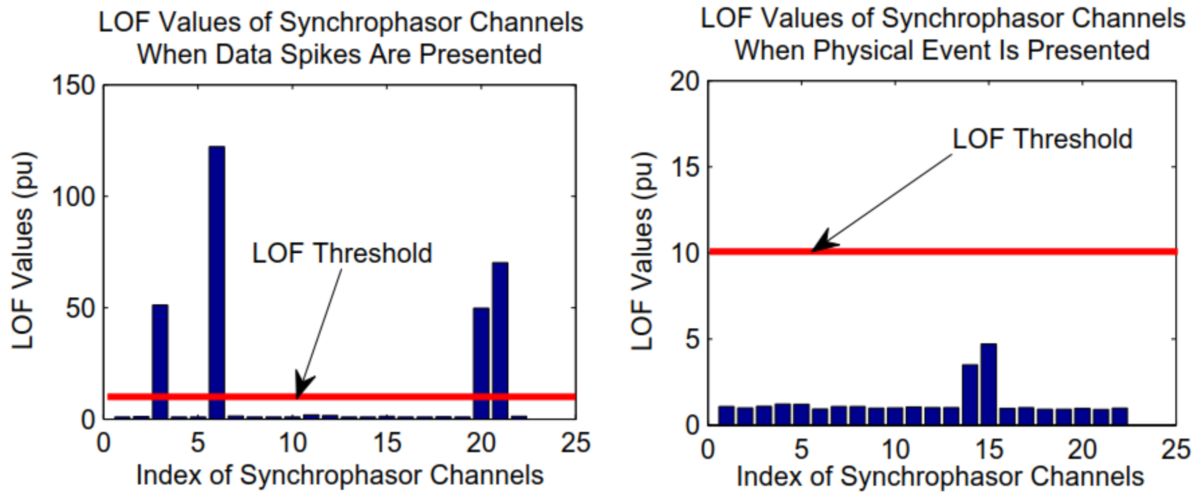


Figure 4.10 LOF values of real-world PMU channels when physical disturbance (right) or data spike (left) is presented [16].

4.2.3 Real-World Data with Un-Updated Data

This test data set contains 13 real-world current magnitude measurement curves, where 4 of them (No. 1, 5, 7, 13) contain un-updated data lasting from 1s to 1.2s. Figure 4.11 shows the 4 curves with data quality problems.

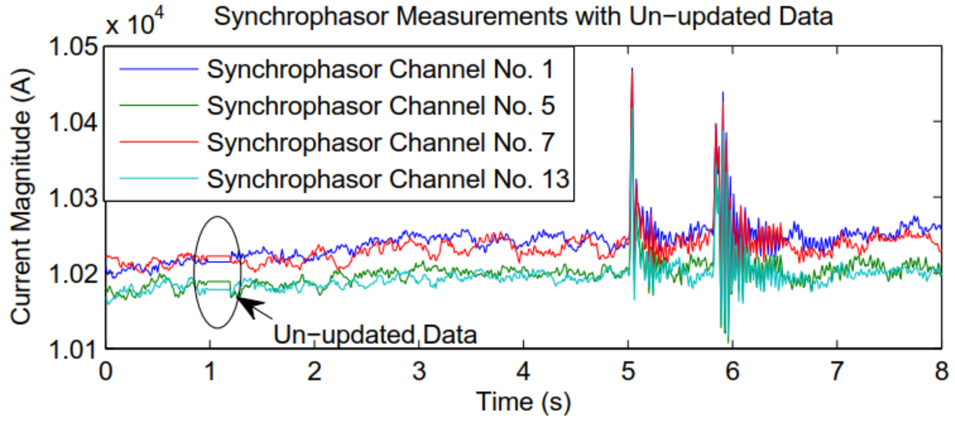


Figure 4.11 Real-world PMU measurements with un-updated data [16].

Table 4.7 presents the detection results. The 4 un-updated data segments are detected, while the presence of physical disturbance does not cause any false alarm. The detection delay introduced by the length of the moving data window is less than 0.18s, and the average computation time for each moving data window is 0.0115s.

Table 4.7 Detection Results for Real-World PMU Data with Un-updated Data [16]

Index of PMU with Un-updated Data	**Starting time of Un-updated Segment**	**Ending Time of Un-updated Segment**
1	1.18s (LOF = 4637.2)	1.20s (LOF = 4537.2)
5	1.18s (LOF = 4637.2)	1.20s (LOF = 4537.2)
7	1.17s (LOF = 3317.8)	1.20s (LOF = 4537.2)
13	1.18s (LOF = 4637.2)	1.20s (LOF = 4537.2)

Figure 4.12 presents the current magnitude data obtained from PMU channels No. 1 and No. 2, where PMU channel No. 1 contains un-updated data from 1s to 1.2s, and PMU channel No. 2 contains clean data only. Figure 4.13 presents the normalized deviations of the two PMU channels, as the computation data window moves with time. It is clear that: 1) under normal operating conditions, the normalized deviations of clean data segments lie close to one; 2) under fault-on operating conditions, the normalized deviations of clean data segments increase significantly; 3) the normalized deviations of un-updated data segments decrease towards zero.

Figure 4.14 presents the $f_H(i,j)$ and $f_L(i,j)$ values of PMU channels No. 1 and No. 2, as the computation data window moves with time. Figure 4.15 presents the LOF values of PMU channels No. 1 and No. 2, as the computation data window moves with time. It is clear from Figure 4.14 and Figure 4.15 that $f_L(i,j)$ is more sensitive to the un-updated data than $f_H(i,j)$, and therefore leads to a better detection performance for un-updated data.

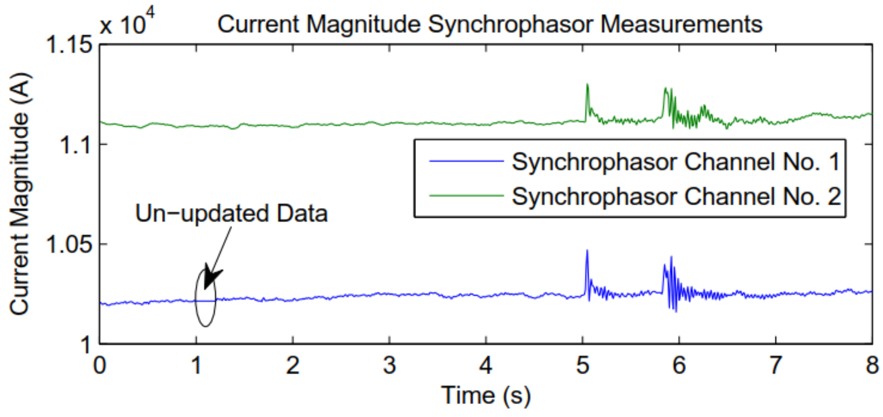


Figure 4.12 Real-world current magnitude PMU measurements [16].

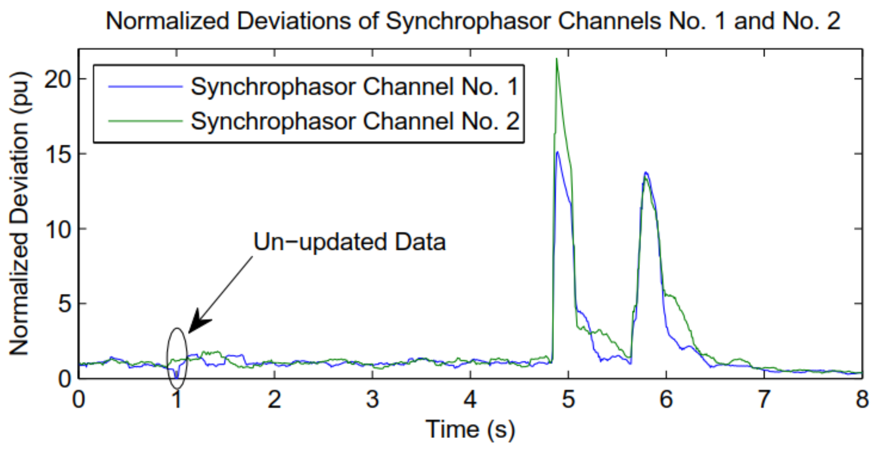


Figure 4.13 Normalized deviation of PMU channel No. 1 and No. 2 [16].

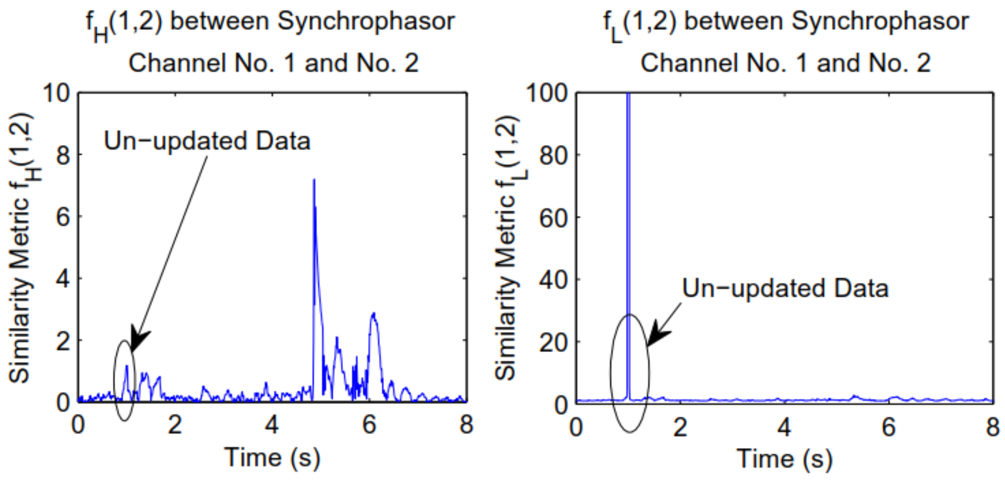


Figure 4.14 Similarity metric $f_H(i,j)$ (left) or $f_L(i,j)$ (right) between PMU channels No. 1 and No. 2 [16].

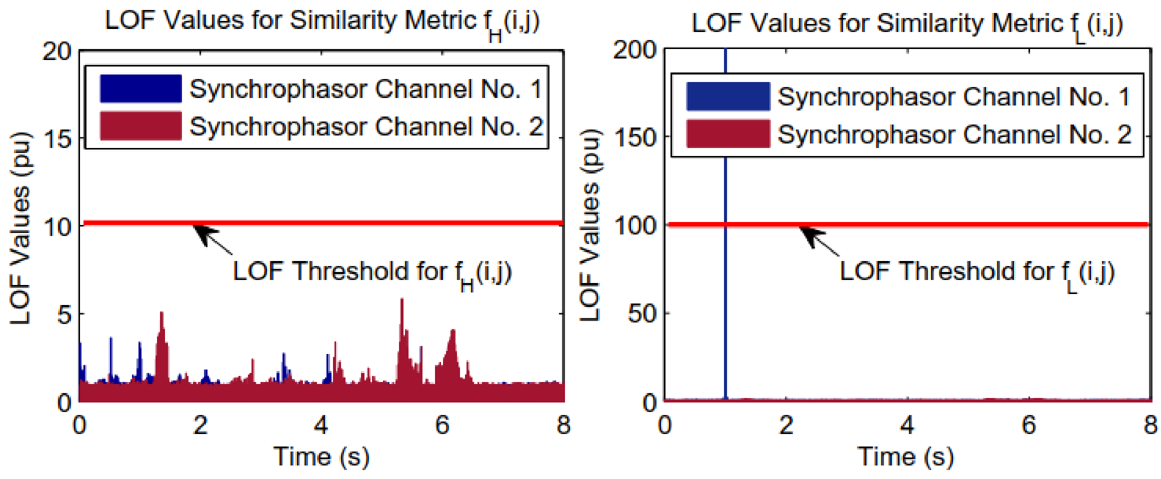


Figure 4.15 LOF values when similarity metric $f_H(i,j)$ (left) or $f_L(i,j)$ (right) is applied [16].

4.2.4 Real-World Data with False Data Injection

This test data set contains 39 real-world voltage magnitude measurement curves, where 4 of them (No. 2, 20, 27, 37) contain false data injections lasting from 1s to 1.2s. Figure 4.16 shows the 4 curves with data quality problems.

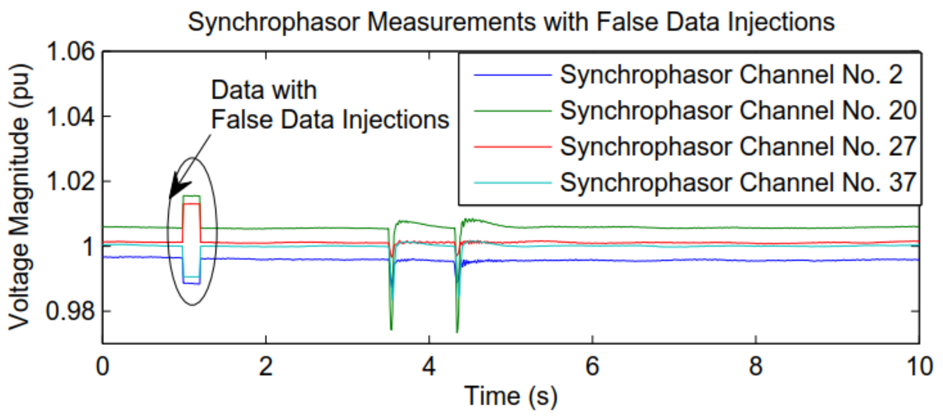


Figure 4.16 Real-world PMU measurements with false data injection [16].

The detection results are shown in Table 4.8. Although physical disturbance is presented, all the 4 false data injections are correctly detected and no false alarm is introduced. The detection delay caused by the length of the moving data window is less than 0.19s. The average computing time for each moving data window is 0.0475s.

In all the above case studies using real-world PMU measurements, the maximum detection delay is less than 0.2s, and the maximum computing time for each moving data window is less than 0.05s. It is summarized in [17] that the data latency requirements for online quasi-steady-state applications (such as state estimation, small signal stability analysis, oscillation analysis, voltage stability analysis, etc.) range from 1s to 5s. It is clear that both the detection delay and the computing time of the proposed method satisfy the latency requirements for PMU-based online quasi-steady-state applications. Therefore, the proposed method is suitable for online detection of low-quality PMU measurements, in order to improve the accuracy of these PMU-based applications.

Table 4.8 Detection Results for Real-World PMU Data with False Data Injections [16]

Index of PMU with False Data Injection	**Starting time of Injected Data Segment**	**Ending Time of Injected Data Segment**
2	1.16s (LOF = 74.3)	1.39s (LOF = 71.2)
20	1.16s (LOF = 117.7)	1.39s (LOF = 111.1)
27	1.16s (LOF = 95.5)	1.39s (LOF = 91.6)
37	1.16s (LOF = 383.4)	1.39s (LOF = 365.7)

Since the detection delay of the proposed algorithm is mainly caused by the length of the moving data window, the delay could be estimated and removed when the occurrence time of the low-quality data is reported. By doing this, the reported occurrence time of the low-quality data could be very close to its actual occurrence time.

For power grids with a large number of PMUs, the computation speed of the proposed algorithm could be further improved by applying the detection algorithm in a decentralized framework. In large systems, multiple detection engines could be applied to process PMU measurements obtained from different physical locations or control areas (such as different states or different local control centers). PMUs lying far from each other could be grouped into different subgroups, and be processed in parallel by different detection engines. This decentralized framework could help reduce the number of PMU channels that need to be processed by each detection engine, and therefore improve the computation speed of each detection engine. Since this method does not require any system-wide information (such as system topology), it can be easily decentralized without spending extra effort on creating the reduced or equivalent system model.

Meanwhile, parallel processing could also help improve the online computation performance of the proposed algorithm. Multiple processors could be applied at each detection engine, so that several consecutive moving data windows could be processed by different processors at the same time. This parallel technique could improve the overall computation speed when the proposed algorithm is applied to power systems with a significant number of PMUs.

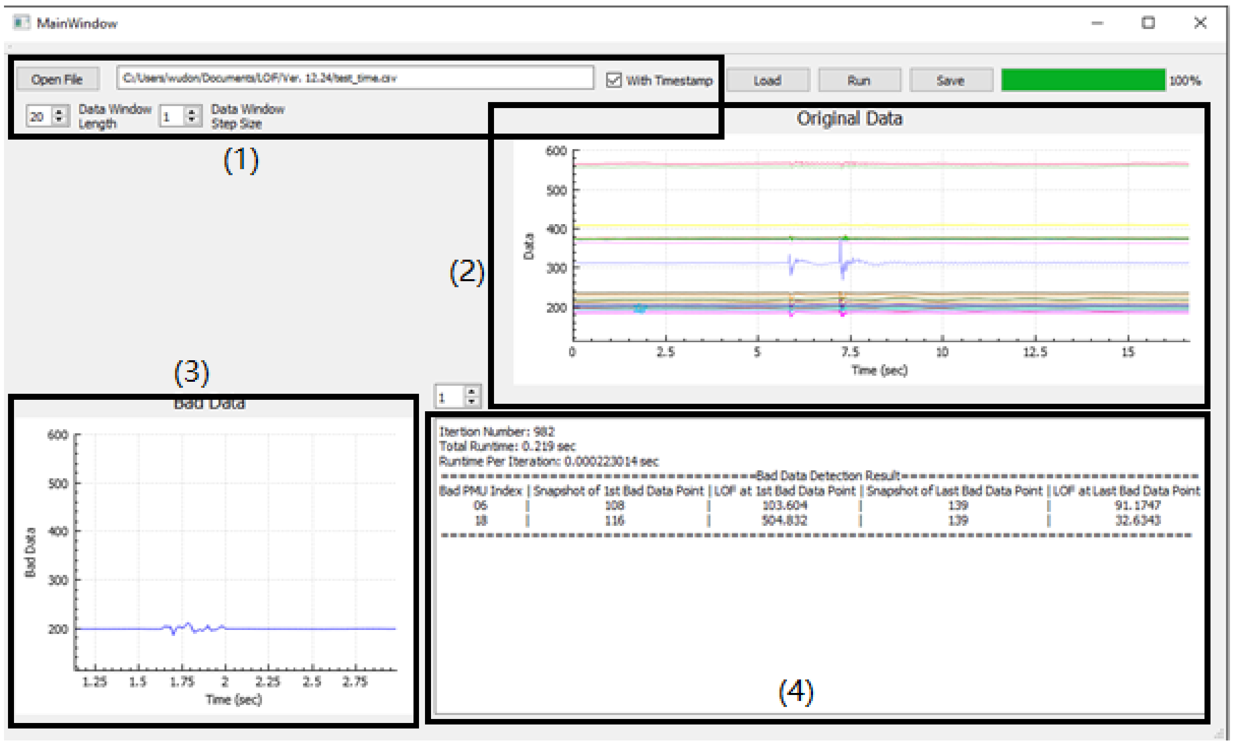
5. Software Implementation

5.1 Software Introduction

The algorithm discussed above is implemented as an open-source desktop toolkit [18] that provides convenient user experience. The executable program and source codes of the toolkit are available for public download through GitHub. A graphical user interface (GUI) was designed for the toolkit, in order to improve its user-friendliness. Moreover, this software features cross-platform capability by using the Qt application framework [19] as its development environment.

5.2 Software User Interface

The GUI is developed using Qt cross-platform software developer [20]. The Qt Software Development Kit (SDK) features a multi-platform compatibility which allows the software to be migrated to other hardware platforms with slight modification to the source codes. Figure 5.1 presents a screenshot of the detection software:



```
Itertion Number: 982
Total Runtime: 0.219 sec
Runtime Per Iteration: 0.000223014 sec
=====================================Bad Data Detection Result=====================================
Bad PMU Index | Snapshot of 1st Bad Data Point | LOF at 1st Bad Data Point | Snapshot of Last Bad Data Point | LOF at Last Bad Data Point
     06       |              108               |          103.604          |               139               |          91.1747
     18       |              116               |          504.832          |               139               |          32.6343
===================================================================================================
```

Figure 5.1 Screenshot of the detection software

The GUI contains four sections that are labeled (1) – (4) in Fig. 5.1:

Section 1: User input that let the user to choose offline PMU measurement data file and adjust the value of user-input parameters used in the detection algorithm.

Section 2: A plot of offline PMU measurement timeseries imported in section 1 by the, showing the voltage magnitude versus time curve for each PMU measurement channel.

Section 3: Zoom-in plots of identified low-quality data segments. The user can use the scroller located to the upper right of the plot to switch between different segments.

Section 4: A summary of the detection result, including:

1. Calculation time of the algorithm indicating the efficiency
2. Starting and ending time of each detected low-quality data segment
3. Channel number of each detected low-quality data segment

5.3 Potential Future Development

This detection software has immense potential for further development. The proposed algorithm is designed for online detection, while the software is currently only compatible with offline PMU data. Collaboration with the industry could potentially lead to interfacing with online PMU data streams and provide real-time detection and analysis for power system operators. Also, the algorithm can be implemented in different platforms including embedded devices for various applications.

6. Conclusions

This report presents a framework that is possible for online detection and improvement of PMU data quality issues. The proposed approach formulates the low-quality PMU data as spatio-temporal outliers among all the PMU measurements and performs detection through a density-based local outlier detection algorithm. Similarity metrics are proposed to quantify the spatio-temporal similarities among multi-time-instant PMU measurements. The proposed approach has satisfactory performance under both normal and fault-on operating conditions. It requires no prior information on system modeling and topology. The computation speed of the proposed algorithm is suitable for online applications. Synthetic and real-world PMU measurements are used to verify the effectiveness of the proposed approach. This framework, if successful, could potentially boost up system operators' confidence of PMU-based analytics in modern power systems.

Built upon this work, future research could focus on developing similarity metrics with more sensitive and robust performance, identifying root causes of the low-quality problems, and correcting the low-quality PMU data.

References

[1] E. Ghahremani and I. Kamwa, “Dynamic state estimation in power system by applying the extended kalman filter with unknown inputs to phasor measurements,” *Power Systems, IEEE Transactions on*, vol. 26, pp. 2556–2566, Nov 2011.

[2] D. Kosterev, “Hydro turbine-governor model validation in pacific northwest,” Power Systems, IEEE Transactions on, vol. 19, pp. 1144–1149, May 2004.

[3] I. Kamwa, R. Grondin, and Y. Hebert, “Wide-area measurement based stabilizing control of large power systems-a decentralized/hierarchical approach,” *Power Systems, IEEE Transactions on*, vol. 16, pp. 136–153, Feb 2001.

[4] J. Bertsch, C. Carnal, D. Karlson, J. McDaniel, and K. Vu, “Wide-area protection and power system utilization,” *Proceedings of the IEEE*, vol. 93, pp. 997–1003, May 2005.

[5] California ISO, “Five year synchrophasor plan,” tech. rep., California ISO, Nov 2011.

[6] W. Qi, “Comparison of differences between SCADA and WAMS real-time data in dispatch center,” in *12th International Workshops on Electric Power Control Centers (EPCC)*, Jun 2013.

[7] S. Ghiocel, J. Chow, G. Stefopoulos, B. Fardanesh, D. Maragal, B. Blanchard, M. Razanousky, and D. Bertagnolli, “Phasor-measurement-based state estimation for synchrophasor data quality improvement and power transfer interface monitoring,” *Power Systems, IEEE Transactions on*, vol. 29, pp. 881–888, March 2014.

[8] K. Jones, A. Pal, and J. Thorp, “Methodology for performing synchrophasor data conditioning and validation,” *Power Systems, IEEE Transactions on*, vol. 30, pp. 1121–1130, May 2015.

[9] K. Martin, “Synchrophasor data diagnostics: detection & resolution of data problems for operations and analysis,” in *Electric Power Group Webinar Series*, Jan 2014.

[10] Q. Zhang and V. Venkatasubramanian, “Synchrophasor time skew: Formulation, detection and correction,” in *North American Power Symposium (NAPS)*, 2014, pp. 1–6, Sept 2014.

[11] S. Dutta and T. Overbye, “Information processing and visualization of power system wide area time varying data,” in *Computational Intelligence Applications In Smart Grid (CIASG)*, 2013 IEEE Symposium on, pp. 6–12, April 2013.

[12] M. Wang, J. Chow, P. Gao, X. Jiang, Y. Xia, S. Ghiocel, B. Fardanesh, G. Stefopolous, Y. Kokai, N. Saito, and M. Razanousky, “A low-rank matrix approach for the analysis of large amounts of power system synchrophasor data,” in *System Sciences (HICSS), 2015 48th Hawaii International Conference on*, pp. 2637–2644, Jan 2015.

[13] L. Liu, M. Esmalifalak, Q. Ding, V. Emesih, and Z. Han, “Detecting false data injection attacks on power grid by sparse optimization,” *Smart Grid, IEEE Transactions on*, vol. 5, pp. 612–621, March 2014.

[14] M. Wu and L. Xie, “Online identification of bad synchrophasor measurements via spatio-temporal correlations,” in 2016 Power Systems Computation Conference (PSCC), pp. 1–7, June 2016.

[15] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, “Lof: Identifying density-based local outliers,” in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, SIGMOD ’00, (New York, NY, USA), pp. 93–104, ACM, 2000.

[16] M. Wu and L. Xie, “Online detection of low-quality synchrophasor measurements: A data-driven approach,” *IEEE Transactions on Power Systems*, vol. 32, pp. 2817–2827, July 2017.

[17] P. Kansal and A. Bose, “Bandwidth and latency requirements for smart transmission grid applications,” *IEEE Transactions on Smart Grid*, vol. 3, pp. 1344–1352, Sept 2012
[18] D. Wu, “Desktop Bad-Data Detection Application”, https://github.tamu.edu/dqwu/LOF-Bad-Data-Detection, January 2018
[19] The Qt Company Ltd, “Open Source Licensing”, https://doc.qt.io/qt-5/opensourcelicense.html, 2018
[20] The Qt Company Ltd, “About Qt”, https://wiki.qt.io/About_Qt, March 2018

Part-II

Multi-signal Analytical Guesstimation using Intelligent Collaboration (MAGIC) for synchrophasor data

Vaithianathan "Mani" Venkatasubramanian
Yang Zheng, Graduate Student
Mohammadreza Maddipour Farrokhifard, Graduate Student

Washington State University

For information about this project, contact

Vaithianathan “Mani” Venkatasubramanian
Washington State University
School of Electrical Engineering and Computer Science
355 NE Spokane St. EME 37
Pullman, WA 99164-2752
Tel: 509-335-6452
Fax: 509-335-3818
Email: mani@eecs.wsu.edu

Power Systems Engineering Research Center

The Power Systems Engineering Research Center (PSERC) is a multi-university Center conducting research on challenges facing the electric power industry and educating the next generation of power engineers. More information about PSERC can be found at the Center’s website: http://www.pserc.org.

For additional information, contact:

Power Systems Engineering Research Center
Arizona State University
527 Engineering Research Center
Tempe, Arizona 85287-5706
Phone: 480-965-1643
Fax: 480-727-2052

Notice Concerning Copyright Material

PSERC members are given permission to copy without fee all or part of this publication for internal use if appropriate attribution is given to this document as the source material. This report is available for downloading from the PSERC website.

© 2018 Washington State University. All rights reserved

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Background

With the proliferation of phasor measurement unit (PMU) devices across power system, there is growing use of PMU data in applications. However, PMU data can get lost as it flows through the data process path from the point of measurement to the point of use. The efforts to reconstruct the missing synchrophasor data have been challenging. The traditional methods such as linear interpolation are inaccurate and cannot preserve the ring-down features of the system responses, when the data is lost during system events. This report introduces a data-driven approach that uses available measurements from nearby signals as cross-references to obtain accurate reconstructions of the missing data. The problem is formulated into a matrix approximation problem, where a collaborative filtering algorithm is applied. To adapt the algorithm into the power system context, the report designs methods for normalizing scales, identifying neighbors and screening reasonable reconstruction results. The methodology has been tested using field data obtained from PMUs installed in the Western and Eastern Interconnections. The results indicate that, with well-tuned hyper-parameters, the proposed technique provides highly accurate reconstructions in preserving system modal information and the results match well with those from actual values.

1.2 Literature Survey

With the proliferation of phasor measurement unit (PMU) devices across the power system, there is growing use of PMU data in applications that enhance grid operations [1], analytics [2], [3], and strategic planning [4]. Some of these applications are critical and require accurate, reliable data, delivered on time [5]. However, data losses often happen as data flows up from measurement at the PMU to the application. To fit the data for using in applications, effort has been made to reconstruct missing PMU data [6].

In industry, linear interpolation is commonly used to patch data gaps [8], which can potentially lose valuable modal information especially when an event or oscillation occurs.

The model-based method is another traditional solution. State estimators generate pseudo-measurements from historical state estimates based on complicated modeling of power system dynamics [9], [10]. The accuracy of pseudo-measurements largely depends on the correct modeling of system dynamics.

A denser coverage of PMUs makes model-free methods possible [11]. Because PMU data are sampled at synchronized instants, and measurements of nearby PMUs are correlated through power system topology, PMU data exhibit low-dimensional structure despite the high-dimensionality of raw data. Specifically, the matrix that contains measurements of nearby PMUs at multiple time instants is approximately low-rank [12], [13]. Therefore, reconstructing missing PMU data can be formulated as a low-rank matrix completion problem. The existing matrix completion algorithms, such as singular value thresholding (SVT) [14], and information cascading matrix completion (ICMC) [15] have one fundamental assumption, that is, each missing entry happens independently of others. The independence assumption does not adequately describe the

PMU data correlations. To characterize temporal and spatial (signal) correlations, [16] assumes the independence of PMU signals and independence of different time instants respectively.

Instead of separating temporal and spatial correlations, available measurements from recent history and any available measurements at the present time from nearby PMU signals can be used as cross-references to reconstruct missing data. In light of this idea, the data itself is represented as a measurement matrix, giving for each signal-time pair, a value that represents the measurement of that signal at that time. Some entries of the matrix are missing due to data losses. The goal is to fill in the blanks of the measurement matrix. The difference between cross-reference approach and traditional matrix completion is a measure of involved PMU signals. Cross-reference approach only uses close PMU signals, while matrix completion uses available PMUs signals over distinct control regions.

1.3 Scope of the Project

We formulate the domain problem of missing PMU data reconstruction as that of collaborative filtering widely used in recommendation system in the social network [17]. Collaborative filtering algorithm predicts missing rating data in user-item matrix according to similar users to make targeted recommendations to interested potential customers. We combine two primary approaches of collaborative filtering (i.e., latent-space-based and neighborhood-based [18]) to solve the PMU data reconstruction problem.

1.4 Organization of the Report

The rest of the report is organized as follows: Section II is the problem formulation and a brief introduction of the latent-space-based collaborative filtering algorithm. Section III implements collaborative filtering into the environment of PMU data reconstruction with three strategies that have significant impacts on the accuracy of results: neighborhood selection, normalization, and global optimization. Section IV tunes the hyper-parameters and configure the algorithms in order to enhance its performance. Section V describes experiments that show the effectiveness of the proposed technique. Section VI concludes with some future directions.

2. Collaborative Filtering Algorithm

2.1 Problem Formulation

Suppose we are given measurements for s signals and t time instants. Here s designates the number of signals and t designates the number of time instants. Formally, let $M = [m_{ij}]_{s\times t}$ be the measurement matrix, where each element m_{ij} represents the measurement of signal i at time j with its value either being a real number or a missing value [19]. The $(i,\ j)$ pairs for which m_{ij} is available are stored in the subset $\mathrm{A} = \{(\mathrm{i},\mathrm{j})|\mathrm{m}_{\mathrm{ij}})\ is\ available\}$. Likewise, $N\ A$ represents the subset of measurements that are not available for some reasons. In this setup, collaborative filtering is designed to estimate missing values in M based on the available values.

2.2 Latent Factor Model

Latent factor models predict the blank entries in the measurement matrix by conjecturing that the matrix is actually the product of two long, thin matrices. If we start with the measurement matrix M, with s rows and t columns (i.e., there are s signals and t time instants), then we want to find a matrix U with s rows and d columns and a matrix V with d rows and t columns, such that the product UV closely approximates M in those entries where M is non-blank. If so, we have mapped both signals and time instants to a joint latent factor space of dimensionality d. We can then use the entries in UV to estimate the corresponding blank entries in measurement matrix M [20]. This process is called UV-decomposition of M. In this way, the incomplete measurement matrix M is approximated by the product of two low dimensional factor space U and V. A visualization of the approximation is given in Fig. 1.

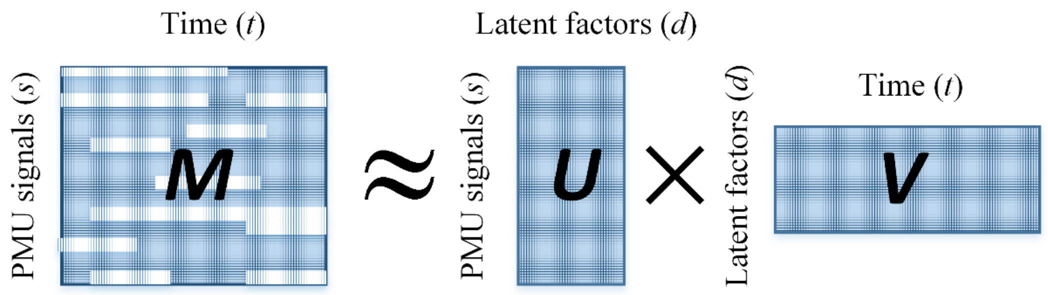


Figure 1. Visualization of matrix-factorization-based collaborative filtering.

More specifically, let $U = [u_j]$ be the PMU signals factor matrix, where the factor vectors $u_i = \subseteq \mathbb{R}^d$ for all $i = 1, \dots, s$, and let $V = [v_j]$ be the time factor matrix, where the factor vectors $v_j = \subseteq \mathbb{R}^d$ for all $j = 1, \dots, t$. The resulting dot product or inner product, $< u_i, v_j >$, captures the interaction between signal i and time j, i.e., the measurement of the signal at certain time.

If the measurements were fully observable and d was sufficiently large, we could expect that each entry of M is the inner product of corresponding signal and time factor vectors: $m_{ij} = < u_i, v_j >$, $\forall\ i, j$. In practice, however, M has unavailable entries, the approximated U and V might not give the perfect inner product $< u_i, v_j >$ that is exactly the same with observed values of m_{ij}. We need to minimize the differences between the approximated predictions $p_{ij} = < u_i, v_j >$ and observed values m_{ij} to obtain u and v.

2.3 Objective Function / Loss Function

A loss function evaluates how close the predictions are to the observed values, that is, how close the entries of product UV is to the non-blank entries of M. A typical choice is the root-mean-square error (RMSE):

$$RMSE = \sqrt{\frac{1}{n_A}\sum_{(i,j)}\left(p_{ij} - m_{ij}\right)^2} \tag{1}$$

where p_{ij} and m_{ij} are predicted and observed values of signal i at time j respectively. Here, we use p_{ij} for the element in row i and column j of the product matrix $P = UV$. n_A is the number of available (non-blank) entries in M. For training U and V purpose, we only calculate all available (non-blank) entries in M. For evaluation purpose, we calculate all blank entries in M.

Since RMSE function measures the quality of the prediction, such a quality function is often referred to as the objective function. Minimizing the sum of the squares is equivalent to minimizing the square root of the average square, so we generally omit the average and square root steps. The objective/loss function minimized for training u and v is:

$$\left(u_i, v_j\right) = \arg \min_{u_i, v_j} \sum_{(i,j)\in A} \left(p_{ij} - m_{ij}\right)^2 \tag{2}$$

where A is the set of available (non-blank) entries.

2.4 Regularization

One problem that often arises when performing a collaborative filtering algorithm is that although the RMSE may be small on the given training data, it does not do well predicting unavailable data. The reason is that the parameters approximate the available data so well that they do not reflect well the underlying process. To avoid overfitting, a common method is to append a regularization term to the objective function:

$$\left(u_i, v_j\right) = \arg \min_{u_i, v_j} \sum_{(i,j)\in A} \left(p_{ij} - m_{ij}\right)^2 + \lambda\left(\| u_i \|^2 + \| v_j \|^2\right) \tag{3}$$

where λ is the regularization parameter. Suitable selection of λ will be discussed in details in Section IV. To learn the model parameters u_i, v_j we minimize the regularized squared error.

2.5 Minimizing an Arbitrary Element in *U* and *V*

Minimizing the regularized squared error (3) is a least square problem. In this section, we develop the general formula for picking the minimum value for a single element in the matrix U or V.

Suppose we want to vary the blank entry u_{ij} and find the value of this element that minimizes the RMSE between M and UV. Note that u_{ij} affects only the elements in row i of the product $P = UV$. Thus, we need only concern with the elements:

$$p_{ih} = \sum_{c=1}^{d} u_{ic} v_{ch} + \sum_{c \neq j} u_{ic} v_{ch} + \left(u_{ij} + \Delta u_{ij}\right) v_{jh} \tag{4}$$

for all values of h such that m_{ij} is non-blank. In the expression above, we have replaced u_{ij}, the element we wish to vary, with the current value of u_{ij} and an adjustment Δu_{ij}, and we use the convention:

$\sum_{c \neq j}$ is shorthand for the sum for those column $c = 1, 2, \ldots, d$, except for $c = j$.
If m_{ih} is a non-blank entry of the matrix M, then the contribution of this element to the sum of the squared errors is:

$$(m_{ih} - p_{ih})^2 = \left(m_{ih} - \sum_{c \neq j} u_{ic} v_{ch} - \left(u_{ij} + \Delta u_{ij}\right) v_{jh} \right)^2 \tag{5}$$

We shall use another convention:
$\sum_h$ is shorthand for the sum over all h such that m_{ih} is non-blank.

Then we can write the sum of the squared errors that are affected by the value of Δu_{ij} as:

$$\sum_h \left(m_{ih} - \sum_{c \neq j} u_{ic} v_{ch} - \left(u_{ij} + \Delta u_{ij}\right) v_{jh} \right)^2 \tag{6}$$

We can find the value of Δu_{ij} that minimizes the squared error by setting the gradient with respect to Δu_{ij} equal to zero:

$$\sum_h v_{jh} \left(m_{ih} - \sum_{c=1}^{d} u_{ic} v_{ch} - \Delta u_{ij} v_{jh} \right) = 0 \tag{7}$$

We solve the above equation for Δu_{ij}, and get:

$$\Delta u_{ij} = \frac{\sum_h v_{jh} \left(m_{ih} - \sum_{c=1}^{d} u_{ic} e_{ih} \right)}{\sum_h v_{jh}^2} = \frac{\sum_h v_{jh} e_{ih}}{\sum_h v_{jh}^2} \tag{8}$$

If we compute error of each entry as $e_{ih} = m_{ih} - p_{ih}$, then Δu_{ij} can be expressed as:

$$\Delta u_{ij} = \frac{\sum_h v_{jh}(m_{ih} - p_{ih})}{\sum_h v_{jh}^2} = \frac{\sum_h v_{jh} e_{ih}}{\sum_h v_{jh}^2} \tag{9}$$

Similarly, there is an analogous formula for the optimum value of an element of V. If we want to vary v_{ij}, then the value of adjustment Δv_{ij} that minimizes the squared error is:

$$\Delta v_{ij} = \frac{\sum_g u_{gi} e_{gj}}{\sum_g u_{gi}^2} \tag{10}$$

Here, $\sum_g$ is shorthand for the sum over all g such that m_{gj} is non-blank.

The denominator of (9) is $\sum_h v_{jh}^2$ and the denominator of (10) is $\sum_g u_{gi}^2$. Those two terms can be ignored because they are not changing in each iteration. Discard the two terms and rewrite (9) and (10) in vector form, yielding:

$$\Delta u_i = \sum_h e_{ih} v_h \tag{11}$$

$$\Delta v_j = \sum_g e_{gj} u_g \tag{12}$$

If we use the regularized objective function, the derived formula will append the regularization factor λ:

$$\Delta u_i = \sum_h e_{ih} v_h + \lambda u_i \tag{13}$$

$$\Delta v_j = \sum_g e_{gj} u_g + \lambda v_j \tag{14}$$

2.6 Gradient Descent

The technique discussed above is an example of gradient descent. We are given some data points - the non-blank elements of the matrix M - and for each data point we find the direction of change that most decreases the error function: the RMSE between the current UV product and M. To find a UV-decomposition, we adjust all available elements of U and$V$, i.e. u_{ij} and v_{ij}, by minimizing the associated objective function in every iteration.

The algorithm loops through all available measurements in M. For each available measurement m_{ij}, a prediction p_{ij} is made, and the associated prediction error $e_{ij} = m_{ij} - p_{ij}$ is computed. For a given training case, we modify the u_{ij} and v_{ij} by moving in the opposite direction of the gradient. So the update rules for row vectors u and column vectors v are:

$$u_i := u_i - \alpha \Delta u_i = u_i - \alpha \sum_j (e_{ij} v_j + \lambda u_i) \tag{15}$$

$$v_j := v_j - \alpha \Delta v_j = v_j - \alpha \sum_i \left(e_{ij} u_i + \lambda v_j\right) \quad (16)$$

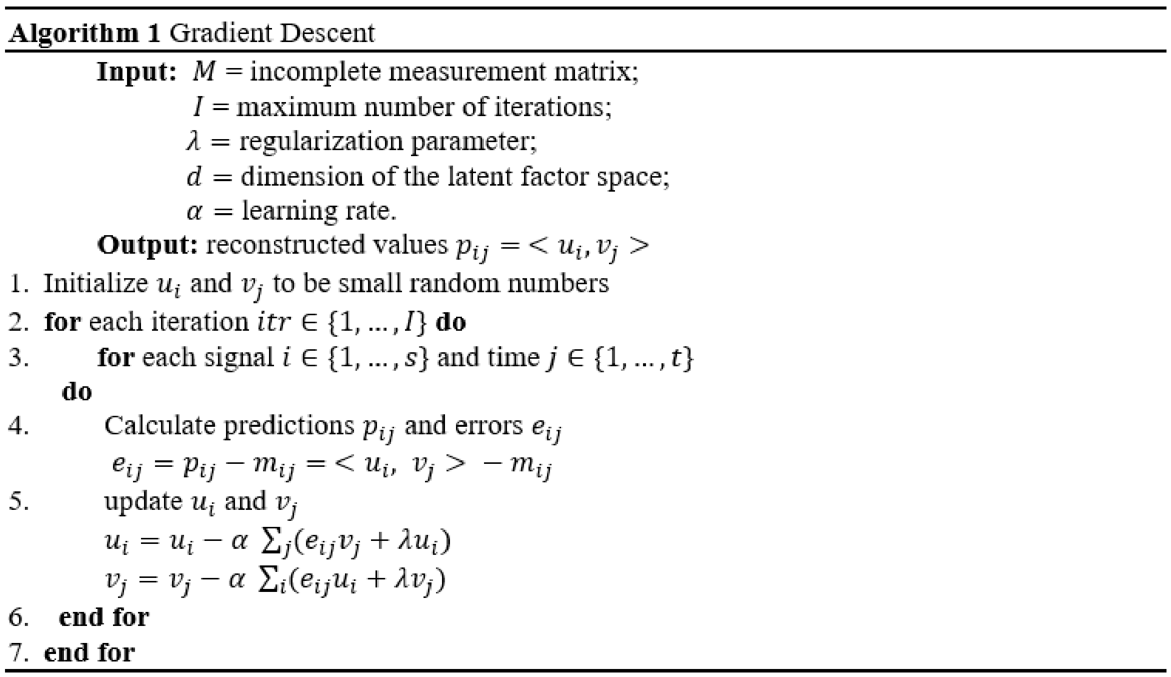
where α is the learning rate represents how quickly the prediction is getting close to the local optimum. One can expect better accuracy by dedicating separate learning rates α to each type of learned parameter [21]. However, in our case, the learning rate does not make a great impact on accuracy of reconstruction. Hence, when producing results for this report, the learning rate is not fully tuned. We use $\alpha = 0.01$, which is a typical choice for many collaborative filtering algorithms.

The process of gradient descent to minimize the associated objective / loss function is as follows:

Step 1 Initialize u and v by assigning small random numbers;
Step 2 Adjust u and v to make the objective function smaller;
Step 3 Repeat Steps 2 until a stopping criterion is satisfied;
Step 4 Use the obtained u and v to make final predictions $p_{ij} = < u_i, v_j >$.

The stopping criterion in this study is reaching the maximum number of iterations I, which assigned by users. Alternatively, we can track the amount of improvement in the RMSE obtained in each iteration of the optimization and stop when that improvement falls below a threshold. For example, after one iteration of updating both u and v, if the difference between the observed RMSE on good dataset is less than 1 bps (1 bps equals 0.0001), the iteration stops, and we use the obtained u and v to make final predictions (i.e. data reconstruction). The pseudo-code of gradient descent is shown in Table I.

Table 1: Pseudo-code of gradient descent



```
Algorithm 1 Gradient Descent
        Input:  $M$ = incomplete measurement matrix;
                $I$ = maximum number of iterations;
                $\lambda$ = regularization parameter;
                $d$ = dimension of the latent factor space;
                $\alpha$ = learning rate.
        Output: reconstructed values $p_{ij} = < u_i, v_j >$
1. Initialize $u_i$ and $v_j$ to be small random numbers
2. for each iteration $itr \in \{1, \dots, I\}$ do
3.     for each signal $i \in \{1, \dots, s\}$ and time $j \in \{1, \dots, t\}$
   do
4.         Calculate predictions $p_{ij}$ and errors $e_{ij}$
           $e_{ij} = p_{ij} - m_{ij} = < u_i, v_j > - m_{ij}$
5.         update $u_i$ and $v_j$
           $u_i = u_i - \alpha \sum_j (e_{ij} v_j + \lambda u_i)$
           $v_j = v_j - \alpha \sum_i (e_{ij} u_i + \lambda v_j)$
6.     end for
7. end for
```

3. Implementation Strategies for PMU Data Reconstruction

To make sure the collaborative filtering algorithm work well in the context of PMU data reconstruction, three very important considerations are 1) the normalization of measurements matrix, 2) the computation of the similarity weights and the selection of neighbors, and 3) optimization from randomness. The three components can have a significant impact on the accuracy of reconstruction results.

3.1 Identifying Neighbors and Similarity Weights

In power systems, although substations are connected by lines, they do not necessarily perform similar fluctuation in signal measurements, due to the phase difference. A filtering step where only the most likely signals are kept is one of the most critical aspects of building a collaborative filtering reconstruction, as it can have a significant impact on accuracy and performance. The most likely signals, also referred as neighbors, can be determined by a similarity measure.

In the measurement matrix M, each row vector represents one signal's data at t time instant. Consider a t-dimensional signal a as a vector $m_a \subseteq \mathbb{R}^t$, where m_a has entries of measurement values if available, and 0 otherwise. Here, we treat blanks as a 0 value. Cosine can be employed to find similar signals by computing the cosine of the angle that the two signal vectors form. The cosine similarity between two signals a and b would then be computed as:

$$\cos(m_a, m_b) = \frac{< m_a, m_b >}{\| m_a \| \| m_b \|} \tag{17}$$

While the sign of a cosine similarity indicates whether the correlation is direct or inverse, its magnitude (ranging from 0 to 1) represents the strength of the correlation. A larger (positive) cosine implies a smaller angle and therefore a smaller difference between two signals [22].

The pre-filtering of neighbors is an essential step that makes collaborative filtering approaches practicable and accurate by limiting the number of candidate neighbors to consider in the reconstructions. For each signal, only a list of the k nearest neighbors (k_NN) [23] whose cosine similarities have the greatest magnitude is kept. It is noteworthy that if the cosine similarity index value for the kept neighbor is negative, the inverse form of the neighbor is used for calculations. The number of neighbors k used in the prediction requires tuning, which will be further discussed in Section IV. In general, a small number of high-confidence neighbors is preferable to a large number of neighbors for which the cosine similarities are not trustworthy [24]. The overall computational efficiency is much less expensive than involving all signals in the reconstruction.

The cosine similarities not only allow the selection of trusted neighbors whose values are used in the reconstruction, but also provide the evaluations to give more or less importance to these neighbors. The principle of this strategy is to increase or reduce the magnitude of a signal value depending on the level of similarity between the signal and the missing signal. Specifically, each row vector of matrix M which stands for each signal, is multiplied by a weight factor: its corresponding cosine distance with the missing signal. Thus, each row of matrix M is penalized by a corresponding weight factor to take into account the significance of each signal. Then the signals with larger magnitude of cosine will be considered as “fully similar” and will play an important role in each other’s reconstruction. The implementation of identifying neighbors and similarity weights will be discussed in Section IV.

3.2 Normalization

Another consideration when reconstructing missing PMU data is the fact that each signal has its own scale of values to respond to the same level of fluctuation for an event. The scale could be very different, ranging from 0 to 2,000 for current magnitude, or 0 to 550,000 for voltage magnitude. Therefore, the data needs to be preprocessed to remove these influences before the analysis step of collaborative filtering. This problem is usually addressed by converting the measurements m_{ij} to normalized ones m_{ij}^{norm} [25] [26].

The idea is to normalize the matrix by subtracting the mean value and then dividing by the maximum value for a signal. A raw measurement m_{ij} is transformed to m_{ij}^{norm} by subtracting m_{ij} form the average $\overline{m_i}$ and then deviding by k_i the maximum value of the measurements given by this signal i in available (non-blank) measurement subset A:

$$m_{(i,j)\in A}^{norm} = \frac{m_{(i,j)\in A} - \overbrace{\frac{1}{t_A} \sum_{j \in A} m_{(i,j)\in A}}^{\overline{m_i}}}{k_i} \tag{18}$$

t_A are numbers of available (non-blank) columns elements that represent for time instants. That way, we turn below-average values into negative numbers and above-average values into positive numbers.

Note that the predicted data must be converted back to the original scale. If we choose to normalize M, then after we obtain the predictions, we need to undo the normalization in post-processing. The final prediction is created by multiplying k_i, then adding the aforementioned mean $\overline{m_i}$ back. Thus, if the algorithm results in $p_{(i,j)}^{norm}$ for an element in the normalized matrix, the value we predict for the element in the original measurement matrix is $p_{(i,j)}^{norm}$ plus whatever amount was subtracted during the normalization process:

$$p_{(i,j)} = \left(p_{(i,j)}^{norm}\right)(k_i) + \overline{m_i} \tag{19}$$

After normalizing the matrix, if we then calculate the cosine between signals, the cosine similarities will be different. With this change, the neighbors given by the cosine similarities can provide more accurate reconstructions. The effect of normalization will prove useful when we compare results with and without normalization in Section IV.

3.3 Optimization and Randomness

Finding the U and V with the least RMSE involves starting with some arbitrarily chosen U and V, and repeatedly adjusting U and V to make the RMSE smaller. When the RMSE does not get smaller, we find the local minimum U and V However, there is no guarantee that the best local minimum from one trial will be the global minimum. To not getting stuck in local minima and increase the chances of finding the global minimum, we need to pick many different starting points, that is, different choices of the initial matrices U and V leading to various results of UV-Decomposition. The process of repeating the reconstructions with random choices of starting points is called Monte-Carlo method. The final reconstruction can be the average of reconstruction results among the multiple rounds or the round with best reconstructions on the non-blank data entries. In Section IV, we will explain with field data how to find out the final reconstruction from local optimum.

The procedure of seeking an optimum solution from repeatedly computation is as follows:

Step 1 Normalize the matrix M;

Step 2 Select k nearest neighbors and multiply each neighbor vector by its corresponding cosine weight to form a new smaller matrix M', which is used in reconstruction steps;

Step 3 Repeat the UV-decomposition for multiple rounds by randomly picking initial matrices U and V;
Step 4 Take the average of reconstruction results among the multiple rounds;

Step 5 Undo the normalization.

Fig. 2 is the diagram of building a data reconstruction algorithm based on collaborative filtering and three strategies mentioned in this section. The procedure is applied to the case studies in Section V.

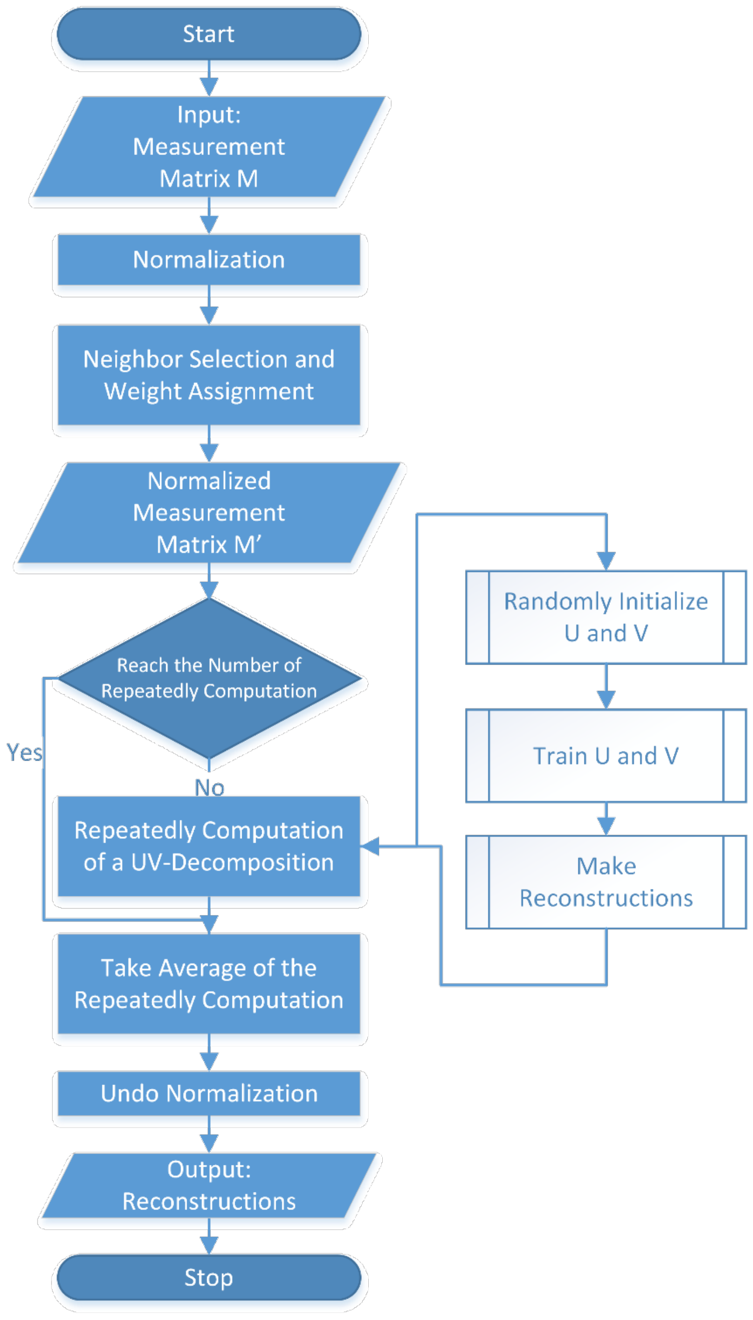

Start
Input: Measurement Matrix M
Normalization
Neighbor Selection and Weight Assignment
Normalized Measurement Matrix M'
Reach the Number of Repeatedly Computation
Yes
No
Repeatedly Computation of a UV-Decomposition
Randomly Initialize U and V
Train U and V
Make Reconstructions
Take Average of the Repeatedly Computation
Undo Normalization
Output: Reconstructions
Stop

Figure 2. Data reconstruction using collaborative filtering algorithm.

4. The Hyper-Parameter Tuning and Performance Evaluation

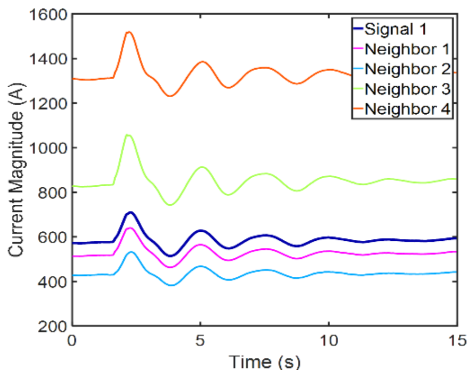
In this section, we implement the latent factor model in Section II and proposed strategies in Section III on field PMU data with a concentration on comparing the effects before and after applying the strategies, as well as showing how the strategies can enhance system performance. To configure the proper hyper-parameter values of the algorithm, we control for the number of nearest neighbors k, the number of iterations I, dimension of the latent factor space d and regularization factor λ to identify the appropriate configurations of the algorithm.

A set of archived PMU data of the Western Interconnection is used to evaluate the performance of the algorithm. The sampling rate is 30 samples per second. We select one-minute event data are float numbers spanning 774 signals measured at over 1,800 time instants. The data includes a total of 126, 124, 260 and 264 available PMU signals of bus voltage magnitude, bus voltage phase angle, line current magnitude and line current phase angle, respectively. An abnormal event causes the disturbances in measurement values. To manually create data loss, we pick some signals, erase consecutive data that contain the ring-down features for different time length during the event. The proposed technique is then applied to reconstruct the lost data. Root-mean-squared error (RMSE) is the main performance criteria of the quality of the results.

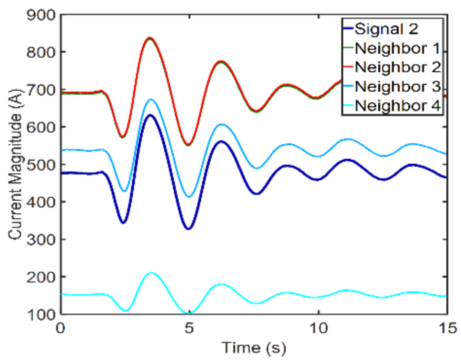
4.1 Identifying Neighbors and Similarity Weights

4.1.1 Neighborhood Selection

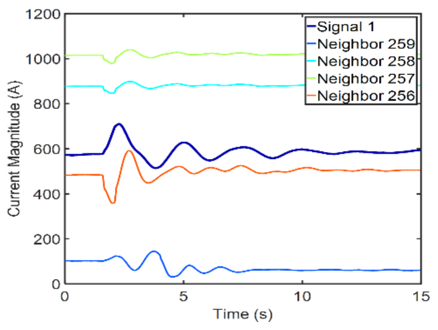
Neighbors selection is one of the most critical aspects of building a reconstruction system, as it can have a significant impact on both its accuracy and performance. We use cosine distance to measure similarities of signals.



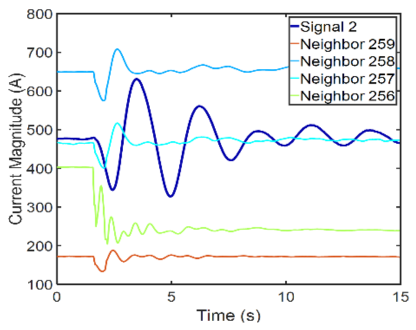
(a) Good neighbor selection of signal 1



(b) Good neighbor selection of signal 2



(c) Bad neighbor selection of signal 1



(d) Bad neighbor of signal 2

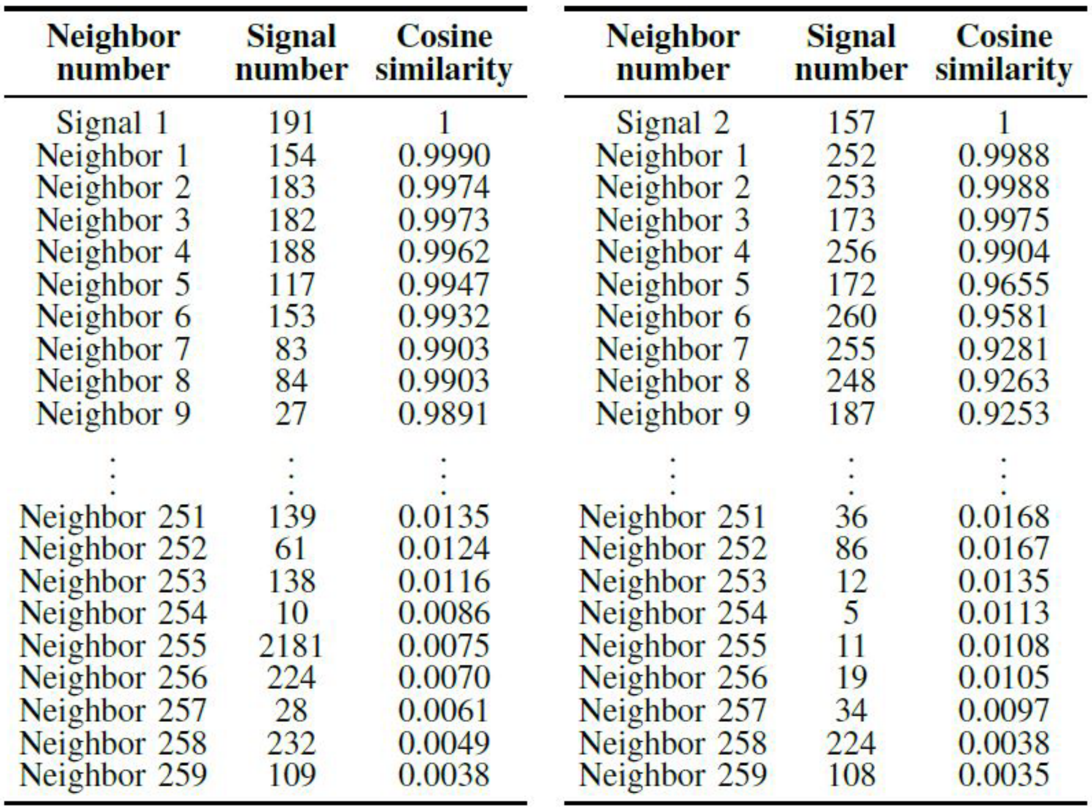
Figure 3. Neighbor Identification

Fig. 3 is current magnitude data of two signals that have phase lag between them. Signal 1 and signal 2 are the thick blue lines in Fig 3a and Fig 3b, respectively. The two signals have completely different changing directions. In this case, neighbors whose values are used in the reconstruction should be chosen carefully. If we accidentally chose the signals having opposite fluctuation with the missing signal, the accuracy usually drops, due to the fact that the good neighbors with strong relations with the missing signal are "diluted" by the bad ones.

There are 260 available signals of current magnitude in the system besides the above two signals. We compute the cosine similarity between signal 1 and the other 259 signal vectors as the inner product divided by the product of the vector norms. Then repeat the cosine calculation for signal 2. Table I shows the partial cosine similarity of the two signals and their candidate neighbors. Here, we only show the top 9 and the last 9 candidate neighbors for signal 1 and signal 2. The plots of these signals in Fig. 3 will give us a more intuitive idea about how cosine measures the similarity. From the plots, we can see that the cluster of neighbor 1 to 9 (Fig. 3a) is apparently a better choice than the cluster of neighbor 251 to 259 (Fig. 3c) if we want to find neighbors of signal 1.

The cosine values in Table II confirm this observation. The cosine of the missing signal and itself is 1. If the cosine similarities are close to 1, then the signals will be considered as similar and will likely play an important role in each other's reconstructions. However, if cosine similarities are small numbers, the signals' behaviors are in fact different, this may lead to poor reconstructions. So the cluster of neighbor 1 to 9 is a better choice than the cluster of neighbor 251 to 259. The same goes for signal 2.

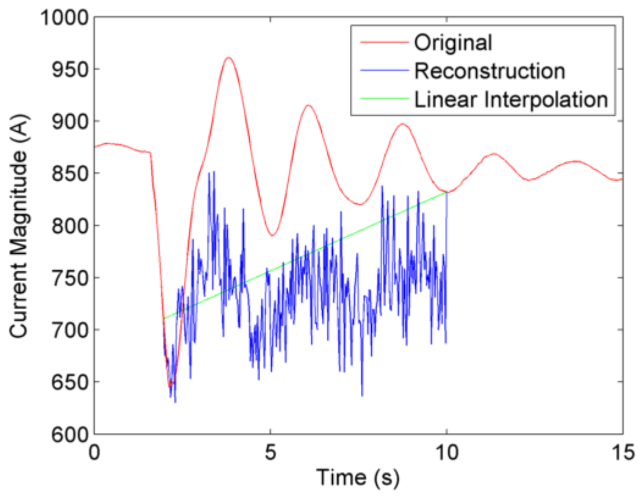
Table 2: Candidate neighbors and their cosine similarity



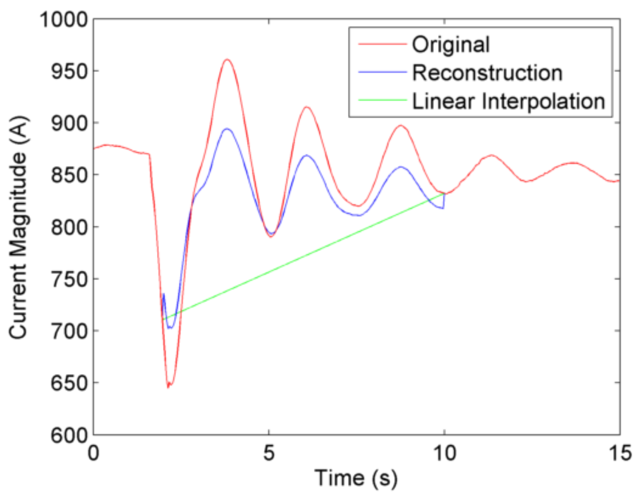
Neighbor number	Signal number	Cosine similarity	Neighbor number	Signal number	Cosine similarity
Signal 1	191	1	Signal 2	157	1
Neighbor 1	154	0.9990	Neighbor 1	252	0.9988
Neighbor 2	183	0.9974	Neighbor 2	253	0.9988
Neighbor 3	182	0.9973	Neighbor 3	173	0.9975
Neighbor 4	188	0.9962	Neighbor 4	256	0.9904
Neighbor 5	117	0.9947	Neighbor 5	172	0.9655
Neighbor 6	153	0.9932	Neighbor 6	260	0.9581
Neighbor 7	83	0.9903	Neighbor 7	255	0.9281
Neighbor 8	84	0.9903	Neighbor 8	248	0.9263
Neighbor 9	27	0.9891	Neighbor 9	187	0.9253
⋮	⋮	⋮	⋮	⋮	⋮
Neighbor 251	139	0.0135	Neighbor 251	36	0.0168
Neighbor 252	61	0.0124	Neighbor 252	86	0.0167
Neighbor 253	138	0.0116	Neighbor 253	12	0.0135
Neighbor 254	10	0.0086	Neighbor 254	5	0.0113
Neighbor 255	2181	0.0075	Neighbor 255	11	0.0108
Neighbor 256	224	0.0070	Neighbor 256	19	0.0105
Neighbor 257	28	0.0061	Neighbor 257	34	0.0097
Neighbor 258	232	0.0049	Neighbor 258	224	0.0038
Neighbor 259	109	0.0038	Neighbor 259	108	0.0035

4.1.2 Applying *k* Nearest Neighbors *(k-NN)*

Fig. 4 is the contrast of reconstruction results with *k-NN* or not. Fig. 4a is the reconstruction result of using all available signals in the measurement matrix. Many signals do not have similar fluctuation with the signal we want to reconstruct. So the reconstruction results in a signal with undesired noises. Fig. 4b shows the reconstruction result of using *k-NN*. Only the top *k* neighboring signals with highest similarity weights are involved in the reconstruction. Apparently, the second approach has a significant improvement on reconstruction results, except the magnitude of the reconstructed signal is not large enough compared to the original signal. The RMSE is 0.1797 for not using *k-NN* in Fig 4a. The RMSE is reduced to 0.0394 if using *k-NN* as in Fig. 4b. We can conclude that applying *k-NN* can have a significant influence on the quality of the reconstruction. The number of neighbors *k* will be tuned in Section IV-D1.



(a) Using all available signals.



(b) Using *k*-NN.

Figure 4. Reconstruction results using *k*-NN or not

4.1.3 Weight Factors

Among the k neighbors that we select, we weigh the importance of each neighbor by its cosine similarity to the missing signal. Before reconstruction, each signal of matrix M is multiplied by the corresponding cosine similarity weights. The reconstruction result considering weight factors is shown in Fig. 5. The RMSE score does not improve much, rising to 0.0366 from 0.0394 when there are no weight factors. The power of weight factors will not show up until normalization. We will go back to the effect of weight factors at the end of the normalization Section IV–B.

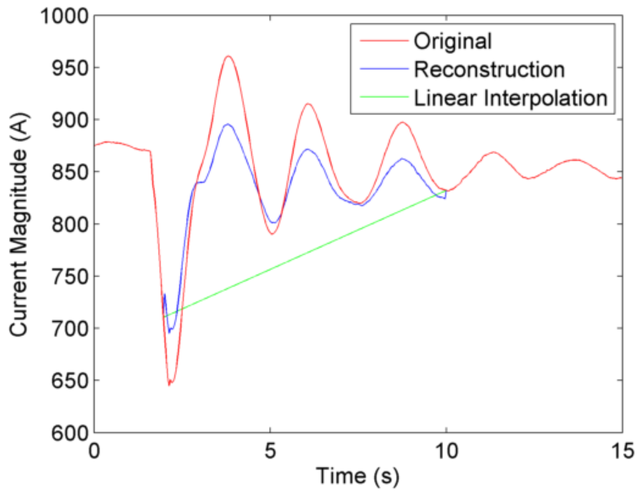
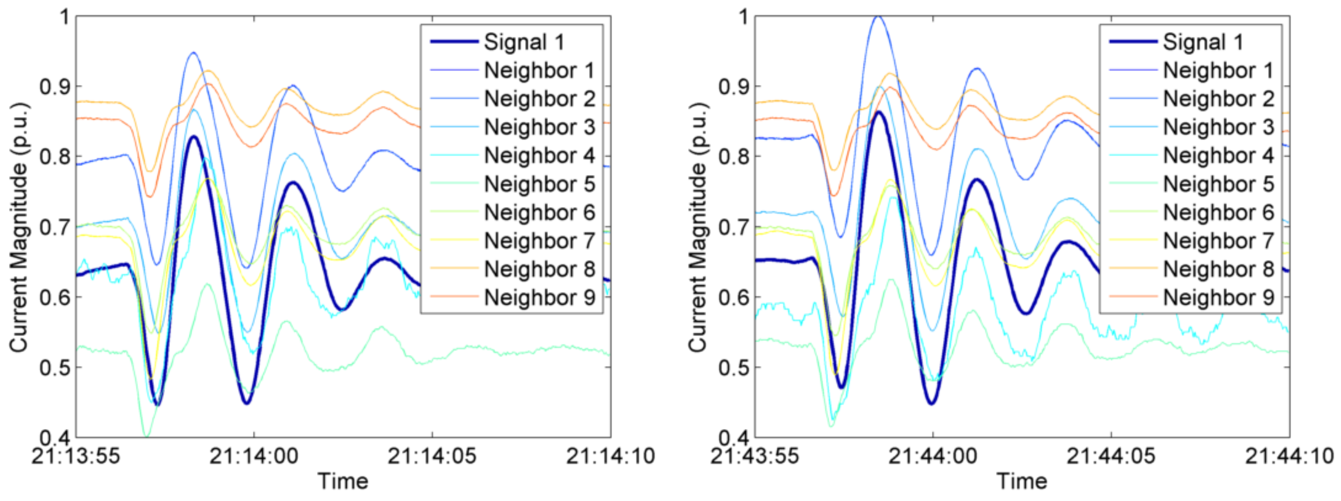


Figure 5. Reconstruction result using *k*-NN with weight factors.

4.1.4 Maintaining Neighbor Database

In practice, the above neighbor analysis has to be done automatically a priori. In real-time, there is no valid information from the missing signal to identify which are the neighbors of this signal. Think about two signals with 15 seconds data losses. The lost data is treated as 0, which does not impact the value of the dot product. Any other signals multiplying 0 would become 0, making all signals no difference. In this case, similarity weights are not trustworthy and will result in biased reconstructions.



(a) The event at around 21:13:55

(b) The event at around 21:43:55

Figure 6. Two events at a different time

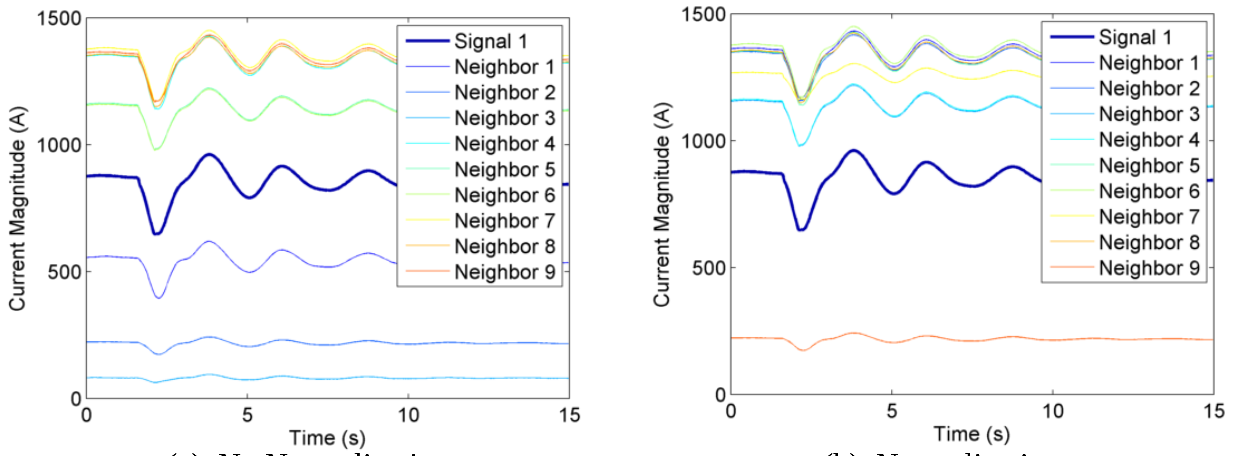
A common solution for these problems is to do similarity computation and neighbor selection beforehand when there are plenty of available measurements. And we maintain a database of sets of neighbors for each signal from previous events. For instance, in Fig. 6a, an event happened at around 21:13:55. The above analysis of similarity computation and neighbor selection is done using the event data recorded at this moment. A subset of neighboring signals is set aside for each signal in case some data is lost for that signal. Table III is the database of sets of neighbors that we maintain from historian events. And their respective cosine values are kept as similarity weights to specify the importance of each neighbor. When another event happened at around 21:43:55 as shown in Fig. 6b, we use the database of neighbors to reconstruct missing data. The two figures show that the behavior of each signal has little variance during two events because the statuses of the system are similar.

Table 3: Candidate neighbors and their cosine similarity

	Neighbor 1 (cosine)	**Neighbor 2 (cosine)**	**Neighbor 3 (cosine)**	...
Signal 134	161 (0.9669)	160 (0.9665)	51 (0.9609)	...
Signal 94	143 (0.9909)	128 (0.9881)	3 (0.9877)	...
Signal 75	69 (0.9509)	74 (0.7978)	68 (0.7535)	...
Signal 167	163 (0.9996)	169 (0.9993)	9 (0.9842)	...
Signal 158	253 (0.9988)	254 (0.9988)	174 (0.9975)	...
Signal 124	134 (0.9430)	161 (0.8936)	160 (0.8935)	...
⋮	⋮	⋮	⋮	⋮

4.2 Normalization

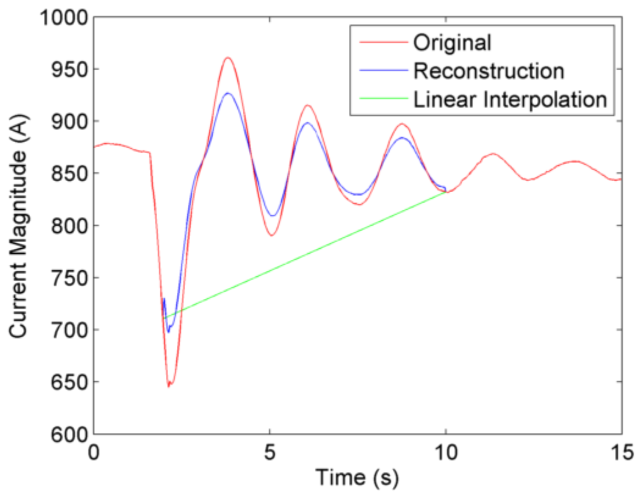
Normalization transforms each entry to positive or negative by comparing it to the mean value. As mentioned in Section III-B, normalization will affect cosine similarity in a good way. Fig. 7a plots the 9 neighbors that we used for reconstruction in the previous Section IV-A2. In measurement matrix, the 9 neighbors have original numbers as [24, 118, 187, 27, 23, 115, 116, 119, 83]. Normalization gives us a different neighbor set as expected in Fig. 7b with their original numbers as [83, 84, 115, 23, 27, 116, 145, 119, 118]. Some common neighbors appear in both neighbor sets. It is hard to say which neighbor set is better from the plots. So we test the neighbor sets and evaluate the reconstruction results.



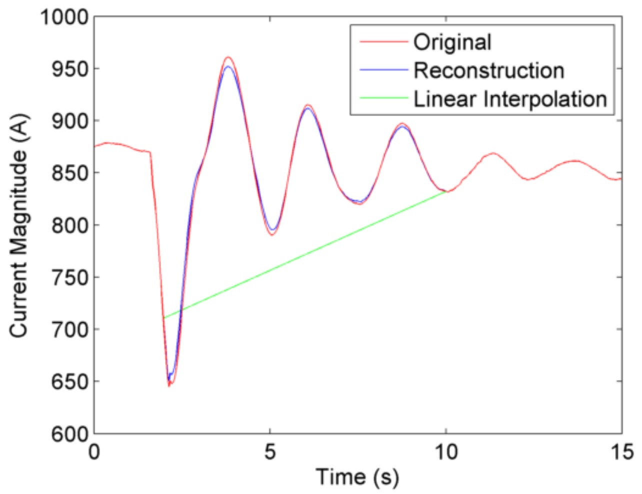
(a) No Normalization (b) Normalization

Figure 7. Neighbor sets with or without

If we run the reconstruction using normalized measurement matrix and the neighbor set given by this normalized matrix, the RMSE will improve to 0.0224, compared to 0.0394 of the one using neighbor set without normalization. The improvement is also reflected by the plots in Fig. 8a compared to Fig. 4b. The magnitude of the reconstructed signal is getting closer to the original values. If multiplying the normalized measurement matrix by weight factors to each signal, the reconstruction result will be improved significantly as shown in Fig. 8b. The RMSE reduced to 0.0060. So it is generally better to normalize the matrix first as preprocessing.



(a) Normalization only

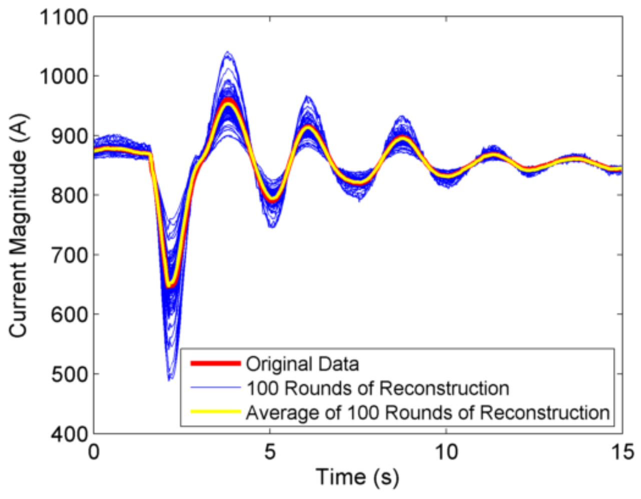


(b) Normalization and weight factors.

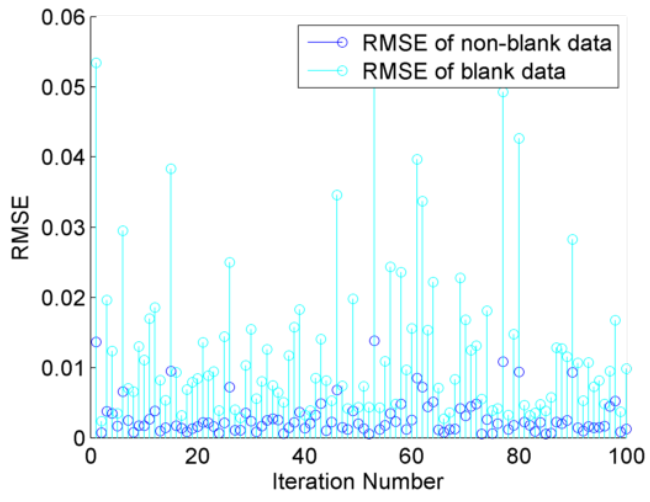
Figure 8. Reconstruction results with normalization and weight factors.

4.3 Optimization from Randomness

Monte Carlo method repeats the UV-decomposition with random starting points of U and V to search for the global optimal decomposition of a measurement matrix M. We run the algorithm for 100 independent rounds. Each running of the algorithm converges to a local optimum as the blue lines seen in Fig. 9a. We compute the RMSE of each round compared to true values (red line) and show the corresponding RMSE in Fig. 9b. Among 100 rounds of RMSE, the minimum error is 0.0024 and the maximum error is 0.0571. The average (yellow line) of 100 rounds has an RMSE of 0.0048. So generally, taking the average of multiply rounds is a better strategy than just one round that might lead to bad reconstruction result.



(a) 100 rounds of reconstruction.



(b) RMSE of 100 rounds

Figure 9. Optimization from randomness

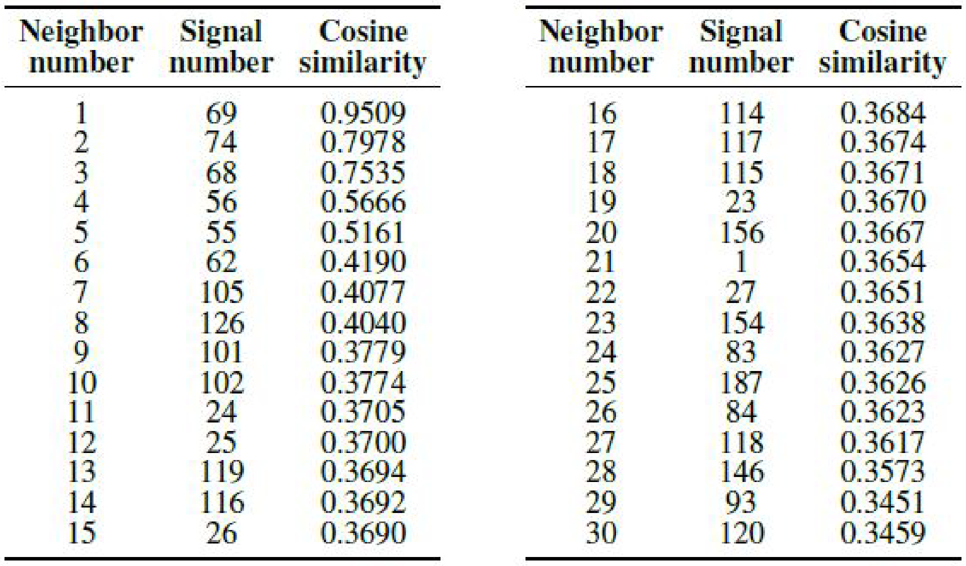
4.4 Other Hyper-Parameters Tuning

In this section, we study the effects of four hyper-parameters: number of nearest neighbors k, number of iterations I, dimension of the latent factor space d and regularization parameter λ.

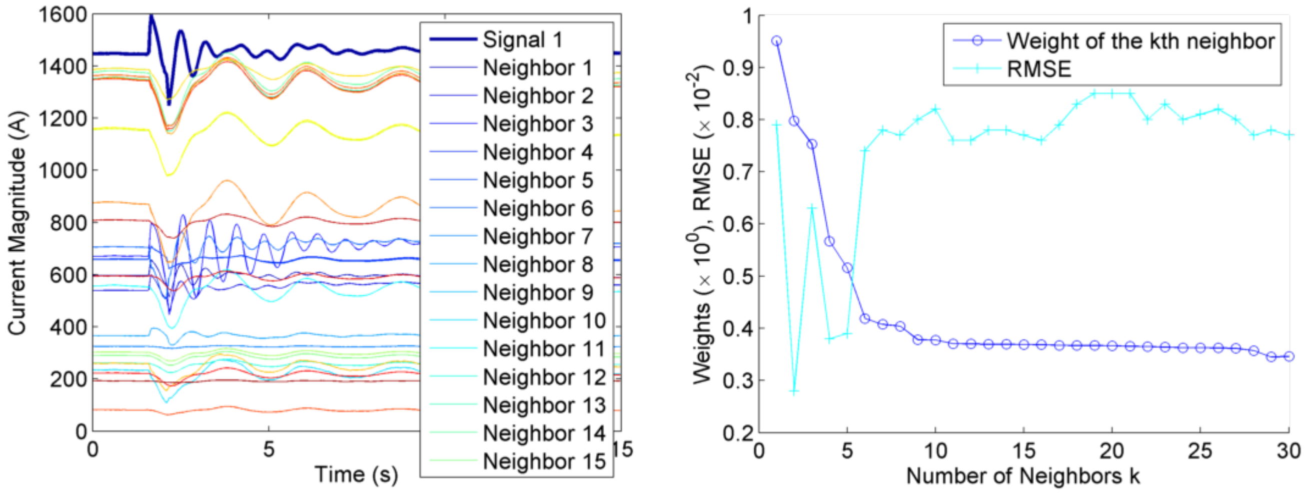
4.4.1 The Number of Nearest Neighbors *k*

The number of nearest signals whose values will be involved in the reconstruction should be chosen carefully. If k is too small, the algorithm will be sensitive to noise signals. But if k is too large, the neighborhood might include too many signals that are not so close. Although a number of neighbors between 20 to 50 is most often described in the literature, see e.g. [27], [28] the optimal value of k need to be tuned considering the number of available signals in the system. We pick a signal and erase 8 seconds data in it. A list of its top 30 candidate neighbors and their cosine similarities with the missing signal is shown in Table IV. Fig. 10a shows the plots of the missing signal and its top 30 neighbors. We reconstruct the missing signal with increased number of neighbors k. Fig. 10b shows the RMSE under each reconstruction with k ranging from 1 to 30.

Table 4: Candidate neighbors and their cosine similarity



Neighbor number	Signal number	Cosine similarity
1	69	0.9509
2	74	0.7978
3	68	0.7535
4	56	0.5666
5	55	0.5161
6	62	0.4190
7	105	0.4077
8	126	0.4040
9	101	0.3779
10	102	0.3774
11	24	0.3705
12	25	0.3700
13	119	0.3694
14	116	0.3692
15	26	0.3690
16	114	0.3684
17	117	0.3674
18	115	0.3671
19	23	0.3670
20	156	0.3667
21	1	0.3654
22	27	0.3651
23	154	0.3638
24	83	0.3627
25	187	0.3626
26	84	0.3623
27	118	0.3617
28	146	0.3573
29	93	0.3451
30	120	0.3459



(a) Plots of the missing signal and its top 30 neighbors.

(b) Performance comparisons with Different *k*.

Figure 10. Parameter tuning for *k*.

As shown in Fig. 10b, the reconstruction accuracy observed for increasing values of k typically behaves differently in three phases. When the number of neighbors is restricted by using a small k (e.g., $k \leq 5$), the reconstruction accuracy is not stable. As k increases, more neighbors contribute to the reconstruction and the variance introduced by individual neighbors is averaged out. As a result, the reconstruction accuracy becomes stable. Finally, the accuracy usually drops when too many neighbors are used in the reconstruction (e.g., $k \geq 50$ in Fig. 4a), because the few strong local relations are "diluted" by the many weak ones. Since the RMSE when $k = 6 - 30$ is pretty stable, we prefer to choose a small number of high-confidence neighbors, as well as saving some computational time. So we select 9 neighbors in the following analysis for other signals.

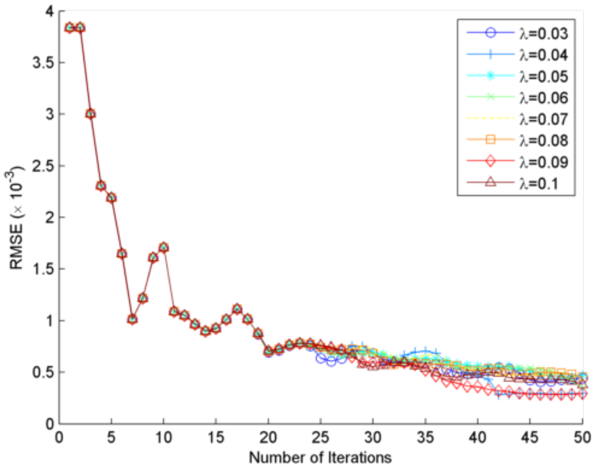
4.4.2 The Number of Iterations *I*

We fix $d = 5$,run 1 to 50 iterations of the algorithm with different λ. As Fig. 11a shows, with fixed d and λ each iteration improves the RMSE score of the reconstructed data, and it converges after about 40 iterations. That is, after 40 iterations, the RMSE score still improves but only less than 1 bps for each iteration afterward. Different λ values give different final RMSE score. The best performer is $\lambda = 0.04,\ I = 42$), giving an RMSE of 2.81×10^{-4}.

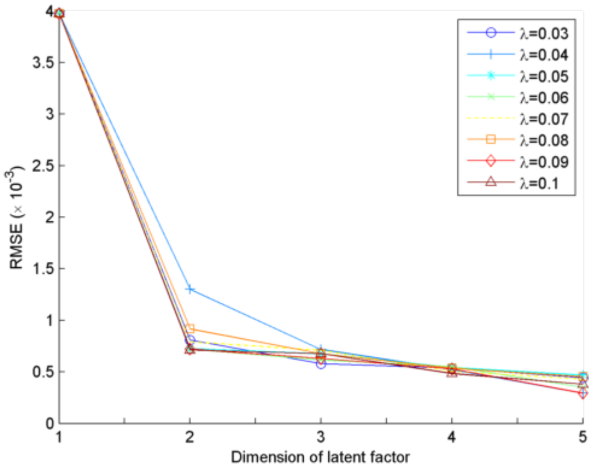
4.4.3 The Dimension of the Latent Factor Space *d*

Fig. 11b shows the performance of the algorithm with fixed number of iterations $I = 50$ and varying λ value and number of latent factors ($d$ranges from 1 to 5). We can tell that the RMSE monotonically decreases with larger d, even though the improvement diminishes gradually. The best performer is($\lambda = 0.04, d = 5$), giving an RMSE of 2.92×10^{-4}.

Thus, the optimal values of hyper-parameters are then determined. When implementing the algorithm on field data in Section V, we use the following values for hyper-parameters: $I = 42, d = 5, \lambda = 0.04, k = 9$.



(a) Fix $d = 5$, vary λ and I



(b) Fix $I = 50$, vary λ and d

Figure 11. Performance comparisons with different d, λ and I.

5. Case Studies

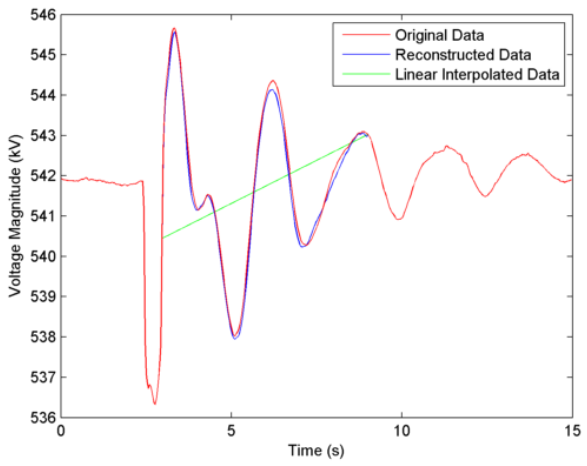
In this section, the proposed technique is applied on two different set of field data in the Western Interconnection. In the first set of data, all the PMU channels are available and missing parts of signals are intentionally designed. By doing so, the efficiency of the proposed method is shown by comparing results with original signals.

5.1 Results and Discussion

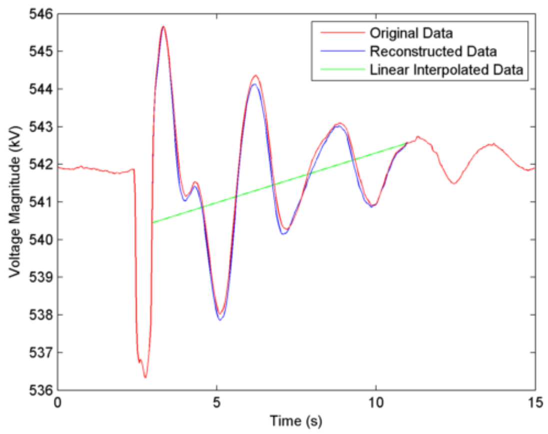
In this case, we design two scenarios by erasing consecutive measurements: 1) one signal has data losses for different time length, and 2) two or more signals have data losses. The reconstruction uses collaborative filtering algorithm with the optimized parameter set obtained in Section IV. Each scenario enables comparative research on performance between reconstructions and linear interpolations. In addition to comparing RMSE score, we compare the modal information, including frequency, damping ratio, total energy, and mode shape, from an event analysis application [29], using reconstructions and linear interpolations as input data.

5.1.1 One Signal Has Data Losses for Different Time Length

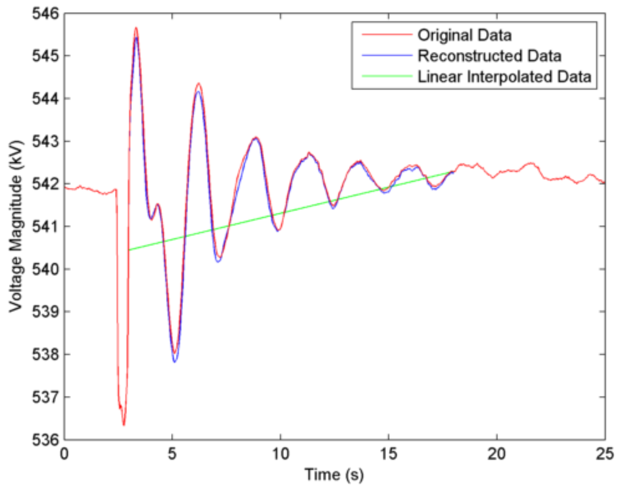
First, we conduct experiments with different length of data losses in a voltage magnitude signal. We erase consecutive measurements for 6 seconds, 8 seconds, 15 seconds, and 20 seconds in the signal. Fig. 12 and Table IV show the comparisons between collaborative filtering reconstruction and linear interpolation. Generally, the reconstructed data has higher accuracy and can give us modal information comparable to those from true values; while linear interpolation has larger errors, leading to incorrect damping and energy information. The mode shape plots are shown in Fig. 13. Here we only plot the mode shape of ten good signals including the bad/reconstructed signal for demo purpose. We can see that the linear interpolation destroys the mode shape component of the bad signal, while reconstruction gets it mostly right. We would have no information about mode shape magnitude and phase of this bad signal if we had discarded it. In 20 seconds data loss case, almost the entire event data is missing. We still obtain an RMSE of 6.64×10^{-4} and the modal information is acceptable.



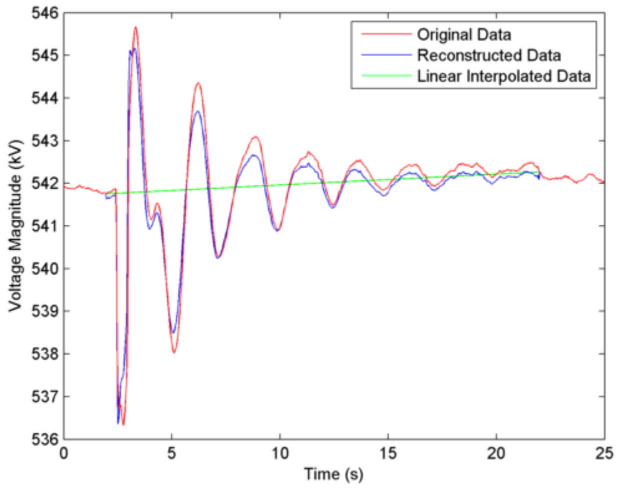
(a) 6s data losses.



(b) 8s data losses.



(c) 15s data losses.

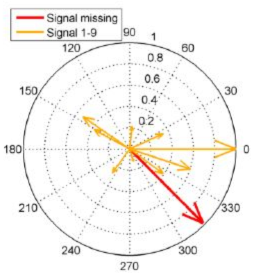


(d) 20s data losses.

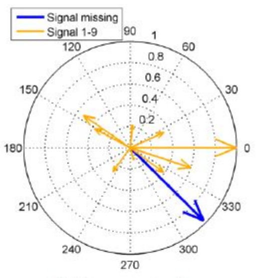
Figure 12. Comparisons between collaborative filtering reconstruction and linear interpolation for different length of data losses.

5.1.2 One Signal Has Data Losses for Different Time Length

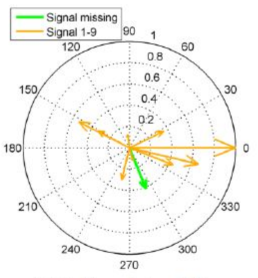
Another study is instrumented in this section that focuses on multiple missing signals. Successful reconstruction depends on whether the neighbors are similar enough to the missing signals. We take current magnitude signals as test data because they have phase lags between them at event time. For example, the two missing signals in Fig. 14a have phase lag between them. If we choose the signals whose trend of changing is completely contrary with the missing signals, it will result in poor reconstruction. However, the collaborative filtering preserves the phase lag in the missing signals by using their respective neighbors to reconstruct the signals.



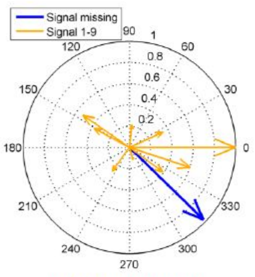
(a) Original.



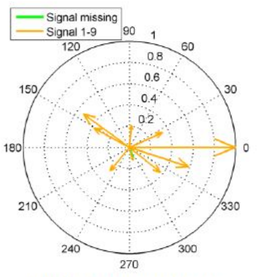
(b) 6s reconstruction.



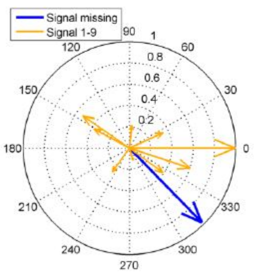
(c) 6s linear interpolation.



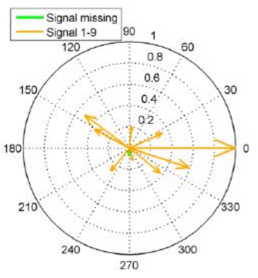
(d) 8s reconstruction.



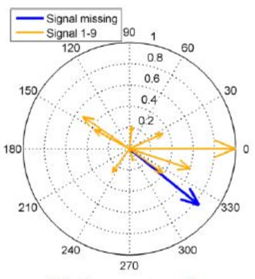
(e) 8s linear interpolation.



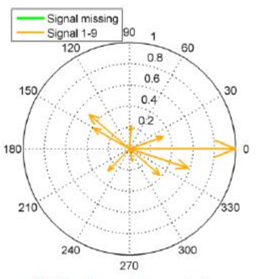
(f) 15s reconstruction.



(g) 15s linear interpolation.



(h) 20s reconstruction.



(i) 20s linear interpolation.

Figure 13. Comparisons of modal information between collaborative filtering reconstruction and linear interpolation for different length of data losses.

Table 5: Comparisons of modal information between collaborative filtering reconstruction and linear interpolation for different data losses

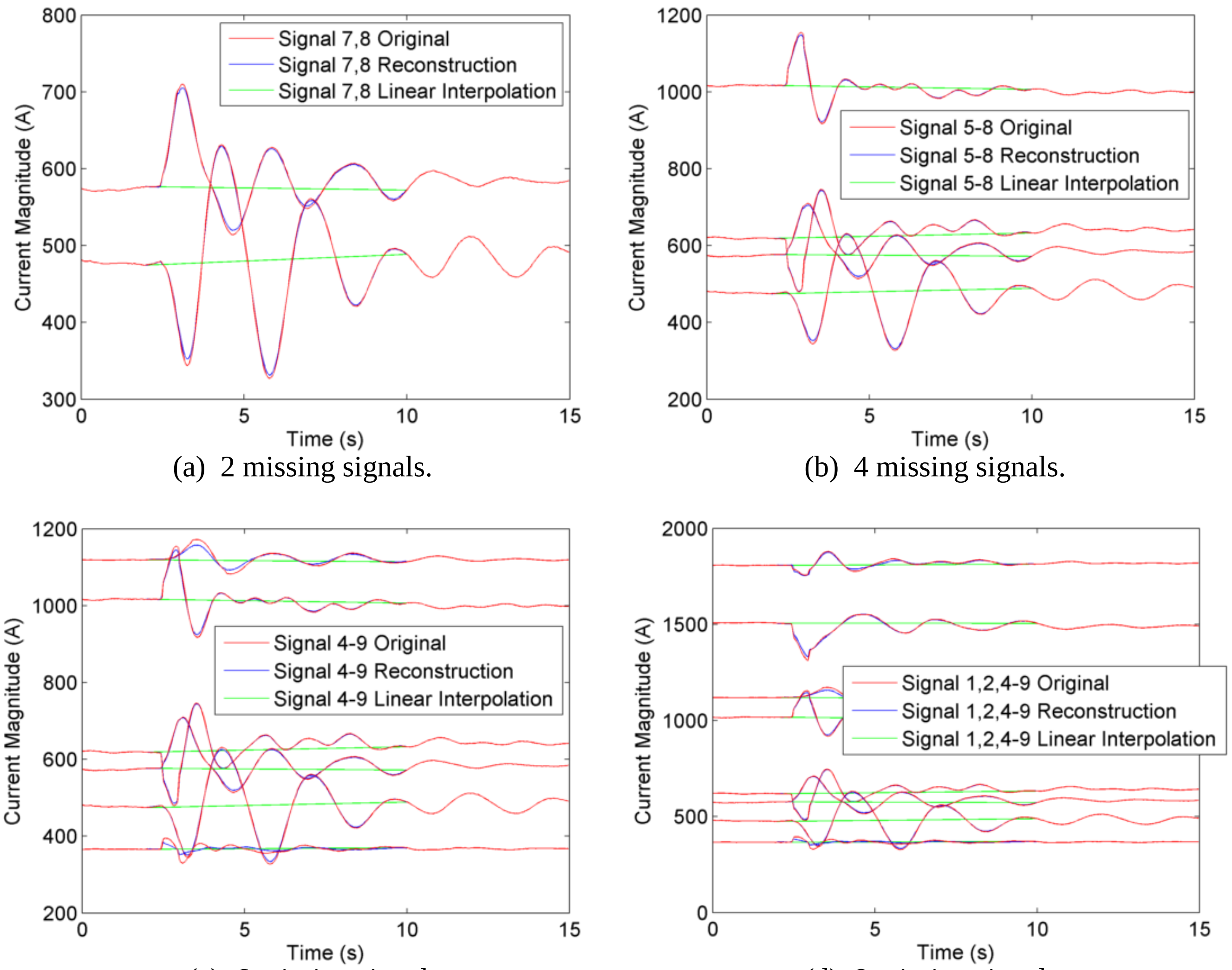
The length of data losses	RMSE ($\times 10^{-4}$)	Frequency (Hz)	Damping (%)	Total Energy
Original data	-	0.40	8.03	4.38
6s reconstruction	3.57	0.40	8.03	4.38
6s linear interpolation	40.40	0.41	7.22	1.90
8s reconstruction	3.79	0.40	8.03	4.39
8s linear interpolation	35.96	0.40	7.83	3.66
15s reconstruction	2.98	0.40	8.02	4.38
15s linear interpolation	28.81	0.40	7.94	3.87
20s reconstruction	6.64	0.40	7.99	4.33
20s linear interpolation	26.11	0.40	7.93	3.92

Table 6: Comparisons of modal information between collaborative filtering reconstruction and linear interpolation for multiple missing signals

The number of signals with data losses	$\overline{\text{RMSE}}$ ($\times 10^{-3}$)	Frequency (Hz)	Damping (%)	Total Energy
Original data	-	0.40	10.63	5.82
2 signals reconstruction	4.53	0.40	10.45	5.44
2 signals linear interpolation	118.59	0.41	3.99	0.83
4 signals reconstruction	3.25	0.40	10.40	5.34
4 signals linear interpolation	86.73	0.41	3.46	0.63
6 signals reconstruction	6.68	0.40	9.43	3.157
6 signals linear interpolation	65.26	0.38	2.98	0.71
8 signals reconstruction	5.78	0.40	8.36	3.17
8 signals linear interpolation	54.99	0.37	4.45	1.40

We erase 8 seconds data in 2, 4, 6, and 8 signals and study how the results of collaborative filtering reconstruction and linear interpolation affect the modal estimation. Fig. 14 and Table VI show the comparisons, where collaborative filtering does better regarding accuracy. The number of missing signals does not severely affect the RMSE and modal analysis of collaborative filtering. The reason is that no matter how many signals are missing; we can still find a few available neighbors to make the reconstruction. In contrast, the result sets of linear interpolation turn out to be very bad, and even worse with increasing number of missing signals. The linear interpolations result in damping ratios below 5% which means the system is poorly-damped. However, the original system is welldamped with a damping ratio of 10.63%. The mode shape of the missing signal, as shown in Fig. 15 is preserved by reconstruction while it is destroyed by linear interpolation.

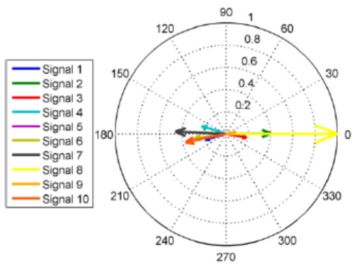
5.2 The Second Case



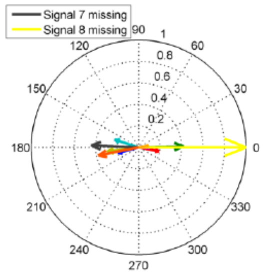
(a) 2 missing signals. (b) 4 missing signals.

(c) 6 missing signals. (d) 8 missing signals.

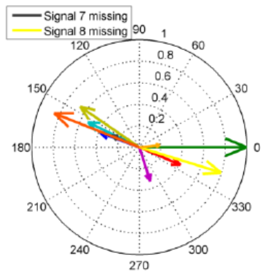
Figure 14. Comparisons between collaborative filtering reconstruction and linear interpolation for multiple missing signals.



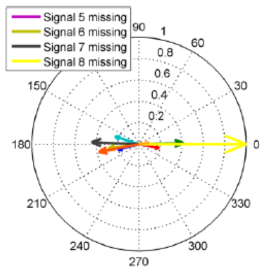
(a) Original.



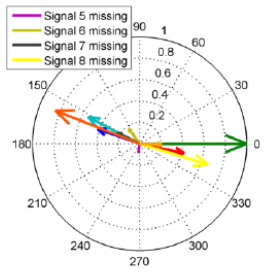
(b) 2 signals reconstruction.



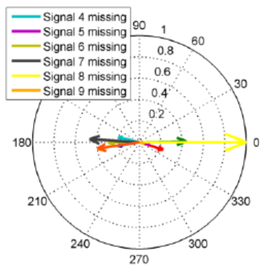
(c) 2 signals linear interpolation.



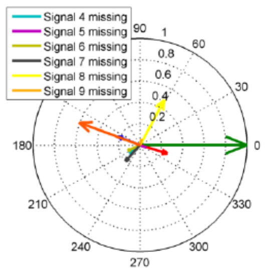
(d) 4 signals reconstruction.



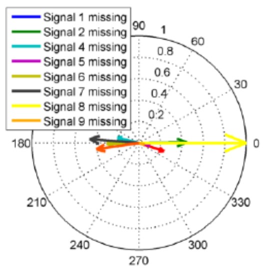
(e) 4 signals linear interpolation.



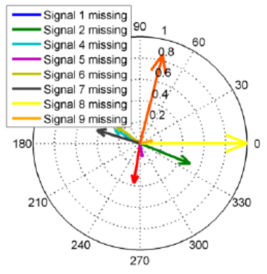
(f) 6 signals reconstruction.



(g) 6 signals linear interpolation.



(h) 8 signals reconstruction.



(i) 8 signals linear interpolation.

Figure 15. Comparisons of mode shapes between collaborative filtering reconstruction and linear interpolation for multiple missing signals.

6. Conclusions

In this project, available measurements from recent history and nearby signals are used as cross-references to reconstruct missing data. A measurement matrix M contains measurements of signals at certain time instants, while some entries are unknown because of data loss. The essential problem is to reconstruct the values of the unknown entries based on the values of the known entries. Collaborative filtering algorithms are employed to identify neighboring signals and make predictions on unknown entries.

The algorithm finds two long, thin matrices U and V, whose product is an approximation to the given measurement matrix M. Since the matrix product UV gives values for all signal-time pairs, those values can be used to predict the value of a blank in the measurement matrix. Gradient descent method corrects the errors between non-blank true values and predicted values after each iteration until convergence.

To improve the accuracy of reconstruction, three strategies are implemented: 1) normalizing scales of the matrix, 2) identifying neighbors and assigning similarity weights, and 3) selecting reasonable results from multiple random trials. The effectiveness of strategies is tested using field PMU data. Experiments results show the algorithm is able to achieve accurate reconstructions with appropriate choice of parameters. The successful reconstruction of large-scale PMU data will enable real-time PMU application and significantly improve the accuracy and trustworthiness of the output.

Future works could be on implementations of different types of collaborative filtering algorithms, and on automatic hyper-parameter tuning to obtain the optimal parameter sets, and on real-time platform development.

References

[1] J. De La Ree, V. Centeno, J. S. Thorp, and A. G. Phadke, “Synchronized phasor measurement applications in power systems,” IEEE Transactions on Smart Grid, vol. 1, no. 1, pp. 20–27, 2010.

[2] A. Bose, “Smart transmission grid applications and their supporting infrastructure,” IEEE Transactions on Smart Grid, vol. 1, no. 1, pp. 11–19, 2010.

[3] J. Ning, X. Pan, and V. Venkatasubramanian, “Oscillation modal analysis from ambient synchrophasor data using distributed frequency domain optimization,” IEEE Transactions on power systems, vol. 28, no. 2, pp. 1960–1968, 2013.

[4] J. Tang, J. Liu, F. Ponci, and A. Monti, “Adaptive load shedding based on combined frequency and voltage stability assessment using synchrophasor measurements,” IEEE Transactions on power systems, vol. 28.

[5] L. E. Miller, A. Silverstein, D. Anand, A. Goldstein, Y. Makarov, F. Tuffner, and K. Jones, “Pmu data quality: A framework for the attributes of pmu data quality and a methodology for examining data quality impacts to synchrophasor applications,” NASPI, Tech. Rep. PNNL 26313 / NASPI-2017-TR-002, March 2017. [Online]. Available: https://www.naspi.org/node/352

[6] M. Wu and L. Xie, “Online detection of low-quality synchrophasor measurements: A data driven approach,” IEEE Transactions on Power Systems, vol. PP, pp. 1–1, Nov 2016.

[7] M. Farrokhifard, M. Hatami, and M. Parniani, “Novel approaches for online modal estimation of power systems using pmus data contaminated with outliers,” Electric Power Systems Research, vol. PP, pp. 74 – 84, Nov 2015.

[8] C. I. Contributors, “Ip-1 iso uses synchrophasor data for grid operations, control, analysis and modeling for smart grid roadmap,” California ISO, CA, Tech. Rep., Oct 2010.

[9] V. Miranda, J. Krstulovic, H. Keko, C. Moreira, and J. Pereira, “Reconstructing missing data in state estimation with autoencoders,” IEEE Transactions on Power Systems, vol. 27, pp. 604–611, Dec 2011.

[10] F. Adinolfi, F. D’Agostino, A. Morini, M. Saviozzi, and F. Silvestro, “Pseudo-measurements modeling using neural network and fourier decomposition for distribution state estimation,” in Innovative Smart Grid Technologies Conference Europe (ISGT-Europe), 2014 IEEE PES. IEEE, 2014, pp. 1–6.

[11] M. Wang, J. H. Chow, P. Gao, X. T. Jiang, Y. Xia, S. G. Ghiocel, B. Fardanesh, G. Stefopolous, Y. Kokai, N. Saito et al., “A low-rank matrix approach for the analysis of large amounts of power system synchrophasor data,” in System Sciences (HICSS), 2015 48th Hawaii International Conference on. IEEE, 2015, pp. 2637–2644.

[12] N. Dahal, R. L. King, and V. Madani, “Online dimension reduction of synchrophasor data,” in Transmission and Distribution Conference and Exposition (T&D), 2012 IEEE PES. IEEE, 2012, pp. 1–7.

[13] Y. Chen, L. Xie, and P. Kumar, “Dimensionality reduction and early event detection using online synchrophasor data,” in Power and Energy Society General Meeting (PES), 2013 IEEE. IEEE, 2013, pp. 1–5.

[14] J.-F. Cai, E. J. Cand`es, and Z. Shen, “A singular value thresholding algorithm for matrix completion,” SIAM Journal on Optimization, vol. 20, no. 4, pp. 1956–1982, 2010.

[15] R. Meka, P. Jain, and I. S. Dhillon, “Matrix completion from powerlaw distributed samples,” in Advances in neural information processing systems, 2009, pp. 1258–1266.

[16] P. Gao, S. G. Ghiocel, and J. H. Chow, “Modeless reconstruction of missing synchrophasor measurements,” in PES General Meeting— Conference & Exposition, 2014 IEEE. IEEE, 2014, pp. 1–5.

[17] (2009) The netflix prize. [Online]. Available: http://www.netflixprize.com

[18] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds., Recommender Systems Handbook. New York: Springer., 2011.

[19] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan, “Large-scale parallel collaborative filtering for the netflix prize,” in International Conference on Algorithmic Applications in Management. Springer, 2008, pp. 337– 348.

[20] J. Leskovec, A. Rajaraman, and J. D. Ullman, Mining of Massive Datasets. United Kingdom: Cambridge University Press, 2014.

[21] G. Tak´acs, I. Pil´aszy, B. N´emeth, and D. Tikk, “Matrix factorization and neighbor based algorithms for the netflix prize problem,” in Proceedings of the 2008 ACM conference on Recommender systems. ACM, 2008, pp. 267–274.

[22] G. Salton and M. J. McGill, “Introduction to modern information retrieval,” 1986.

[23] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-based collaborative filtering recommendation algorithms,” in Proceedings of the 10th international conference on World Wide Web. ACM, 2001, pp. 285–295.

[24] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens, “Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation,” IEEE Transactions on knowledge and data engineering, vol. 19, no. 3, 2007.

[25] J. S. Breese, D. Heckerman, and C. Kadie, “Empirical analysis of predictive algorithms for collaborative filtering,” in Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence. Morgan Kaufmann Publishers Inc., 1998, pp. 43–52.

[26] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, “Grouplens: an open architecture for collaborative filtering of netnews,” in Proceedings of the 1994 ACM conference on Computer supported cooperative work. ACM, 1994, pp. 175–186.

[27] N. Good, J. B. Schafer, J. A. Konstan, A. Borchers, B. Sarwar, J. Herlocker, J. Riedl et al., “Combining collaborative filtering with personal agents for better recommendations,” in AAAI/IAAI, 1999, pp. 439–446.

[28] J. Herlocker, J. A. Konstan, and J. Riedl, “An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms,” Information retrieval, vol. 5, no. 4, pp. 287–310, 2002.

[29] G. Liu, J. Ning, Z. Tashman, V. M. Venkatasubramanian, and P. Trachian, “Oscillation monitoring system using synchrophasors,” in Power and Energy Society General Meeting, 2012 IEEE. IEEE, 2012, pp. 1–8.