Dissertation

Extraction, Characterization and Modeling of Network Data Features - A Compressive Sensing and Robust PCA based Approach

Submitted by

Vidarshana W. Bandara

Department of Electrical and Computer Engineering

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Spring 2015

Abstract

Extraction, Characterization and Modeling of Network Data Features - A Compressive Sensing and Robust PCA based Approach

Designing computer networks resilient to failures, excessive loads, and attacks and then monitoring them are challenged by a range of issues inherent to networks. These limitations include inability to effectively characterize anomalous and baseline behaviors, access restrictions to place probes, and measurement limitations due to load limits. We develop a suite of solutions for extracting network data given the accessibility and load limitations of networks, and for characterizing interesting behaviors that capture the features useful for designing and monitoring networks. To the network monitoring and data extraction end, we build solutions for (1) retrieving interesting network data such as anomalies, with a limited amount of instrumentation, e.g., at the network periphery, (2) reconstructing a description of the network from a limited number of measurements, with different approaches suitable for computer networks and sensor networks. To the network data characterization end, we develop solutions for (1) extracting interesting features of data, such as anomalies and baselines, and (2) concisely and accurately modeling these features. These four classes of solutions build up the contributions of this thesis, which include: (1) adaptive compressive sensing algorithms for network monitoring, (2) spatiotemporal modeling of network traffic anomalies, (3) modeling and extracting network traffic baselines using Robust PCA, (4) Compressive Sensing (CS) based data recovery for phenomena discovery, (5) Wavelet based recovery with applications to plume tracking in sensor networks, (6) Subtle pattern detection algorithm with usage in hardware trojan detection, and (7) TCP/IP filter for extracting features with deployments in network attack detection.

We present multiple algorithms that adaptively perform CS to resolve network tomographic measurements and rapidly localize anomalies. These algorithms are scalable to large networks, operable with probes placed at arbitrary locations, and guide measurements using advanced IP options. The adaptive compressive sensing algorithms demonstrat over 99% detection rates and less than 1% false positive rates in realistic test environments, for networks reaching over 10,000 links. These require a limited number of monitoring probes residing at the edge of the network and are capable of localizing faults with orders of magnitudes fewer measurements than traditional approaches. Thus, we overcome the accessibility restrictions in networks and also provide an efficient method to retrieve critical information from networks.

Due to the sporadic behavior, modeling spatiotemporal characteristics of network traffic anomalies has been a challenge. The solution we developed models different spatial and temporal properties of anomalies and integrate them into a single model. The combined anomaly model captures statistical behaviors of anomalies as they propagate through nodes and subnets. Calibrating the model requires only local measurements. However it is capable of capturing the global anomaly behaviors. This characterization enables reproducing statistically similar network anomalies. By incorporating such characterization, traffic generators can produce realistic behaviors containing realistic anomalies. Based on the anomaly model, a real-time anomaly monitoring system and a real-time parameter learning system are also proposed. The model concisely and hierarchically characterizes anomaly behaviors of networks, which so far was not possible. Moreover, the model provides vital information for designing robust networks.

Our experiments reveal that Robust PCA achieves recovery over much wider ranges of sparsity and rank, than the published sufficient conditions suggest. These findings enable the use of Robust PCA for separating features of network data. We employ Robust PCA and Fourier methods for separating baseline and anomalous components. Further, these methods are made into real-time filters for baseline extraction and modeling. The baseline extraction algorithm is used in anomaly detection and compared against a number of existing methods to establish its fidelity. Extracted baselines are modeled using two types of patterns: one for link level behaviors and another for network level behaviors. Such characterizations are crucial for network design and provisioning.

Applications of the work done in this thesis go beyond the scope of infrastructure networks. For example, we derive theoretical recovery bounds of CS for any pairing of sampling and orthogonality measures. This work leads to cost effective phenomena discovery in sensor networks, where every node in the network becomes aware of the overall network state. Phenomena awareness experiments implemented with random walk sampling indicate superior energy savings and comparable accuracy to classical uniform sampling. Random walk measurements provide a practical means of gathering measurements, while uniform sampling is costly to implement. When applied with CS, phenomena awareness provides a practical and efficient approach to build a complete picture.

We developed an energy efficient approach that require fewer instrumentation using wavelet transforms. This work has applications in chemical plume tracking. It achieves an error of less than 7% with 25% of measurement points. This data reconstruction scheme saves energy and extends longevity of energy starved sensor network by a factor of five in average. A second improved solution is proposed by combining matrix completion and CS. An algorithm for detecting subtle patterns in an array of time-series is also developed. It is capable of extracting interesting but obscure features from data. The subtle pattern extraction algorithm lay the pathway for detecting hardware trojans whose existence makes only a slight impact on the impedance measurements. To extract interesting features from live network data, we develop a TCP/IP feature filter and successfully use it for detecting Neptune, Smurf, IP and port sweep, and Ping of death attacks. These approaches provide different means of extracting data and features for a range of applications, which extend beyond the scope of networks.

The proposed solutions are tested on either real-world or realistic data in order to establish performance and accuracy. To obtain real data, we develop tools to extract data from planet-lab infrastructure and other networks. To construct realistic data, we use established data models such as the Gilbert-Elliot model for packet loss and heavy-tailed distributions for delay, then calibrated these models using actual measurements. Further, commonly used synthesis tools such as IGen are used to simulate large scale realistic networks. These test environments help establish the fidelity of the developed solutions. This research provides effective, practical and accurate solutions for network monitoring, data extraction and modeling.

Acknowledgements

I am extremely grateful for the constant guidance, patience and opportunity Prof. Anura Jayasumana extended to me. This work would not have been a possibility without his most kind support. I would also like to recall all the kind support and encouragements I received from Prof. Ali Pezeshki. Lessons and practices I learned from Prof. Pezeshki were life changing. I remain utterly thankful for the opportunity to work with Prof. Louis Scharf, whose constant guidance was a blessing. I was also fortunate to be continually guided by Prof. Randy Paffenroth, who exposed me to a wide range of disciplines. I am also extremely grateful to my committee Prof. Rockey Luo and Prof. Indrajit Ray for guiding and helping me throughout the years. I am very thankful for the kind support and help I received from Mr. Rick Whitner, Dr. Philip DuToit, Dr. Ryan Nong, and Mr. Kenneth Parker.

I am grateful for the support received from NSF grants CNS-0720889, CCF-0916314, CCF-1018472 and ERC program with award number 0313747, for in part enabling this thesis. Contributions and guidance received from the JDSU Advanced Technology Program helped produce some of the most interesting outcomes. The support received from Air Force Office of Scientific Research under STTR contract FA9550-10-C-0090 facilitated a significant part of this thesis.

I would also like to express my deepest gratitude to my friends and colleagues Yang Zhang, Ramin Zahedi, Dulanjalie Ganegedara, and Dilum Bandara, who supported and constantly encouraged me. I am blessed with the most supportive parents to whom I would like to dedicate this thesis and the most supportive brother and the sister. Finally, I dedicate a very special thank to Prof. Jayantha Herath for opening the doors to my future and for being a constant support.

Table of Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

Computer networks connect resources and deliver information to build information systems that range from Personal Area Networks (PANs) to internets. Each category of networks has their own class of problems. For example, networks of the scales ranging from Local Area Networks (LANs) to the Internet would be concerned about the throughput, latency, reliability and security; whereas wireless sensor networks would be more concerned about the energy efficiency. Therefore, when designing networks, their respective concerns should be taken into account. Further, during operation, the networks should be monitored to assure a healthy operation and to learn when faults occur. The premise of this thesis revolves around monitoring networks, retrieving network information, characterizing and modeling network behaviors. This study addresses issues related to robust network designing and efficient network monitoring.

Network data are rich in various features. External to the data content, physical features such as throughput and delay reveal information about the network performance. By digesting the data content, much about the communication can be revealed. At a higher level, packet headers can be conveniently extracted by filters, which provide a range of information about the communication such as the origin, destination and the protocol used. By performing deep packet inspection more intricate information about the communication can be learned. Information extracted from features of different levels can be used to diagnose network issues of a few categories. Most operational issues such as link failures can be diagnosed by observing physical features as the throughput. Tracking traffic flows can be done by observing the packet headers. Networks may observe unusual traffic behaviors in three main scenarios: (1) excessive but legitimate user behaviors such as flash crowds, (2) network failures, and (3) attacks. The network administrators have to identify these scenarios distinctly as they have to react in different manners to each of the cases. If the unusual behavior is due to legitimate user behaviors, the administrators should take action to throttle traffic, keep the system alive, and continue to serve requests. Whereas during an attack, the administrators should first take action to protect the system and prevent further damage. They may decide to trace data to identify and localize the perpetrators. All these scenarios require the administrators characterizing the nature of the unusual behavior. Methods based on deep packet inspection are likely to provide more information. However, deep packet inspection is a costly operation and may not be possible to perform against a large incoming traffic volume. Thus, fast and light-weight techniques based on packet header characterization are more attractive. Further, packet header extraction is built into most network hardware making methods based on packets headers cost-effective. We focus on building a feature extraction and correlation scheme based on information revealed by packet headers. We apply this method to detect a number different network attacks.

Modeling network traffic behaviors has been of interest since the early days of computer networks. The initial approaches treated computer networks similar to telephony networks and used Poisson models to describe traffic behaviors. As most communications on initial computer networks were via dial-up connections, these models were quite valid. But as broadband communication took prominence, validity of Poisson model decayed. Although a group of researchers believe that Poisson model is still valid at an Internet backbone level. Over time, traffic modeling evolved into heavy-tailed distributions as most network parameters demonstrated occurrence of large values quite frequently. Furthermore, network data showed long-range dependence [3]. Some recent approaches view network data as a fractal process and continue use heavy-tailed distributions to describe them. The most recent approaches take truncation into account, as in [4]. For example, if the delay of a transmission surpasses a certain threshold, a retransmission occurs. This leads to occurring large values but with a maximum limit.

Another one of our interests is in identifying and characterizing the important and interesting properties of data. One of the techniques we employ is Robust Principal Component Analysis (RPCA) [5, 6]. It additively separates a data matrix into a low-rank matrix and a sparse matrix. A number of practical scenarios exists where data matrices are a sum of low-rank and sparse matrices. Video surveillance, face recognition, latent signal indexing and collaborative filtering are a example few applications. If the data matrix was in fact a sum of a "true" low-rank matrix, i.e. a matrix which is not also sparse, and a "true" sparse matrix, i.e. a matrix that is not also low-rank, then the output matrices match the original low-rank and sparse matrices. As low-rank-ness and sparse-ness can coexist, the conditions to assure true low-rank-ness and true sparse-ness prevent ambiguous recoveries. The conditions laid down in literature however are either quite conservative or impractical to calculate. Thus, we could not use them to predict the recoverability of a matrix pair. As an alternative, we empirically establish the recoverability of RPCA and as well a cross validation principle to determine whether a decomposition was actually a recovery. We study a range of matrix constructions and example matrices from real-world applications to establish our claims.

One of the primary focuses of this thesis is extracting and modeling network specific behaviors. We identify two major component network data: (1) baselines and (2) anomalies. However extracting these components are challenging as network data carry a vast amount randomness that mask these properties. Baseline behavior describes the expected behavior under regular operation. Although it is a well conceptualized property, mathematically describing a baseline is difficult. Thus, we use a few properties that are admissible to a human network monitor's point of view and amenable to mathematical representation to describe baselines. Baselines are expected to be common across the dataset. They are also expected to be the prominent component of the data. Further, baseline behaviors are expected to re-occur at regular intervals. We employ PCA and Fourier analysis to extract baseline behaviors, as they capture these properties. We also use RPCA to cleanse data of any major anomalies. Anomalies on the other hand show a drastically different behavior. Their sporadic occurrences with unpredictable volumes and durations make it difficult to model anomalies with typical random processes. Thus, we device a new approach where features of anomalies are separated and modeled individually. This leads to a concise description of anomalies that captures the statistical properties of anomalies accurately. The proposed method enables describing anomaly behaviors hierarchically. Further, based on this new model, we propose two real-time applications: (1) a technique to communicate model parameters efficiently track anomaly behaviors of regions on the fly, and (2) a technique for intelligent real-time anomaly tracking.

As a maintenance service, network monitoring should be least intrusive as possible. Moreover, instrumentations used for monitoring can only be placed at authorized locations such as end points. In designing an efficient monitoring system, a number of concerns should be taken into consideration: (1) it should produce a minimal monitoring traffic, (2) identification of the presence of faults and localizing the faults should be swift in order to minimize the damage, (3) it should learn network conditions irrespective of the instrument locations, and (4) it should be scalable in terms of the monitoring traffic and the instrumentation cost as the networks grow in size. As pointed out in [7], network faults or anomalies in modern day reliable networks are rare occurrences over time and space. Thus, anomalies have a sparse presence in network data. Pioneering work in [8] demonstrated the usage of Network Tomography [9] for identification and localization of these sparse network faults via end to end measurements. Work presented in [10] extends the network tomography to mesh networks which by then had been limited to tree structures. Compressive sensing (CS) [11] (reviewed later on) provided a means to evaluate under-determined linear systems for sparse solutions. Authors of [12, 1] proposed to resolve for network faults using compressive sensing. A key limitation in networks is that measurements can only be made over network paths, unlike applications such as image processing where an arbitrary combination of samples can be made. This leads to a poorer performance in terms of sampling for compressive sensing in networks compared to other compressive sensing applications. More specifically, unlike other CS applications, the common place logarithmic scaling of measurements was not achieved in literature so far in the networking domain.

We in [13, 14] proposed adaptive schemes for network fault localization using compressive sensing, which achieve the logarithmic scaling of measurements. This work presents efficient and fast algorithms to monitor networks for faults and adaptively localize them with a minimal number of additional measurements. The proposed schemes use the TCP/IP Loose Source Routing and Route Recording (LSRR) option to be able reach interested network locations from the instrumented locations, as well to record the path the measurements took. This strategy overcomes the access restrictions on co-operating network segments. The proposed schemes are applicable for network faults in a variety of QoSs (Quality of Services) which are additive over network paths. Some examples are link delays, packet losses, and log of packet loss rates. The schemes are implemented in two-stages. The first stage monitors the network with a minimal number of measurements marginally sufficient to cover the network. The objective of this phase is to detect when the network contains faults. Separability conditions laid down in [8] are used to determine the existence of a fault. Once the existence of a fault is detected, the second phase launches an adaptive algorithm to iteratively learn the locations of the faults. Each iteration uses CS to resolve the network. By evaluating the solution provided by CS, the algorithms determine whether the faults are successfully localized, and if not, the next feedback measurement to make. These algorithms are tested on realistic network topologies with realistic network data models, for accuracy, performance and scalability.

Though the main focus of this thesis is on networking domain, the techniques developed are valid for a range of applications that use time-series data. The primary approach in detecting unusual behaviors is to seek for deviations in data from known acceptable behaviors. Methods such as data clustering [15], and PCA scores [16] seek for differences in prominent behaviors of data. Most of the methods detect when a behavior of a data sample is distinctly different from the reference data. While that is an important contribution, detection of subtle difference of behavior is a difficult and also an important issue. Being able to identify a subtle but unusual behavior helps detecting issue before they cause a significant damage. We propose an optimization based algorithm for detecting sparse pattern. To test it, we use real PCB test data. The proposed method detects behavior changes in data, even when the deviations are within a narrow margin.

A key factor in building an efficient data recovery scheme is to exploit the energy packing of data in different domains. Network faults are sparse in time and space domains. Phenomena such as chemical plumes which resembles natural images are dense in time and space, but have a sparse representation in domains such as Fourier and Wavelets. Compressive sensing [17–20, 1] provides an efficient means to solve sparse linear systems. We review the details of CS principals in Section 3.3. In essence when the domain on which the signal is sparse is known, CS recovers the signal with a minimal number of measurements of the order $\mathcal{O}(s\log(n/s))$ where n is the length of the signal and s is the sparsity of the signal in the sparse domain. To achieve best performance of CS, sampling measure is required to have an orthogonalizational property [21] – which states that sampling matrix is approximately orthogonal. We extend this work to understand the recoverability of CS with any sampling measure as practical sampling schemes may not be endowed with the orthogonalizational property. Then we apply CS to achieve phenomena awareness of sensor networks with a minimal measuring overhead. Sensor networks achieve global knowledge of the sensor field via phenomena awareness which leads to intelligent data routing.

Common place image compression such Joint Photographic Experts Group (JPEG) [22] relies heavily on sparse representations and has achieve significant image compression performance. These concepts can be exploited to compress data on sensor networks and minimize the communication cost. Wireless sensor networks are expected to operate with a minimal energy usage. Communicating sensed data is their costliest operation. Thus, reducing the communication cost has been of significant interest. Compressing the data to be communicated will reduce the cost of communication. We propose a distributed approach which enables data compression over the communication structure of the sensor network, to reduce the amount of data to be transmitted. These developments are applied to track a chemical plume at a minimal cost with a high accuracy [23].

The best energy compaction scheme known to date is Karhunen-Loève transform or Principal Component Analysis (PCA). PCA implies that if the most prominent principal components can be estimated, the signal can be recovered with a high accuracy. Based on this principle, authors of [24] propose to use nuclear norm minimization to reconstruct a matrix from a subset of entries. The strategy is to fill in the missing entries such that the rank of the completed matrix is minimal. However, minimizing the nuclear norm does not guarantee a smooth reconstruction, which affect data matrices of applications such as natural images. Thus, we employ nuclear norm minimization in conjunction with compressive sensing based image reconstruction to recover images with a high accuracy from a minimal number of samples.

As a part of this research, several software tools are developed and made available to the public. These tools encapsulate collecting, extracting, and modeling network traffic and other data. The network anomaly analysis toolkit is a suite of tools to separate anomalies from network data, characterize and visualize them. The Robust PCA toolkit provides generation and analysis of a variety of data matrices. The network data collection tools can be deployed against any typical network to gather data similar to those used in this work. Publishing these tools enables future enhancements to the concepts proposed herein and grants the researchers a head start.

Rest of the thesis is arranged as follows. Chapter 2 presents the precise problem statement addressed in this work. Network monitoring and fault localization are addressed in Chapter 3. Data recovery techniques are researched in Chapter 4 and feature extraction is researched in Chapter 5. Then we proceed to network behavior modeling. Chapter 6 discusses modeling anomalies and Chapter 7 discusses baseline modeling. Conclusions are presented in Chapter 8. Finally, Appendix A reviews the algorithms used in this work, Appendix B describes the software developed under this work, and Appendix C lists key sources codes.

CHAPTER 2

Problem Statement

Network performance is a key factor of the performance of an information system. A healthy network delivers a vast quantity of information faster, i.e., a healthy network has a high throughput and a low delay. Whereas an unhealthy network has a poor information delivery capacity, which in turn affects the performance of the entire computer system. Therefore maintaining network performance targets is a major responsibility of network administrators. These targets are referred to as Service Level Agreements (SLAs) [25].

As with any real-world system, computer networks are vulnerable to component failures. Even with redundancy and replication, time lag bringing up fail over resources may affect performance. Thus, sufficient provisioning and constant monitoring of the network is a major part of the administrators responsibilities. Unusual user behaviors are another aspect that has to be taken into consideration. Events such as flash crowds [26], may exceed the capacities of even the best provisioned computer systems. Network administrators are expected to detect, throttle and maintain operation under such overwhelming but legitimate traffic. Resources of a computer system are accessible via its network. Therefore the resources have to be protected against malicious connections arriving via the network that may harm the computer system. Due to criticality of such protection measures, fields such as cyber-security are gaining momentum nowadays. Also, the cost of deploying an efficient computer network cannot be discounted. To this end, efficient and low-cost solutions for various computer system related problems are of great interest. Further, an Understanding of the distribution of loading and utilization of elements in a computer system allows designing networks with appropriate provisioning.

In order to support provisioning, maintaining health, and assuring security of computer networks, information about the network has to be constantly retrieved and analyzed. However, care should be taken to make the process of information extraction and evaluation not a burden to the network, especially when the network is overloaded or affected in some other way. Otherwise the solution will contribute to and worsen the problem. Therefore efficient data reconstruction techniques via parsimonious sampling are of much interest. Such techniques lead to drawing a complete picture of a given scenario from a minimal set of information. Once the information is extracted, they need to be analyzed rapidly. Thus, fast fault localization algorithms are another branch of vital solutions needed. If certain actions should be taken to regain the health of the network, those actions have to be triggered. Further, the characterization of interesting features such as the baseline and anomalous behaviors, leads to optimum network design. The presented work attempts to encapsulate the above discussed breadth of problems.

Challenges that inspired this work is discussed in Section 2.1. The overall research goals are reviewed in Section 2.2. In Section 2.3 the specific research objectives are discussed. Finally, in Section 2.4 the solution approach employed under this work is presented.

2.1. Challenges and Motivation

The problem domain concerned in this work is centered around monitoring networks, diagnosing issues and modeling behaviors. As with any other problem domain, a range of issues arises as we seek practical solutions. Computer networks contains an abundance of data. However, the efficient retrieval and use of which is yet to pick up speed. The work presented aims at efficient use of computer network data to solve prevalent problems in the domain. This work address challenges in scalability, accessibility, speed, and insight in computer network traffic.

As networks grow in size, challenges of a few different natures begin to arise. One aspect is the escalation of instrumentation to cover the increased network elements. Thus, a large cost has to be incurred to monitor the network. Another aspect is the escalation in complexity. Elements of a network are connected and thus, are correlated in many ways. As the number of network elements increase, these correlations become even more complex. The increased complexity poses a challenge to fault localization algorithms. Furthermore, data needed to process for monitoring, fault localization, and network characterization also increases. Even more, these data are needed to be delivered over the network itself to the processing locations. Despite the important role played by network performance data, retrieving them without burdening the network is challenging. Thus, developing methods to efficiently retrieve interesting and important information about the network aids a wide variety of networking and other applications.

Larger networks such as the Internet span beyond the scope of a single authority. Access granted for co-operating entities by each other are mostly restricted. For example an entity operating at network end points may not have complete access to the core of the network to place monitoring devices. However, end to end performance is affected by issues in the core. Thus, schemes have to be devised to operate at end to end points, but have capabilities to diagnose and localize issues even in the core without complete access. Such problems are not restricted to a single core. Large networks such as the Internet have multitudes of sub-network units, such as Autonomous Systems (AS), which may be owned by different groups but operate in parallel. In such scenarios access restrictions would become a more complicated problem. So schemes that require minimal access privileges and that maximally exploit the available information are in great demand.

In spite of the size of the network, administrators require to know the existence and locations of issues in the network soonest possible. This is a major challenge as complexity of any algorithm would only grow with network size. Thus, fast algorithms that have low complexities are in demand. Complexity of the algorithm is not the only factor that governs the speed. Algorithms such as those are adaptive, require feedback, in these cases, from the network. Thus, efficient feedback mechanisms contribute to the speed of the algorithm. A key improvement here would be to identify the optimum feedbacks that deliver the most information leading to a rapid convergence of the algorithm.

Gaining insight into the network would tremendously improve network management, provisioning and also designing future networks. However the breadth of mathematics developed to describe key behaviors of networks is limited. Thus, new approaches have to be identified to efficiently and comprehensively capture network behaviors. To this end, network behavior modeling is of much interest. However, identifying parameters to model and the approaches for modeling is a major challenge. Features such as network traffic anomalies do not follow commonly known and used distributions. Thus, new approaches have to be investigated and devised to effectively capture behaviors of such properties.

Network administrators maintained networks facing the above challenges over the last few decades. However as the network grow in scale, the current methods used are increasingly becoming insufficient. Finding solutions to these challenges would greatly enhance network performance and will ease the duties of administrators leading to more performant and reliable networks.

2.2. Research Goals

The ultimate goal of this work is to develop network monitoring infrastructures that are highly scalable to a point they are almost invariant to network size, highly efficient that they optimally collect and use network information, and completely model all aspect of networks. This thesis innovates on the prevalent technologies and presents new concepts leading to the above goal. For this, we take a data-centric approach.

The context of this thesis centered around data mostly specific but not limited to computer networks. The goal is to seek optimum usage of data to address problems identified in previous sections. For that, efficient means of data collection, processing, and interpretation are needed. Impact on the network made by collecting data should almost be negligible. This becomes critical when the network is under threat. Data gathering should not add up weight to an already overloaded system. Once the data is collected they should be analyzed in a timely manner. Fast algorithms and efficient feedback systems are critical for this. Data analysis should be able to construct the larger picture of the network with a minimal amount of data and also reveal interesting features of the behavior. If network behaviors can be characterized once the features are extracted that would help provisioning and designing future networks.

Thus, the goal of this thesis is to investigate and develop efficient gathering, processing, feature extraction and characterization of network data, with the possibility to extend beyond the scope of networks. For this, we seek to borrow concepts from signal processing, machine learning, optimization, and TCP/IP. We also seek to establish the validity of the finding by testing on realistic scenarios whenever possible. Furthermore, we pay special attention to performance and accuracy of the developed schemes, as well as the practicality of deployment.

2.3. Research Objectives

The research objectives of this work are organized around the following four aspects.

(1) Retrieving network data

(2) Data recovery

(3) Extracting features

(4) Characterizing behaviors

Under retrieving network data aspect, we investigate approaches to retrieve important information from networks efficiently. A primary concerns in collecting data is to gather and deliver data in a timely manner and also not to burden the network. Thus, schemes relying on naturally available or easily measurable properties are of interest. This work leads to efficient network monitoring. Retrieving location information of faults is also viewed as a part of this. We also investigate ways to exploit the sparse nature of network faults. Another aspect to study is the accessibility to the network. Retrieving internal network information from end points in particular is a key interest.

Optimal data retrieval schemes would only require a minimal amount of data to build a complete picture of the network behavior. Thus, we investigate data recovery schemes that operate on sparse and parsimonious samples. This involves looking into domains where data has a compact representation and investigating sampling strategies to enable successful data recovery. Practicality is another key factor to be taken into consideration. The data recovery methods developed should be able to perform against real world data. Furthermore, theoretical guidelines should be established to predict the recoverability and select sampling specifications. We also extend the scope of investigations beyond network data into general data occurring in other applications.

Once data is collected, features have to extracted and analyzed. We seek to investigate different domains for representing data and different data decomposition schemes to extract features. We extend our investigations into algorithm implementations and evaluations. We pay a special attention to practical recoverability and thus, attempt to establish empirical recovery regions for the feature extraction algorithms. We focus on the feature extraction methods that may be tied to interesting physical features of data. Network data contains a wide range of features whose interplay reveals interesting behaviors. Moreover, we investigate methods that can extract features obscure to a human operator.

Another objective of this thesis is to characterize network behaviors. Such characterizations have applications in efficient provisioning and designing. We seek to investigate extracting and characterizing interesting behaviors such as baselines and anomalies. These behaviors are to be modeled so that their properties are well captured and also enable regeneration. The ability to regenerate behaviors is imperative for studies. Due to sporadic nature of most network data, they are not amenable to classical modeling methodologies. Therefore, new approaches for effectively modeling network behaviors should be investigated. The models developed should be able to concisely represent behaviors enabling compact storage and communication of interesting network behaviors.

2.4. Solution Approach

In this section we establish the solution approaches to address the research objectives and goals identified so far. Our approaches are data centric and we pay a special attention to practical implementation of the solutions. We also try to leverage from the existing technologies and build upon them. Where no existing methods answer to our requirements, we develop new solutions. In all cases we test on real world or realistic data and evaluate performance of the solutions.

Recently, a wide range of algorithms were introduced that take advantage of data sparsity. We study and borrow concepts from these developments. Further, as discussed in [8], faults in the core can be felt by end to end paths going thru the faulty point(s). Therefore we build on Network Tomography algorithms to localize faults using end to end measurements. This allows data extraction through access restricted networks. As network sizes grow, solutions that resolve the entire network would be costly and inefficient. Thus, we research on adaptive algorithms that use feedbacks to localize faults. The idea behind here is to limit probing to the area of interest and narrow down the area of interest iteratively - thereby efficiently localizing faults and limiting the measurement load.

Compressive sensing, matrix completion, and wavelet analysis are some of the concepts we use for data reconstruction. When a signal has a sparse representation on some known domain, compressive sensing allows reconstruction of the entire signal from a very small set of samples [19, 27]. The "signal" in our case is the fault state of all the network elements. As network faults are uncommon, the signal would be sparse in time and spatial domains. In the cases of measurements such as those from chemical plumes, the signals are sparse in domain such as discrete cosine and wavelets. We intend to identify the sparsity domains for each application and use the appropriate transform to recover signals. As most data matrices can be closely approximated by a few principal components, nuclear norm based matrix completion [24] allows estimating matrices from a limited number of samples. Wavelet compression is widely used for compressing natural images. By applying wavelet compression distributively over a sensor field can drastically reduce the amount of information that needs to be transmitted over the network. This is highly beneficial especially for applications such as resource limited wireless sensor networks.

We investigate extracting features from data using techniques as feature filters, Fourier methods and Robust PCA. Network data carry a number of different fields, making them amenable to convenient feature extraction via different filters. While each feature individually is interesting, their inter-connection and correlation are also quite interesting as well. Further, we view these features as distributed in space and time. Thus, we study spatiotemporal correlation of features. Fourier methods reveal interesting features of signal that are repetitive. Many network measurements such as, CPU usage, memory usage, and network usage show diurnal and weekly patterns. Therefore Fourier methods are can decompose and extract interesting features in those data. Karhunen-Loève transform or Principal Component Analysis (PCA) is the best known energy compacting scheme. Therefore PCA will produce the most prominent features of a dataset. However as pointed out in [28] contamination can significantly spoil the identification of principal features. As a remedy, we investigate the usage of Robust PCA [5, 6]. We establish its recoverability and applicability to problem of our interest.

Most of the prevalent models do not capture enough information to successfully reproduce traffic with similar nature. This issue is especially noticeable with traffic anomalies. Thus, we propose a new approach for modeling network data. First we decompose data into physically meaningful components, namely, baseline and anomalies. Then we model each component separately with appropriate techniques that is best suited to describe each feature. Baselines are prominent and consistent across the data. Thus, we employ methodologies such as PCA and Fourier analysis to describe baseline behaviors. As anomalies are sporadic, they are not so amenable to commonly used to random processes. Thus, we introduce a new modeling scheme for anomalies where important properties of anomalies are modeled separately with common random processes. This leads to a more accurate and a concise description of anomaly behaviors. We further investigate the ease of communication and real time application of the extracted behaviors.

CHAPTER 3

Network Monitoring and Fault Localization

3.1. Introduction

This chapter focuses on monitoring computer networks for faults, and upon detection of faults to localize them. We mainly present two algorithms, one an enhancement of the other, for this purpose. We also present the evolution of the algorithms that lead to these two. We begin by reviewing the problem of network monitoring and fault localization and discuss the state of art. We also review theoretical background leading up the proposed methodologies. The presented algorithms have two phases. The initial phase monitors the network for faults with a minimal cost. Then upon detection of presence of faults, they initiate the second phase to localize the faults. It should be pointed out that we do not assume access to individual network elements, preventing us from directly verifying a detection. This would be the case in a realistic network where the access to the entire network may not be available to a single party. Thus, we device indirect methods to verify a localization and if a localization is not accurate to generate further measurements to improve the localization. The first complete network monitoring and fault localization algorithm is presented in Section 3.2. Then a more improved, noise resilient version is presented in Section 3.3. In Section 3.4, we additionally present another algorithm to select effective monitoring paths and the evolution of the fault localization algorithms. The presented algorithms are tested on realistic topologies and with a few different data models which includes realistic loss and delay models.

3.2. An Adaptive Compressive Sensing Scheme for Network Tomography Based Fault Localization

A scalable network fault localization scheme based on compressive sensing is proposed. Aimed at large networks, the proposed scheme monitors a network with a few paths covering the network, and upon detection of anomalies in one or more paths, adaptively carries out additional end-to-end measurements to localize the faulty links. Each adaptive measurement covers a set of links identified based on the previous resolution. The scheme is highly scalable as the total number of measurements required grows logarithmically with the number of links in the network - a level of scalability not practically achieved for network data inference with compressive sensing so far. The scheme is tested on realistic Internet topologies with Gilbert-Elliott loss model calibrated with measurements made on Planet-Lab infrastructure. Results indicate that the converged solution of the proposed scheme achieves over 99% detection rates and less than 1% false positive rates. The proposed scalable scheme is accurate in terms of detection, cost effective in terms of implementation, and casts a minimal monitoring traffic load.

3.2.1. Introduction.

A number of considerations challenge monitoring and localizing faults in large networks. As large networks are formed by subnetworks, which often are managed as separate entities, complete access of the network to a single party may not exist. Also, the sheer size of networks or even a subnets demands a significant amount of instrumentation. Thus, schemes such as "Network Tomography" [9] have gained a significant interest recently. Network tomography techniques probe the network from endpoints and infer internal performance and QoS characteristics. Schemes of interest to large networks have to scale well with the network size, with respect to the instrumentation cost, the number of measurements or measurement time, and the monitoring traffic load. This dissertation presents an efficient and a scalable scheme that detects and localizes faulty network links.

Many existing network tomography methods, e.g., [8], impose a tree-like probing scheme where test packets are injected from a root node and observed at multiple leaf nodes. Based on the anomaly characteristics established in [8], authors of [10] propose an efficient scheme to monitor mesh networks and localize anomalies using end-to-end measurements. A framework of three algorithms for monitoring and localizing anomalies in [29] seeks breaches of network performance guarantees and localizes faults based on the anomalous path measurements. Using second order statistics to characterize and localize lossy links is addressed in [30]. Based on this work more recent developments such as Netscope[31] and LIABLI[32] have emerged. Many loss tomography methods also rely on tree probing structures [33–35]. Network delay tomography over tree structures is addressed in [36]. Usage of passive probing techniques as well as active techniques for loss tomography is addressed in [37]. A key issue in network tomography involves setting or finding the routes for the measurements. Possible path selection strategies include OSPF paths, and use of pre-configured MPLS [38]. However, when the exact path cannot be found, routing matrices can be inferred using non-negative matrix factorization [39, 40].

The concept of “Compressive Sensing” [20, 19, 27] sheds new light to the network tomography domain. Compressive Sensing (CS) provides a mathematical foundation to retrieve a sparse solution to an under-determined linear system, from a logarithmic fraction of realizations. Considering that in most production networks today, anomalies occurring at a given time are on a small set of links, the idea of sparse solutions is highly attractive. However, satisfying CS requirements in a network setup is quite challenging as highlighted in [1, 41]. In spite of such theoretical challenges, [42] demonstrates the use of CS for network tomography. However, CS based methods tested on networks have achieved only a limited amount of savings (in the order of 50%) [1]. As pointed out in [43], resource restrictions in the measurement systems cause CS based recoveries to fail. Using an insufficient number of measurements and the measurements not satisfying recovery conditions are some examples that cause CS to fail in the network monitoring domain. An adaptive scheme is proposed in [43] as a remedy for scenarios that lack resources for CS. Practical adaptive CS algorithms are currently used in areas such as radar and image processing. However they are not readily applicable in the networking domain.

The contribution of this work is an adaptive compressive sensing scheme for network tomography based fault localization requiring far fewer tomographic measurements than state-of-the-art [1]. The number of measurements scales logarithmically with the number of links, therefore it requires much less instrumentation, especially in large networks. Furthermore, the proposed scheme casts a significantly less monitoring traffic load on to the network.

3.2.2. Background. Efficiency of the proposed scheme is due to the use of compressive sensing (CS). If a signal is sparse, i.e., it contains only a few non-zero elements in a known domain, CS can recover the signal with far fewer samples of the signal than the number of elements. In fact, compressive sensing literature [19, 18] states that the number of samples required for successful reconstruction of the signal is a logarithmic fraction of the signal length. If the signal has n elements, k of which are non-zero, the signal can be reconstructed with m samples where $m = \mathcal{O}\left(k \log\left(n/k\right)\right)$. Internet traffic anomalies typically affect only a small fraction of network elements [7]. Thus, we seek to exploit this similarity between network anomalies and sparse signals to efficiently monitor for and localize network faults.

For a network with n links, of which k are anomalous, a successful implementation of CS is expected to localize the faulty links with $m = \mathcal{O}\left(k \log\left(n/k\right)\right)$ samples.

Signal recovery via CS can be formulated as follows. Consider a linear system consisting of a matrix $A \in \mathbb{R}^{m\times n}$, a vector $x \in \mathbb{R}^{n\times 1}$, and a vector $p \in \mathbb{R}^{m\times 1}$:

$$Ax = p \tag{3.1}$$

Here A is referred to as the measurement matrix, while x is the unknown but sparse signal and p contains the compressive measurements. In a network tomography setup, A indicates the routes for each of the tomographic measurements, x represents the unknown QoS values of each link, and p contains the cumulative QoS values over each measurement path. The case of interest to us is when $m \ll n$, i.e., when the system given by (3.1) is highly under-determined. CS literature shows that when matrix A satisfies certain conditions such as Restricted Isometry Property [19, 27], the solution to sparse x is unique, and that it can be found by solving for minimum $\|x\|_0$ (L_0 norm) solution, i.e., the solution with the minimum number of non-zero elements. CS literature recommends using L_1 norm minimization since L_0 minimization is intractable. The mathematically tractable L_1 norm minimization achieves the L_0 norm minimum solution with a very high probability for sparse signals x, when A is well conditioned.

Random matrices have been shown to be good candidates for A [21]. But realizing random measurements matrices is difficult in networks. The construction of binary matrices that are good measurement matrices is addressed in [44]. Exploiting the fact that routing matrices are binary, [42] connects compressive sensing to network tomography and demonstrates that the recovery conditions in [44] can be met in network setups when the networks have only one faulty link. However, results presented in [1, 41] do not hold with the logarithmic factor

$m = \mathcal{O}(k \log(n/k))$ for general CS recovery in network setups. Further, the recovery bounds of existing CS methods rely on the knowledge of the signal, such as sparsity. However, such knowledge is not available in reality. A practical recovery algorithm has to depend only on the information provided by the measurements themselves.

CS recovery is vulnerable to many factors including noise, poor measurement matrix A, dense x, etc. As discussed in [43] a possible remedy for such scenarios is an adaptive approach. An adaptive scheme takes into account the problem at hand and even partial knowledge obtained in the process to drive the solution, rather than seeking a solution to a general class of problems. When a network contains a fault, the requirement is only to localize that particular fault. However, adaptive signal recovery via compressive sensing on a network setup has not been addressed in literature as of now. The practical scheme proposed in this dissertation adaptively applies compressive sensing to localize faulty links and only uses knowledge provided by the measurements themselves to determine the convergence.

3.2.3. Network Monitoring and Fault Localization.

We refer the linear system (3.1) as the "measurement set." The measurement value of path i is the i^{th} element of the vector p. The i^{th} row of A indicates the number of times path i goes through each link. If path i goes through link j once, a_{ij} - the element on row i column j of A - is set to one, if path goes through the link twice, a_{ij} is set to two, and so on. If path i does not go through link j , a_{ij} is zero. This representation can be used for any additive network QoS parameter, such as link delays [36, 45], log of packet transmission rates [30–32], and packet losses [33]. The proposed scheme is of two phases. The first phase monitors the network for presence of a fault. Upon detection of a fault, the second phase for localizing the faulty network elements is initiated.

3.2.3.1. *Monitoring Phase.* The network is monitored with a few tomographic monitoring path measurements. Monitoring measurements can be implemented with random walks [1] or more strategically as discussed in Section 3.4.1. These measurements cover all the network links and form the initial set of measurements. Coverage of all the links is required to guarantee the detection and subsequent localization of a fault on any of the links. Though the monitoring measurements may not be sufficient to localize a fault, they are indicative of when an anomaly is present. During the monitoring phase, the path measurement vector p is inspected for significant deviations. If a path measurement does not exceed a certain threshold, none of the links on the path are in error. A significant deviation in p indicates that the network contains one or more faults that affect the end-to-end network performance. In such cases the adaptive fault localization phase discussed below is initiated.

3.2.3.2. *Adaptive Fault Localization Phase.* The monitoring path measurements are merely indicative of a presence of an anomaly and in general insufficient to localize faults. The scheme discussed here adaptively carries out further path measurements to localize the faults. It follows the algorithm in Fig. 3.1 whose steps can be summarized as follows:

(1) Reduce measurement set to $A_{as}y = p_a$
(2) Solve system of reduced measurements
(3) Check for convergence, and if converged exit
(4) Find link set f for additional adaptive measurements
(5) Collect additional adaptive measurements
(6) Append measurements to the measurement set and repeat the procedure from step 1

Reducing the measurement set: Let set a indicate the subset of paths that have anomalous readings. Then construct a vector p_a by selecting elements on a from p. In addition, build a sub-matrix A_a by selecting rows of A that correspond to a. Figure 3.2a illustrates this

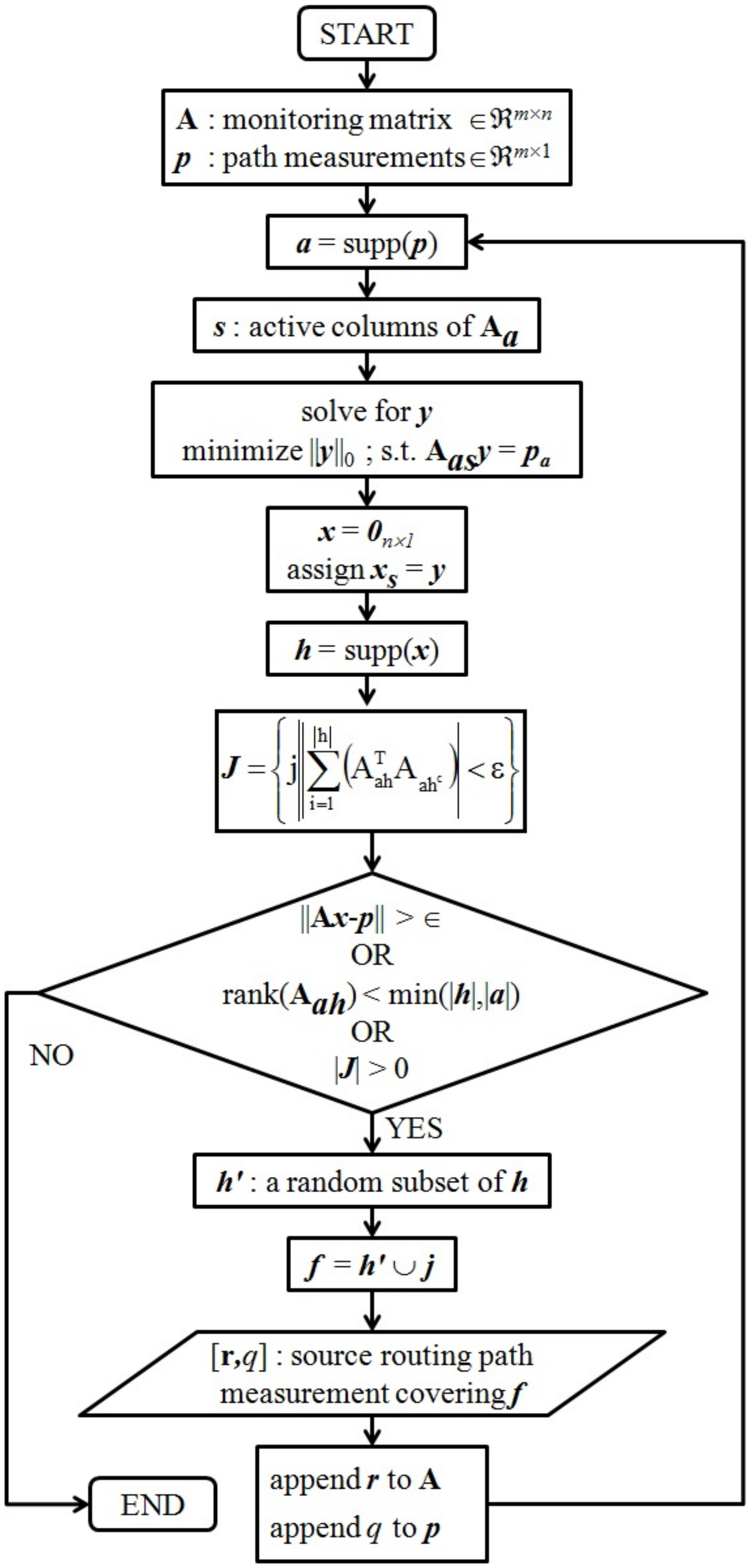
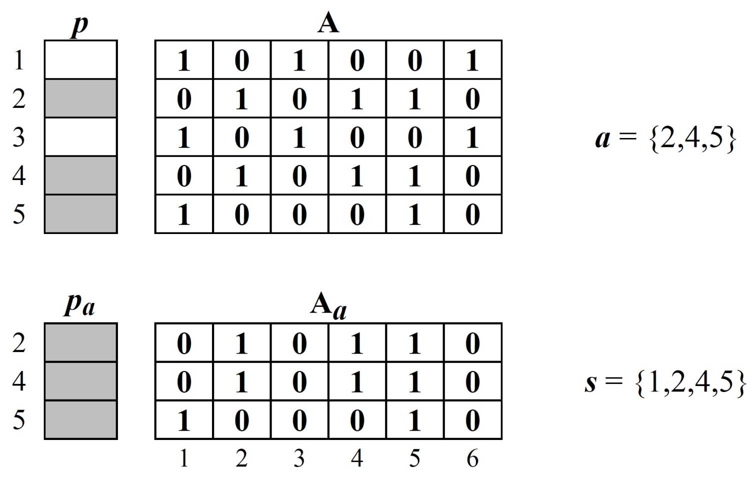
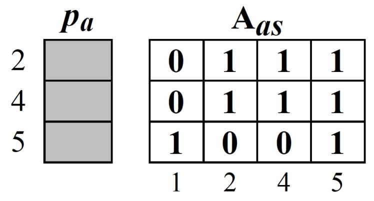

START
$\mathbf{A}$: monitoring matrix $\in \Re^{m\times n}$
$\boldsymbol{p}$: path measurements $\in \Re^{m\times 1}$
$\boldsymbol{a} = \mathrm{supp}(\boldsymbol{p})$
$\boldsymbol{s}$: active columns of $\mathbf{A}_{\boldsymbol{a}}$
solve for $\boldsymbol{y}$
minimize $\|\boldsymbol{y}\|_0$; s.t. $\mathbf{A}_{\boldsymbol{as}}\boldsymbol{y} = \boldsymbol{p}_a$
$\boldsymbol{x} = \boldsymbol{0}_{n\times 1}$
assign $\boldsymbol{x}_s = \boldsymbol{y}$
$\boldsymbol{h} = \mathrm{supp}(\boldsymbol{x})$
$\boldsymbol{J} = \left\{ j \,\middle|\, \left| \sum_{i=1}^{|h|} \left(A_{ah}^{T} A_{ah^c} \right) \right| < \varepsilon \right\}$
$\|\mathbf{A}\boldsymbol{x}\text{-}\boldsymbol{p}\| > \in$
OR
$\mathrm{rank}(\mathbf{A}_{\boldsymbol{ah}}) < \min(|\boldsymbol{h}|,|\boldsymbol{a}|)$
OR
$|\boldsymbol{J}| > 0$
NO
YES
$\boldsymbol{h'}$: a random subset of $\boldsymbol{h}$
$\boldsymbol{f} = \boldsymbol{h'} \cup \boldsymbol{j}$
$[\mathbf{r}, q]$: source routing path measurement covering $\boldsymbol{f}$
append $\boldsymbol{r}$ to $\mathbf{A}$
append q to $\boldsymbol{p}$
END

FIGURE 3.1. Adaptive fault localization algorithm.



(A) Reducing rows



(B) Reducing columns

FIGURE 3.2. Reducing the measurement set. Anomalous path measurements are indicated by a shaded cell.

step. As shown, entries of p corresponding to paths 2, 4 and 5 are anomalous. Thus, set $a = \{2, 4, 5\}$. Then A_a is built with rows 2, 4 and 5 of A. Once the set of rows is reduced, a new set s is built with the columns of A_a that have at least one non-zero element. Then by selecting the columns on s from A_a, the reduced matrix A_{as} is built. This is illustrated in Fig. 3.2b. As shown, columns 3 and 6 of A_a are all zero, in Fig. 3.2b. Thus, $s = \{1, 2, 4, 5\}$. Then the reduced A_{as} is built with columns 1,2,4, and 5 of A_a.

Solving the reduced measurement set: The reduced measurement set contains the anomalous path measurements and the links those paths go through. The goal in solving this linear system is to recognize the exact links that caused the path measurements to be anomalous. The reduced measurement set can be expressed as a linear system:

$$A_{as}x_s = p_a \tag{3.2}$$

where x_s is the vector of unknown link QoS values of links in s. The reduced measurement set (3.2) is extremely likely to be an under-determined set of equations, especially for large networks monitored with a few tomographic measurements. We seek a solution that explains the path anomalies with a minimal number of faulty links. Thus, we solve the following CS problem as discussed in Section 3.2.2.

$$\textbf{minimize}\|x_s\|_0 \text{ such that } A_{as}x_s = p_a \tag{3.3}$$

where, $\|\cdot\|_0$ is the L_0 norm. In a traditional CS implementation, the process exits at this point. The faulty links are indicated by the non-zero elements in x_s. As measurement matrices for networks are unlikely to satisfy properties required by traditional CS, leading to solutions that does not indicate faulty links, we employ an adaptive approach.

Unlike the existing adaptive sensing methods where the entire measurement matrix is rebuilt [43] or where the measurement matrices are impossible to be realized on networks, the proposed adaptive approach appends additional measurements to the measurement set and iteratively solves until convergence. In each iteration, (3.3) is solved as a $\|\cdot\|_1$ minimization. Though publicly available solvers such as L1 magic [46] can be used, we developed a solver based on Doughlas-Rachford iterations [47] for stability. Should one seek to solve $\|\cdot\|_0$ minimization without the $\|\cdot\|_1$ relaxation, a number of algorithms such as the class of matching pursuit algorithms [48] can be used. It should be noted that the solution delivered from L_1 minimization may not necessarily be the sparsest solution we desire. Further, the solutions may require post-processing. For instance, if link delays were used, they are non-negative. On the other hand, if log of packet loss rates were used, they are non-positive. Thus, any out of range entries in the solution including other invalid entries such as NaNs

(Not a Number) have to be removed via post-processing. The solution is then analyzed for convergence.

Convergence: Since individual link measurements are not available, a convergence criteria to terminate the iterations using only the available path measurements need to be found. Here we present three conditions which guarantee an accurate, minimal, and a unique fault localization using only the path measurements. Violation of any of the three conditions implies an un-converged solution, and therefore continuation with additional measurements.

The first condition is the "accuracy." Since the solution obtained for (3.3) is subjected to post-processing, it may no longer satisfy (3.2) closely. If the processed solution deviates from the path measurements by some preset threshold ϵ, the accuracy condition is considered violated.

$$\|Ax - p\| > \epsilon \tag{3.4}$$

The second condition is the minimality. It implies that the solution cannot be further reduced to fewer links, i.e., every link marked as faulty is needed to describe the anomalies observed at path level. For this, a set h is formed with the indices of the non-zero entries of the solution. Then a sub-matrix A_{ah} is formed by selecting columns corresponding to h from A_{ah}. If the solution can be further reduced, then a linear dependence between the links in h and the corresponding columns A_{ah} should exist. Therefore, if A_{ah} is rank deficient, the minimality condition is considered to be violated.

$$rank\left(A_{ah}\right) < \min\left(|h|, |a|\right) \tag{3.5}$$

The third condition is uniqueness. This condition prevents alternative solutions. A set h^C which is the complement of set h is formed first. By selecting columns of A_{as} on h^C a sub-matrix A_{ah^C} is then formed. Let set J indicate the zero entries in the inner product $\langle A_{ah}, A_{ah^C} \rangle$. Elements in J correspond to possible alternative links that may have caused the same path anomalies. Thus, a non-empty J indicates a violation of the uniqueness condition.

$$|J| > 0 \tag{3.6}$$

If none of the above violations (3.4), (3.5), and (3.6), occur, then set h contains the minimal set of faulty links and the fault localization scheme terminates. Otherwise, the process continues through the next steps.

Selecting additional adaptive measurements (AAMs): If convergence was not achieved, further path measurements are made. Termed as additional adaptive measurements (AAMs), these measurements are designed to remove any ambiguities and converge to the actual solution. Two requirements are sought in the links selected for adaptive measurements: (1) selected links break linear dependences between faulty links indicated in the solution, and (2) selected links include possible alternatives links which are not identified as faulty in the solution. These requirements are achieved by selecting a random subset of links f from h and the links that correspond to zeros in the inner product $\langle A_{ah}, A_{ah^C} \rangle$. Then a path measurement is made to cover links in f.

Carry out AAMs: The goal here is to make AAMs that will route measurement packets through the links in f. It is to be noted that an AAM is not required to cover only or all the links in f. A path measurement may contain other links and even may skip some of the selected links due to routing limitations. If the path measurement did not achieve the anticipated coverage, and as a result convergence was not reached, the next iteration will account for the deficiencies.

Appending the measurement set: Each additional adaptive measurement is appended to the measurement set $Ax = p$. Each new AAM adds a new row to A corresponding to the path and the measurement value adds a new entry to p. If the new AAM value is q, then the current path values vector p is updated as $p = \left[p^T q\right]^T$. Similarly, the new route indicated by row r is appended to measurement matrix A as $A = \left[A^T r^T\right]$. Then adaptive fault localization scheme is repeated on the updated measurement set.

3.2.4. Experiments.

3.2.4.1. *Experimental Setup.* The proposed adaptive compressive sensing based fault localization algorithm is tested via the simulation setup described here and details can also be found in Section 3.4.1. Realistic network topologies are generated using the IGen topology generator [49] simulating backbone networks on one continent, connected using Delaunay triangulation. The link faults are simulated with Gilbert-Elliott model [50] which uses a two state Markov chain. This model emulates bursty packet losses prevalent in the Internet, using a faulty state where packets are continually dropped and a no-faults state where no packets are dropped. To obtain realistic parameters, we made measurements on the Planet-Lab infrastructure [51] for link losses and tuned the fault model. The Planet-Lab measurements suggested a probability of $1/\left(1.5 \times 10^6\right)$ to transition to faulty state and a probability of 0.05 to transition to no-faults state. Each measurement consisted of 1000 packets transmitted over a path. Measurement packets are generated at a network probing device attached to a network node, and the measurement path terminates again at a probing device which retrieves test packet information. All the simulated networks are assumed to be monitored by 10% of nodes with attached probes scattered across the network. However, it was noticed that the number of probes used has little impact on performance, long as more than 10 probes are used. A random set of paths initiating and terminating at probes that cover the network are selected for network monitoring. For AAMs, two probes closest to the set of links f are selected and source routing measurement are made between the two probes over path that includes the links on f. Path measurement vector p contains the log of path pass rates. Pass rate is (1 - loss rate). Pass rate of a path j denoted by d_j is the product of pass rate of all links on the path. This formulation is similar to the previous work [31, 32]. If the loss rate of a link i which is a member of path j is r_i then

$$d_j = \prod_i (1 - r_i) \tag{3.7}$$

$$p_j = \log d_j = \sum_i \log (1 - r_i) \tag{3.8}$$

The performance is analyzed with two parameters: detection rate (DR) and false positive rate (FPR). Links that have a loss rate over 5% are treated as faulty. Let the set of links that actually are faulty be T and the set of links the scheme identified as faulty be S, then

$$\text{DR} = \frac{|T \cap S|}{|T|} \times 100\% \tag{3.9}$$

$$\text{FPR} = \frac{|T \cap S^C|}{|S|} \times 100\% \tag{3.10}$$

3.2.5. Results. The main goal of the proposed scheme is to monitor and localize faults in large networks with a small number of measurements. Further, the measurements required were expected to scale slowly with the network size. The accuracy of detection is also of significant interest. The results presented below answers these concerns.

Figure 3.3 shows the number of measurements needed for a range of network sizes. As can be noted, the total number of measurements needed is significantly less than the number of links in the network. Notably, only a few additional measurements were required for at all network sizes to successfully localize the faulty links. This demonstrates the credibility of the criteria used to specify additional adaptive measurements, and also the ability to rapidly localize faults.

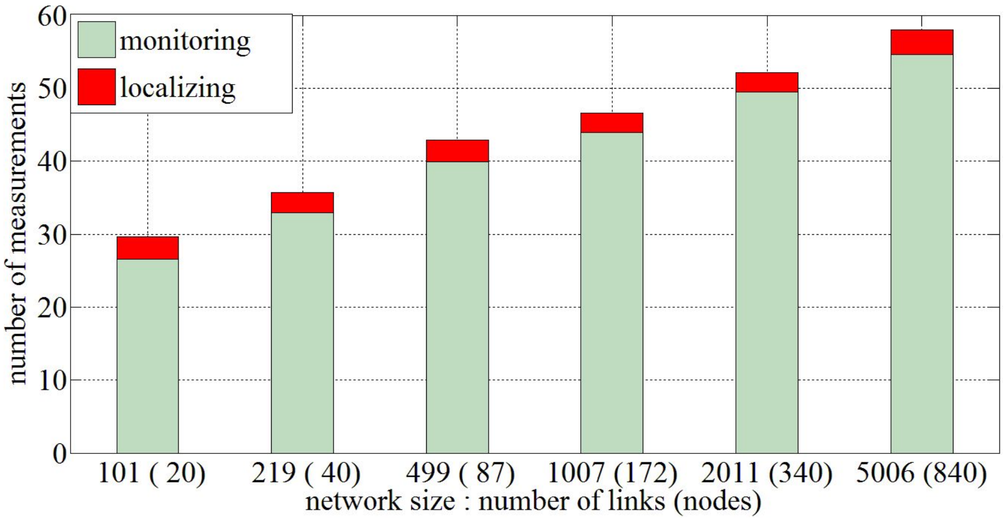


FIGURE 3.3. Measurement cost vs. network size. Each column corresponds to a certain network size. The average number of links and the number of nodes (within paranthesis) for each size is shown.

Next, we demonstrate the scalability of required number of measurements with the network size. As of now, there is no evidence that traditional compressive sensing approach on networks has practically achieved logarithmic scaling of number of network measurements with links. But as the results in Fig. 3.4 show, the required total number of measurements for successful localization of faults increases logarithmically with the number of links in the network for the proposed scheme. This result supports the scalability of the proposed scheme. Since the cost of instrumentation increases with number of measurements, the scheme can be implemented with less instrumentation, the scheme is cost effective.

As illustrated in Fig. 3.5 the proposed scheme achieves a very high detection rate and a very low false positive rate for the range of network sizes using the realistic loss model. The detection rate is consistently over 99.0% and the false positive rate is consistently below

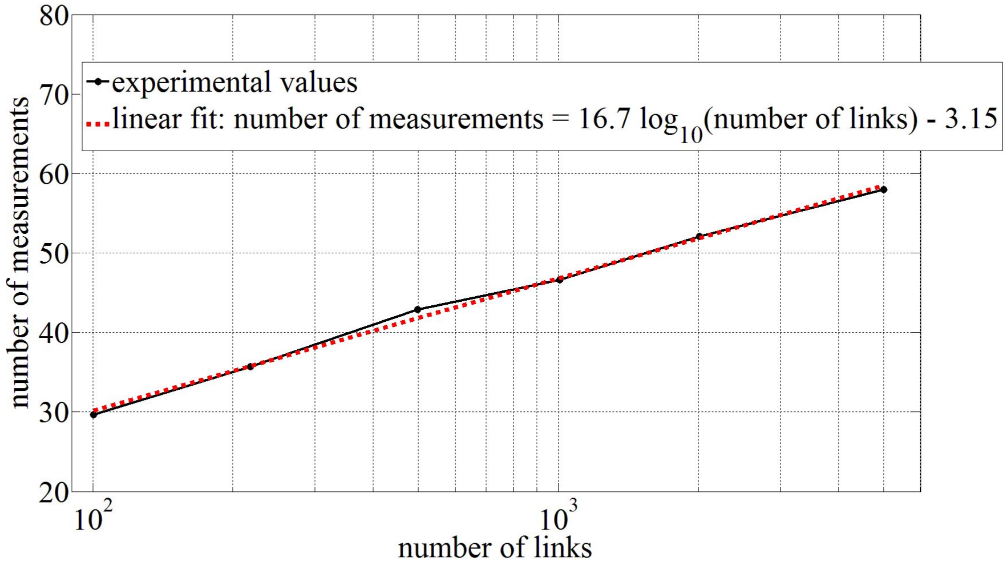


FIGURE 3.4. Logarithmic scaling of number of measurements with number of links.

1.0% for all the network sizes tested. This demonstrates the credibility of the conditions used for convergence test. If the network is instrumented so that link level measurements can be made for verification, detection rate achieves 100% and false positives drop to 0%. Finally

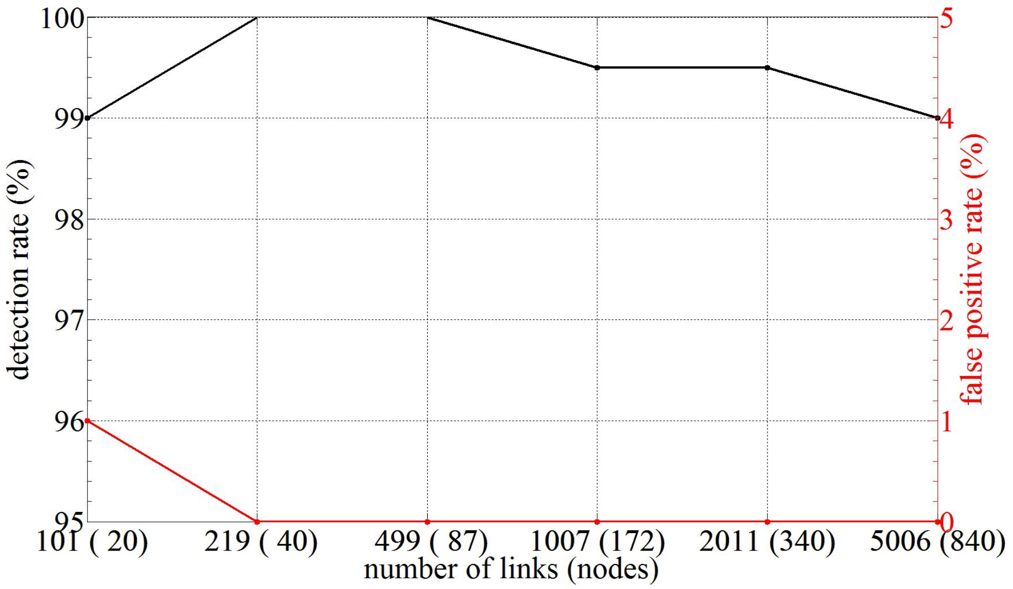


FIGURE 3.5. Fault localization performance.

we provide a comparison against a standard compressive sensing over graphs implementation in Fig. 3.6. Here we apply the proposed scheme to bidirectional complete networks similar to those used in [1] and then compare the number of measurements required for the proposed scheme against the 50% measurement savings claimed [1]. We note that detection and false positive rates of both schemes are same. As can be observed the proposed adaptive compressive sensing scheme for network fault localization provides tremendous measurement savings.

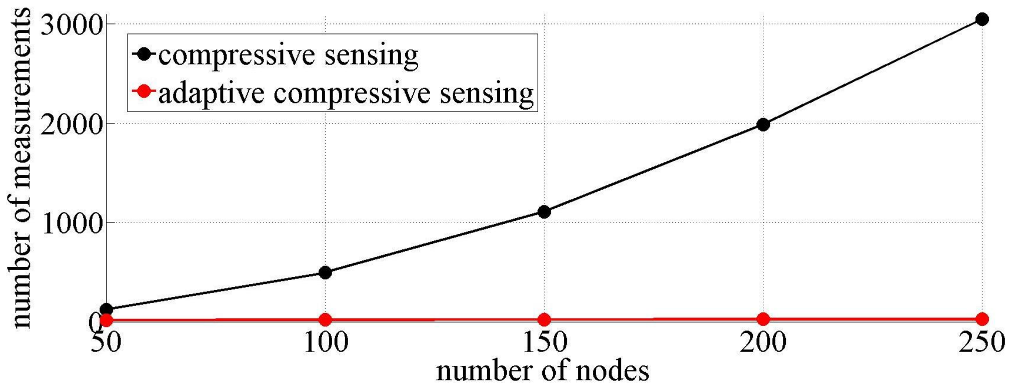


FIGURE 3.6. Comparison of proposed adaptive CS method against standard CS implementation [1].

3.2.6. CONCLUSIONS. An adaptive compressive sensing scheme for network tomography based fault localization was proposed. It achieves a very high detection rate and a very low false positive rate with a number of measurements that scales logarithmically with the number of links in the network. Experiments on realistic internet topologies with 100 links to 5000 links show the total number of measurements required scaled logarithmically - a level of scalability that has not been demonstrated so far for network tomography. Further, the proposed three conditions for convergence and the two criteria for selecting the links for additional adaptive measurements, lead to a fault localization with a minimal number of additional measurements and assures a rapid localization process. Thus, the proposed scheme is efficient for monitoring and localizing faulty links of large networks in terms of accuracy, speed, instrumentation cost, and measurement traffic load.

3.3. Adaptive Compressive Sensing for Network Fault Localization

A scalable technique that localizes network faults efficiently in large networks while imposing a minimal overhead is proposed. The scheme monitors the network of interest with minimal end-to-end monitoring measurements, and upon detection of a fault, initiates an adaptive fault localization process. An adaptive compressive sensing method that employs practically available information to guide localization measurements, is developed for fault localization. The algorithm indicates measurements that lead to an accurate, minimal and a unique solution which rapidly converge on to the faulty links. The scheme requires only a limited number of instrumentations sitting at the edge of the network and does not require access to the internals of the network. The proposed scheme is tested on simulated realistic Internet topologies with a number of different data models including realistic loss and delay models, to evaluate performance and cost. Results show a very high detection rate and a very low false positives rate for all data models tested. Results also indicate that the total number of measurements required grows nearly logarithmically with the network size. This is a scalability that is unachieved so far in literature.

3.3.1. Introduction.

Effective, reliable and economical network monitoring schemes, especially those targeted at large networks, are increasingly in demand. As instrumenting and monitoring each network node and/or link individually is costly and impractical in large networks, efficient approaches such as network tomography [9] are of significant interest. Network tomography estimates internal network parameters with a few end to end measurements. An efficient monitoring and fault localization system should scale well with respect to the links and nodes in the network, in terms of the number of measurement equipment and the amount of test traffic imposed on the network while guaranteeing a high quality of monitoring. As the network size grows, restricting the escalation of equipment and test traffic is crucial to prevent the growth of cost of monitoring. To guarantee quality of monitoring with fewer resources requires efficient and reliable techniques. With products such as JDSU PacketPortalTM [52] and SFProbeTM [53] which allows for filtering of packets, identifying and reporting header associated values such as timestamps, a novel class of monitoring and fault localization techniques for networks has become feasible. In this section, we present a technique for efficiently localize network faults. The presented scalable scheme requires only a very limited number of network nodes to be instrumented with monitoring probes and casts a low monitoring and fault localization traffic on to the network.

A brief review of closely related work is provided next. Many existing work on network tomography proposes a tree-like probing scheme where test packets are injected from a root node and observed at multiple leaf nodes. A few algorithms with plausible performance for localizing faulty links from path measurements for tree networks appear in [8]. Based on the anomaly characteristics established in [8], authors of [10] propose an efficient scheme to monitor mesh networks and localize anomalies using end-to-end path measurements. A framework of three algorithms for monitoring and localizing anomalies is presented in [29]. They seek breaches of network performance guarantees while achieving a good coverage of the network. When such a breach is detected, the anomalous network elements are localized using an elimination algorithm based on the anomalous path measurements. Locating links with excessive losses and delays using network tomography based methods stands out in the literature. Using second order statistics to characterize loss behaviors of links and locating faulty links is addressed in [30]. This work has provided the base for more recent developments such as Netscope [31] and LIABLI [32]. Similar to other network tomography schemes, many loss tomography methods also rely on tree probing structures [33–35]. Network delay tomography over tree structures is addressed in [36]. Usage of passive probing techniques as well as active techniques for loss tomography is addressed in [37]. An active probing scheme called Flexicast is used for delay tomography in [45]. Another common approach for general network tomography is using network coding. Loss tomography via network coding is proposed in [54]. A key issue in network tomography is either setting or learning the routes of the tomographic measurements. As reviewed in [38], among the possible solutions are periodically downloading and calculating OSPF paths, and use of pre-configured MPLS paths. However, when the exact path cannot be learned, routing matrices can be inferred using non-negative matrix factorization [39, 40].

The concept of "Compressive Sensing" [55, 20, 27, 17] facilitates novel techniques for network tomography. Compressive Sensing (CS) provides a mathematical foundation to retrieve a sparse solution to an under-determined linear system. Considering that in most production networks today the anomalies affect only a small fraction of elements at a given time, the idea of sparse solutions seems highly attractive. However, satisfying CS requirements in a network setup is quite challenging as highlighted in [1, 41]. In spite of the theoretical challenges, [42] proposes the use of CS for network tomography. They demonstrate that expander graph based recovery guarantees of CS can be achieved with routing matrices of a network tomography setup. More theoretical bounds are derived in [1] and [41] which are further reinforced by simulated examples. Despite the significant literature on efficient network monitoring techniques, there is a significant gap between the desired performance and the theoretical achievements. Much of the earlier work relies on tree-based probing, which either does not provide sufficient coverage or is costly to provide a complete coverage for mesh topologies. Also, most work is limited to binary faults. Approaches such as in [1] tested on more realistic QoS parameters on mesh networks have achieved only a limited amount of savings (in the order of 50%). As pointed out in [43], resource restrictions in the measurement systems cause CS based recoveries to fail. Adaptive schemes are a remedy proposed for these scenarios. In some of the earlier work, an iterative focusing algorithm that adaptively collects measurements and retains only a subset of the estimates at each iteration is used [43]. Among the more recent developments of adaptive CS algorithms, [56] proposes LASeR (Learning Adaptive Sensing Representation) - an adaptive algorithm to solve convex optimization problems for CS when coefficients exhibit a tree structure in some orthonormal dictionary. The existing adaptive CS algorithms, which have been developed for applications such as radar, image processing, are not readily applicable in the networking domain.

The focus of this section is the development of an adaptive compressive sensing algorithms targeted for network monitoring. A preliminary but a different version of the proposed scheme is presented in the companion paper [13]. The proposed scheme monitors the network with a minimal number of end-to-end measurements for faults and upon detection of a fault, initiates an adaptive fault localization algorithm to localize the faulty links that uses a few additional end-to-end measurements. The total number of measurements required by the scheme grows logarithmically with the number of links in the network. Therefore the scheme is highly scalable and very cost effective for large networks. Furthermore, the results show a high detection rate and a low false positives rate for a number of different data models including realistic loss and delay models, on realistic Internet-like topologies. The remainder of the section is arranged as follows. Section 3.3.2 reviews some preliminaries required for the context of the section. The analytical foundation of the proposed work is laid in Section 3.3.3. The proposed network monitoring and fault localization scheme is presented in Section 3.3.4. Section 3.3.5 discusses the experimental setup and Section 3.3.6 presents the results of the experiments carried out. Finally, Section 3.3.7 makes the concluding remarks.

3.3.2. Preliminaries. We begin by reviewing a few related technical concepts, to ease the conversation forth. First we lay down the taxonomy and then review compressive sensing and fixed sparse signal recovery, as they form the base for the proposed scheme.

3.3.2.1. *Taxonomy.* The following notation is used throughout the section. Scalars are indicated with non-bold italic lower-case characters; for example: x. While vectors and sets are indicated with bold italic lower-case characters as in $\boldsymbol{x}$, matrices are indicated with bold italic upper-case characters as in $\boldsymbol{X}$. When sub-sets or sub-matrices are formed, a subscript representing the index set is attached to these symbols; for example: $\boldsymbol{X}_y$.

If a certain sparse vector $\boldsymbol{x}$ has k non-zero elements whose locations are listed in a set $\boldsymbol{s}$, k is referred to as the sparsity of $\boldsymbol{x}$, and $\boldsymbol{s}$ is referred to as the *support* of $\boldsymbol{x}$. In signal processing, similarity between two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ if often quantified with coherence yielded from their inner product $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \boldsymbol{x}^T \boldsymbol{y}$. Let $\boldsymbol{A}$ be a matrix of n columns, and $\boldsymbol{a}_i$ and $\boldsymbol{a}_j$ denote the i^{th} and j^{th} columns. Then worst case coherence of $\boldsymbol{A} = \min_{i,j} \left\langle \boldsymbol{a}_i^T, \boldsymbol{a}_j \right\rangle$ for $i, j = 1, \ldots, n$ and $i \neq j$. It should be noted that, if the matrix or vectors are strictly binary, *Hamming distance* can be used in place of the inner product. Two norms occur frequently in this section: L_0 norm and L_1 norm. The L_1 norm of a vector or a matrix $\boldsymbol{X}$, denoted by $\|\boldsymbol{X}\|_1$, is the sum of absolute values of the entries of $\boldsymbol{X}$. The L_0 norm of $\boldsymbol{X}$, denoted by $\|\boldsymbol{X}\|_0$, is the number of non-zero elements (also called active elements) in $\boldsymbol{X}$.

The presented work revolves around network faults. Network faults are significant deviations of QoS (Quality of Service) parameters of the network. We use the symbol ξ to denote the strength of a faulty QoS sample, and it is expressed as a factor of the range of the acceptable QoS values. If γ is the maximum acceptable value for a certain QoS parameter, and if a certain faulty sample had a value of ζ, then $\xi = \zeta/\gamma$. When convenient, ξ is expressed in dBs.

Certain QoS parameters are additive over network paths. For example, delay over a network path is the sum of the delays of individual links of the path. If the packet loss rate is denoted by is r, the log of the pass rate, $\log(1-r)$, is also an additive QoS parameter. We refer to such path QoS values as *tomographic end-to-end path values.* A significant deviation of a link QoS value will cause a significant deviation of the tomographic end-to-end path value, which sets the premise of the proposed network monitoring and fault localization scheme.

3.3.2.2. *Compressive Sensing.* Efficiency of the proposed scheme is due to the use of compressive sensing (CS). If a signal is sparse, i.e., it contains only a few non-zero elements in a known domain such as time, space, frequency, wavelet, etc., CS can recover the signal with far fewer samples of the signal than the number of elements. In fact, compressive sensing literature [19, 20, 27, 17, 18] states that the number of samples required for successful reconstruction of the signal is a logarithmic fraction of the signal length. If the signal has n elements, k of which are non-zero, the signal can be reconstructed with m samples where $m = \mathcal{O}(k \log(n/k))$. Internet traffic anomalies typically affect only a small fraction of network elements at a given time [7]. Thus, we seek to exploit this similarity between network anomalies and sparse signals to efficiently monitor for and localize network faults. For a network with n links, of which k are anomalous, a successful implementation of CS is expected to localize the faulty links with $m = \mathcal{O}(k \log(n/k))$ samples. Though this target is not yet practically achieved in literature.

Signal recovery via CS can be formulated as follows. Consider a linear system consisting of a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, a vector $\boldsymbol{x} \in \mathbb{R}^{n \times 1}$, and a vector $\boldsymbol{p} \in \mathbb{R}^{m \times 1}$

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{p} \tag{3.11}$$

Here $\boldsymbol{A}$ is referred to as the measurement matrix, while $\boldsymbol{x}$ is the unknown but sparse signal and $\boldsymbol{p}$ contains the compressive measurements. In a network tomography setup, $\boldsymbol{A}$ indicates the routes for each of the tomographic measurements, $\boldsymbol{x}$ represents the unknown QoS values of each link, and $\boldsymbol{p}$ contains the cumulative QoS values over each measurement path. The case of interest to us is when $m \ll n$, i.e., when a network of many elements is monitored with a few measurements. Notably, the system given by (3.11) is highly under-determined when $m \ll n$, thus, no unique solution is available. CS literature shows that when matrix $\boldsymbol{A}$ satisfies certain conditions such as Restricted Isometry Property (RIP) [19, 27], the solution to sparse $\boldsymbol{x}$ is unique, and that it can be found by solving for minimum $\|\boldsymbol{x}\|_0$ (L_0 norm) solution, i.e., the solution with the minimum number of non-zero elements. CS literature recommends solving (3.11) as a L_1 norm minimization since L_0 minimization is intractable. The mathematically tractable L_1 norm minimization achieves the L_0 norm minimum solution with a very high probability for sparse signals $\boldsymbol{x}$, when $\boldsymbol{A}$ is well conditioned as in RIP

$$(1-\delta)\|\boldsymbol{x}\|_2 \leq \|\boldsymbol{A}\boldsymbol{x}\|_2 \leq (1+\delta)\|\boldsymbol{x}\|_2 \tag{3.12}$$

where the constant δ is called the Restricted Isometry Constant (RIC).

Random matrices have been shown to be good candidates for $\boldsymbol{A}$ [21]. But realizing random measurements matrices as routing matrices in networks is impractical. Work such as in [57] discusses the construction of deterministic matrices which have good recoverability in CS. The construction of binary matrices that are good measurement matrices is addressed in [44]. Exploiting the fact that routing matrices are binary, [42] connects compressive sensing to network tomography and demonstrates that the recovery conditions in [44] can be met in network setups when the networks have only one faulty link. However, results for CS implementations on networks such as those presented in [1] and [41] do not hold with the logarithmic factor $m = \mathcal{O}\left(k \log(n/k)\right)$ for general CS recovery in realistic network setups. Further, the recovery bounds of existing CS methods rely on prior knowledge of the signal, such as sparsity. However, such prior knowledge is not available in reality. Therefore the existing bounds cannot be used to build a practical CS implementation for network monitoring. A practical detection algorithm has to depend only on the information provided by the measurements themselves.

CS recovery is vulnerable to many factors including noise, poor measurement matrix $\boldsymbol{A}$, dense $\boldsymbol{x}$, etc. As discussed in [43] a possible remedy for such scenarios is to use an adaptive approach. Based on the idea of adaptive compressive sensing, a number of extensions are stemmed, such as LASeR (Learning Adaptive Sensing Representation) [56] an adaptive algorithm to solve convex optimization problems for CS when coefficients exhibit a tree structure in some orthonormal dictionary. An adaptive scheme seeks to solve the problem at hand using partial knowledge obtained in the process, rather than solving a general class of problems. When a network contains a fault, the requirement is only to localize that particular fault. However, adaptive signal recovery via compressive sensing on a network setup has not been addressed in literature as of now. The practical scheme proposed in this section adaptively applies compressive sensing to localize faulty links and only uses knowledge provided by the measurements themselves to determine the convergence.

3.3.2.3. *Recovery of Fixed Sparse Signals.* As noted above, if the scheme is adaptive, conditions to recover a general sparse signal are not necessary to meet. As mentioned earlier when a network contains a fault, it is only required to localize that particular fault. Recovery of fixed sparse signals is discussed in [58, 59]. The following theorem is initially proposed in [58] and then is generalized in [59]. A comprehensive review of the theorem can be found in [21]. A signal $\boldsymbol{x}$ with support $\boldsymbol{s}$ can be successfully recovered via Compressive Sensing when the measurement matrix $\boldsymbol{A}$ satisfies

$$\left|\left\langle \boldsymbol{A}_{\boldsymbol{s}}^{+}\boldsymbol{a}_i, \operatorname{sgn}\left(\boldsymbol{x}_{\boldsymbol{s}}\right)\right\rangle\right| \leq 1 \forall i \notin \boldsymbol{s} \tag{3.13}$$

where $\boldsymbol{A}_{\boldsymbol{s}}^{+}$ is the pseudo-inverse of the sub-matrix of columns of $\boldsymbol{A}$ over $\boldsymbol{s}$, $\boldsymbol{a_i}$ is the i^{th} columns of $\boldsymbol{A}$, $\boldsymbol{x_s}$ is the vector of elements of $\boldsymbol{x}$ over $\boldsymbol{s}$, and sgn$(\cdot)$ is the signum function. The proof can be found in [58, 21, 59]. As can be noted, this optimality condition depends on a priori knowledge of the signal's support. If the knowledge of signal sparsity is in fact available, the system equation (3.11) can be constructed to satisfy the above theorem. However, such knowledge is not available in practice.

Using an adaptive scheme to resolve a fixed sparse signal, as we do in this section, has not been discussed yet in literature. Further, as shown above, recovery bounds of existing methods rely on prior knowledge of the signal. The scheme proposed in this section adaptively applies compressive sensing to localize faulty links and use only the knowledge provided by the measurements and the solver to determine the convergence. Thus, it relies only on practically available information, not practically unavailable prior knowledge.

3.3.3. Analytical Foundation.

The proposed scheme localizes faulty links from path measurements. For this we need to develop an understanding of how faults affect network paths. The path measurements are resolved using compressive sensing, but with a novel adaptive approach which uses existing knowledge to converge on the solution. This section discusses the underlying concepts of the proposed scheme.

3.3.3.1. *Sum of Random Variables and Faults.* Network link QoS values can be treated as random variables. We are interested in QoS values which are additive over network paths. Sum of two random variables is obtained by the convolution between the two. By

Central Limit Theorem, it is known that the sum of many (typically greater than 30) random variables approximates to a Gaussian distribution centered around the sum of the means of the individual random variable with a variance which is the sum of the variances of the individual random variables. In other words, with a 68% probability the path measurements would lie within one standard deviation around the sum of the means of link QoS values, and with a 95% probability within two standard deviations and with a 99.7% probability within three standard deviations. More precisely with a probability of $\operatorname{erf}\left(n/\sqrt{2}\right)$, the path measurements would lie within n standard deviations around the sum of the means of link QoS values.

When a path measurement contains a faulty link measurement, the path measurement will significantly deviate from the expected range. However, it should be noted that the variance of longer paths is high and therefore the deviation caused by a link fault may lie within the non-faulty range of path measurements. For a fault on a path of h links to make a deviation beyond n standard deviations, the path length should satisfy

$$h \leq \left(\frac{\zeta - \gamma}{n\sigma_l}\right)^2 \tag{3.14}$$

where ζ is the magnitude of the fault, γ is the maximum non-faulty link QoS value and σ_l is the standard deviation of link level QoS parameter. The derivation is found in Appendix 3.3.8. This result provides a rule for limiting monitoring path length. For the simple random data model discussed later in Section 3.3.5, the limit on the path length is simplified to $12(\xi - 1)^2/n^2$. This quantity is derived in Appendix 3.3.9.

When measurements are beyond a certain threshold they are considered anomalous. Based on the desired detection rate and false positive rate, a threshold can be found more precisely as follows. Let $\beta = \zeta - \mu_l$, where μ_l is the mean of link level QoS parameter, and

α denotes the threshold. Then the distribution of path level QoS values with and without faults can be described as shown in Fig. 3.7. The detection and false positive rates are

$$\text{Detection rate} = \left(1 + \operatorname{erf}\left(\frac{\beta - \alpha}{\sqrt{2}\sigma}\right)\right)/2 \tag{3.15}$$

$$\text{False positives rate} = \left(1 + \operatorname{erf}\left(\frac{\alpha}{\sqrt{2}\sigma}\right)\right)/2 \tag{3.16}$$

where σ is the path level standard deviation. Derivations of these quantities are discussed in Appendix 3.3.10. An ROC (Receiver Operating Characteristic) curve for the above is shown in Fig. 3.8, which enables selecting a suitable threshold to obtain a desired performance in detecting faults of an interested magnitude.

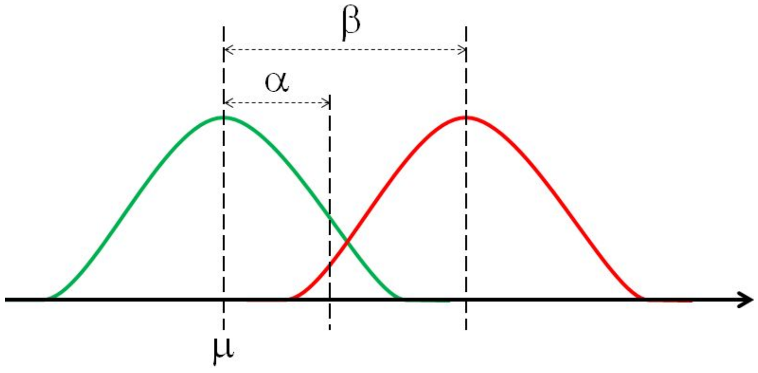


FIGURE 3.7. Distribution of path level QoS values with (red) and without (green) a fault. Mean of the no-fault path measurements is indicated by μ.

3.3.3.2. *Criteria Indicating a Correct Fault Localization.* Compressive Sensing solves the system equation (3.11) as a minimization of L_1 norm. The proposed fault localization scheme iteratively appends adaptive measurements to (3.11) and solves until a correct localization is achieved. Here we develop the criteria that indicate a correct localization. Notably, these criteria depend only on the available information, unlike the prevalent CS methods.

We first study an example where CS L_1 minimization does not yield the sparsest solution we seek. Let us consider the following scenario illustrated in Fig. 3.9. It has multiple possible

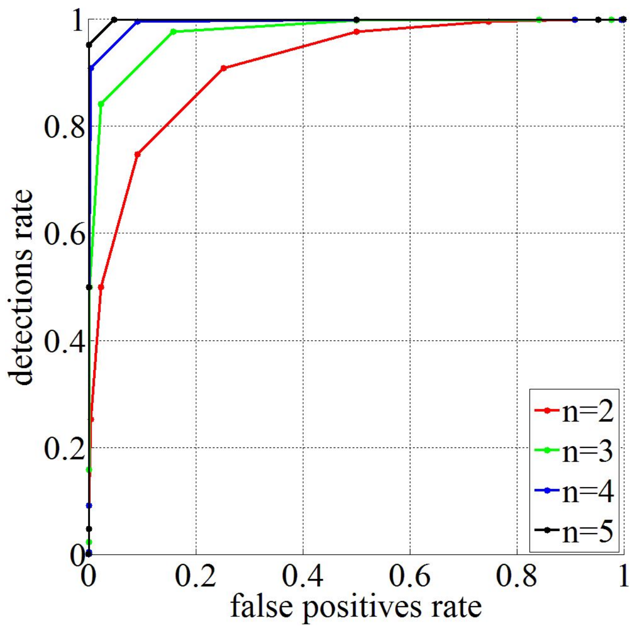


FIGURE 3.8. ROC for different fault magnitudes and thresholds. Deviation caused by faults $\beta = n\sigma$, where σ is the path level standard deviation.

L_1 minimum solutions, to name a few: (1) $x_2 = 1$, $x_3 = x_4 = 0$ (2) $x_2 = 0$, $x_3 = x_4 = 0.5$ and (3) $x_2 = 0.5$, ${}_3 = x_4 = 0.25$. While $\|\boldsymbol{x}\|_1$ of all these three solutions is 1.0, $\|\boldsymbol{x}\|_0$ is 1.0, 2.0 and 3.0 respectively. Though the first solution is the sparsest, an L_1 minimization cannot guarantee to produce the sparsest, as it cannot distinguish the sparsest solution from the other possibilities long as they all have the minimum L_1 norm. Quite possibly, the solver would produce the least squares solution, i.e., $x_2 = x_3 = x_4 = 1/3$ in this case.

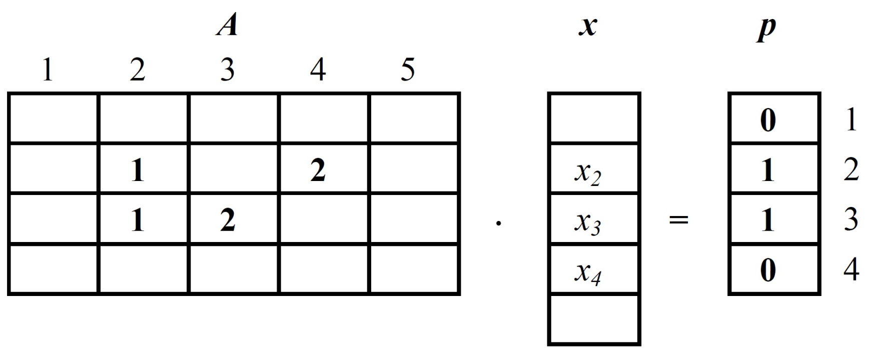


FIGURE 3.9. An example system having multiple L_1 minimum solutions, but has a unique L_0 minimum solution.

Thus, we need one or more criteria that indicate when the solver produces a less than acceptable answer. If the solution is unacceptable, by making additional measurements the ambiguities can be resolved. When the solution satisfies the criteria, no additional measurements will be needed and resolution can terminate. We identified three criteria which indicate when the solver produces an acceptable parsimonious solution. In fact, as discussed in the next subsection, these criteria even guide additional measurements.

When the proposed adaptive scheme achieves a correct localization, the solution had the following properties: (1) it was stable - meaning, the solution does not change with further measurements, (2) it was minimal - meaning, the solution cannot be further reduced, i.e., it had no redundancies, and (3) it was unique - meaning, there are no alternative solutions. Further, these three criteria are testable using available knowledge. Thus, they are very practical to implement.

The first criterion - *stability* - is assessed by keeping track of the support of the solution produced by the solver. When the solution does not show any churn between iterations, the solution is said to be stable. The second criterion is *minimality*. Let the support of an intermediate solution be $\boldsymbol{h}$. Since elements outside $\boldsymbol{h}$ are redundant, the system (3.11) can be expressed as

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{p} = \boldsymbol{A}_{\boldsymbol{h}}\boldsymbol{x}_{\boldsymbol{h}} \tag{3.17}$$

where $\boldsymbol{A}_{\boldsymbol{h}}$ is a submatrix formed with the columns of $\boldsymbol{A}$ on $\boldsymbol{h}$, and $\boldsymbol{x}_{\boldsymbol{h}}$ is vector formed with elements of $\boldsymbol{x}$ on $\boldsymbol{h}$. If $\boldsymbol{h}$ can further be reduced to $\boldsymbol{g}$, then $\boldsymbol{A}_{\boldsymbol{h}}\boldsymbol{x}_{\boldsymbol{h}} = \boldsymbol{A}_{\boldsymbol{g}}\boldsymbol{x}_{\boldsymbol{g}}$ where $|\boldsymbol{h}| > |\boldsymbol{g}|$. That implies $\operatorname{rank}(\boldsymbol{A}_{\boldsymbol{h}}) \leq \operatorname{rank}(\boldsymbol{A}_{\boldsymbol{g}}) < |\boldsymbol{h}|$. But if $\boldsymbol{h}$ is minimal, then $\operatorname{rank}(\boldsymbol{A}_{\boldsymbol{h}}) = |\boldsymbol{h}|$. The final criterion is *uniqueness*. Here we seek for elements in the complementary set of $\boldsymbol{h}$, called $\boldsymbol{h}^c$ that can replace elements in $\boldsymbol{h}$. We take a geometric approach for this. First we build an projector $\mathbb{I} - \boldsymbol{A}_{\boldsymbol{h}}\left(\boldsymbol{A}_{\boldsymbol{h}}^T\boldsymbol{A}_{\boldsymbol{h}}\right)^{-1}\boldsymbol{A}_{\boldsymbol{h}}^T$ and then project columns of $\boldsymbol{A}$ corresponding to $\boldsymbol{h}^c$. This projector projects vectors to the perpendicular space of the solution $\boldsymbol{h}$. Should a vector vanishes completely when projected that vector lies on the subspace of the solution and the link it correspond to can be an alternative to some of the links in the current solution. If none of the columns of $\boldsymbol{A}$ corresponding to $\boldsymbol{h}^c$ vanish when projected, then the solution is unique. We will illustrate implementation of these criteria in Section 3.3.4.

3.3.3.3. *Adaptive Measurements.* For an efficient localization involving a minimal number of additional measurements, the most effective adaptive measurements have to be made. We deduce two rules based on the termination criteria discussed earlier. The first rule is to improve minimality of non-minimal solutions. If two links are visited by the same set of measurements, and if one of them has a fault, it would be difficult to pin point which link actually contained the fault. The L_1 solvers are likely to indicate both links as faulty. Columns of $\boldsymbol{A}$ corresponding to such columns have a high coherence value. An adaptive measurement has to decrease the coherence between the columns of $\boldsymbol{A}$ corresponding to the support of the current solution. That will lead to a minimal solution in future resolutions. The way to decrease coherence between a subset of links is to visit each link with different measurement or visit only a subset of links with a measurement. The second rule is to incorporate possible alternatives. For that, adaptive measurements have to go through the possible alternative links indicated previously. Inclusion of alternative links may lead to a non-minimal solution at first. But eventually minimality will be achieved with the first rule. By then, all the possible alternatives will also be taken into consideration, leading to a minimal unique resolution. Again, the implementations of these are illustrated in Section 3.3.4.

3.3.4. Network Monitoring and Fault Localization.

A “good” monitoring system operates with minimal number of equipment and casts minimal monitoring traffic onto the network, and is efficient. Resource requirements of a good monitoring algorithm will scale well with the network size, thus, is scalable. Accuracy guarantee is a major challenge for algorithms as they operate with far fewer resources compared to the size of the network, thus, a good scheme is reliable. The proposed scheme is endowed with the above characteristics. It consists of two phases:

(1) Tomographic monitoring phase

(2) Adaptive fault localization phase

The first phase monitors the network for a presence of a fault. Upon detection of a fault, the second phase for localizing the faulty network elements is initiated. The procedure is summarized in Fig. 3.10.

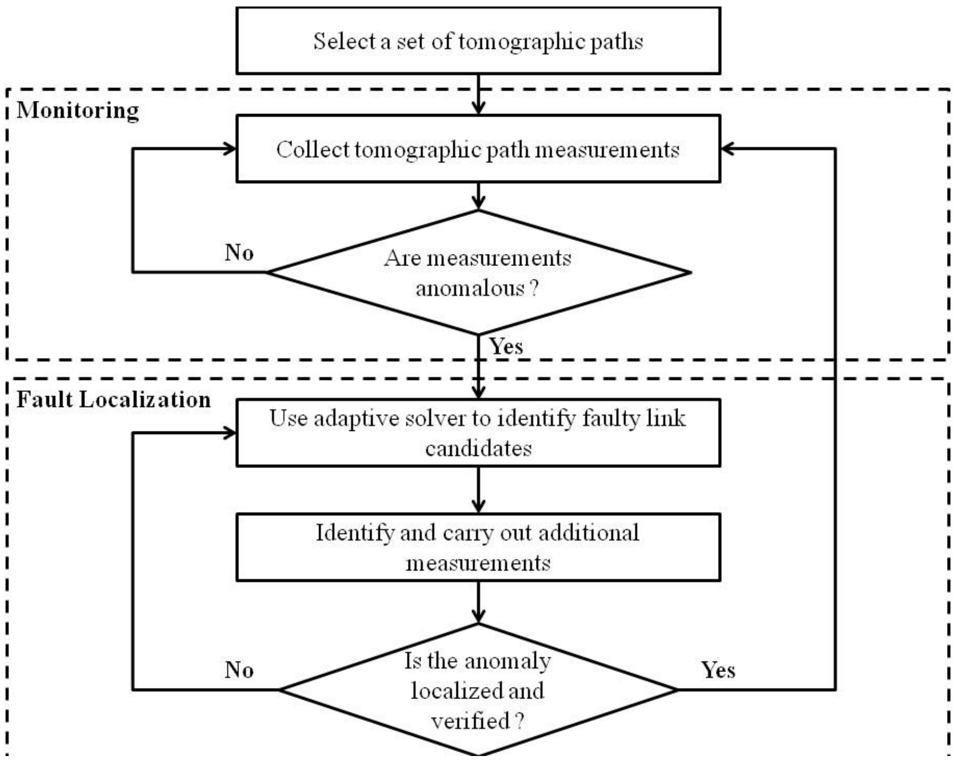

Select a set of tomographic paths
Monitoring
Collect tomographic path measurements
No
Are measurements anomalous ?
Yes
Fault Localization
Use adaptive solver to identify faulty link candidates
Identify and carry out additional measurements
No
Is the anomaly localized and verified ?
Yes

FIGURE 3.10. The monitoring and localization framework.

We refer the linear system (3.11) as the "measurement set." The measurement value of path i is the i^{th} element of the vector $\boldsymbol{p}$. The i^{th} row of $\boldsymbol{A}$ indicates the number of times path i goes through each link. If path i goes through link j once, a_{ij} - the element on row i column j of $\boldsymbol{A}$ - is set to one, if path goes through the link twice, a_{ij} is set to two, and so on. If path i does not go through link j, a_{ij} is zero. It should be noted that though most routing matrices are binary, we do not impose a binary constraint on the measurement matrix. Binary routing matrices are a special case of the general class of measurement matrices considered herein. This representation can be used for any additive network QoS parameter, such as link delays [36, 45], log of packet transmission rates [30–32], and packet losses [33].

3.3.4.1. *Monitoring Phase.* The goal of the proposed approach is to monitor the network with a minimal measurement load on the network, while employing a minimal number of instrumented nodes. The monitoring system keeps the network in check with a few tomographic path measurements. Monitoring measurements can be implemented with random walks [1] or more strategically as discussed in Section 3.4. Subjected to path length restrictions discussed in Section 3.3.3, these measurements cover all the network links and form the initial set of measurements. Coverage of all the links is required to guarantee the detection and subsequent localization of a fault on any of the links. Though the monitoring measurements may not be sufficient to localize a fault, they are indicative of when an anomaly is present. If no path measurement is anomalous, then the network is anomaly-free. But if one or more path measurements are anomalous, then the network contains at least one anomalous link. During the monitoring phase, the path measurement vector $\boldsymbol{p}$ is inspected for significant deviations. If a path measurement does not exceed a certain threshold, none of the links on the path has a fault. A significant deviation in $\boldsymbol{p}$ indicates that the network contains one or more faults that affect the end-to-end network performance. In such cases the adaptive fault localization phase discussed below is initiated.

3.3.4.2. *Adaptive Fault Localization Phase.* The monitoring path measurements are merely indicative of a presence of a fault and quite possibly insufficient to localize the faults. The scheme discussed here adaptively carries out further path measurements to localize the faults. It follows the algorithm in Fig. 3.11 whose steps can be summarized as follows:

(1) Reduce measurement set to $\boldsymbol{A_{as}y} = \boldsymbol{p_a}$

(2) Solve system of reduced measurements

(3) Test termination criteria. If they are satisfied, exit.

(4) Find link set $\boldsymbol{f}$ for additional adaptive measurements

(5) Collect additional adaptive measurements

(6) Append measurements to the measurement set and repeat the procedure from step 1

Reducing the measurement set: Let set $\boldsymbol{a}$ indicate the subset of paths that have anomalous readings. Then construct a vector $\boldsymbol{p_a}$ by selecting elements on $\boldsymbol{a}$ from $\boldsymbol{p}$. In addition, build a sub-matrix $\boldsymbol{A_a}$ by selecting rows of $\boldsymbol{A}$ that correspond to $\boldsymbol{a}$. Fig. 3.12(a) illustrates this step. As shown, entries of $\boldsymbol{p}$ corresponding to paths 2, 4 and 5 are anomalous. Thus, set $\boldsymbol{a} = \{2, 4, 5\}$. Then $\boldsymbol{A_a}$ is built with rows 2, 4 and 5 of $\boldsymbol{A}$. Once the set of rows is reduced, a new set $\boldsymbol{s}$ is built with the columns of $\boldsymbol{A_a}$ that have at least one non-zero element. Then by selecting the columns on $\boldsymbol{s}$ from $\boldsymbol{A_a}$, the reduced matrix $\boldsymbol{A_{as}}$ is built. This is illustrated in Fig. 3.12(b). As shown, columns 3 and 6 of $\boldsymbol{A_a}$ in Fig. 3.12(b) are all zero. Thus, $\boldsymbol{s} = \{1, 2, 4, 5\}$. Then the reduced $\boldsymbol{A_{as}}$ is built with columns 1,2,4, and 5 of $\boldsymbol{A_a}$.

Solving the reduced measurement set: The reduced measurement set contains the faulty path measurements and the links those paths go through. The goal in solving this linear system is to recognize the exact links with the faults. The reduced measurement set can be expressed as a linear system:

$$\boldsymbol{A_{as}x_s} = \boldsymbol{p_a} \tag{3.18}$$

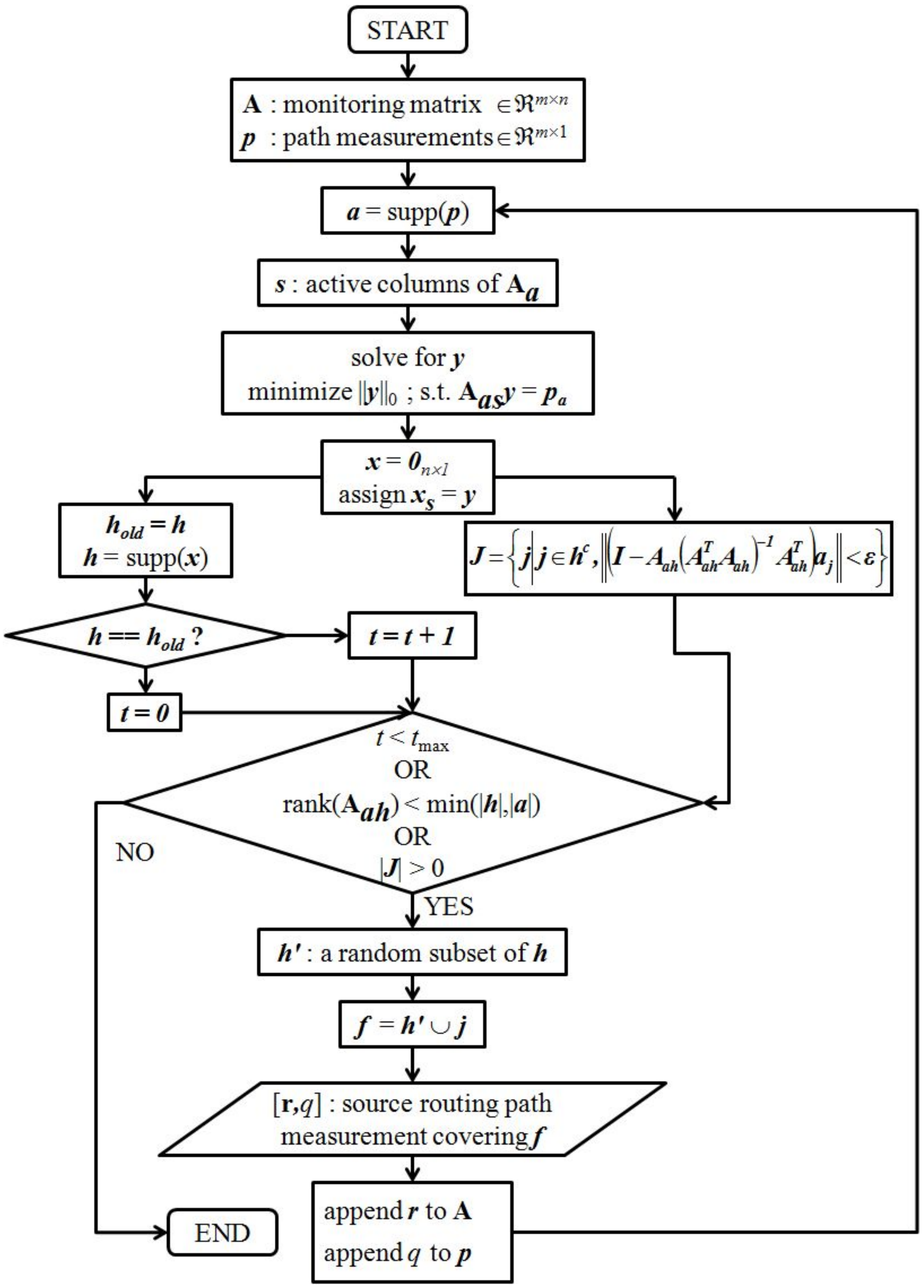

START
$\mathbf{A}$: monitoring matrix $\in \Re^{m\times n}$
$\boldsymbol{p}$: path measurements $\in \Re^{m\times 1}$
$\boldsymbol{a} = \mathrm{supp}(\boldsymbol{p})$
$\boldsymbol{s}$: active columns of $\mathbf{A}_{\boldsymbol{a}}$
solve for $\boldsymbol{y}$
minimize $\|\boldsymbol{y}\|_0$; s.t. $\mathbf{A}_{\boldsymbol{as}}\boldsymbol{y} = \boldsymbol{p}_{\boldsymbol{a}}$
$\boldsymbol{x} = \boldsymbol{0}_{n\times 1}$
assign $\boldsymbol{x}_{\boldsymbol{s}} = \boldsymbol{y}$
$\boldsymbol{h}_{old} = \boldsymbol{h}$
$\boldsymbol{h} = \mathrm{supp}(\boldsymbol{x})$
$J = \left\{ j \middle| j \in h^c, \left\| \left(I - A_{ah}\left(A_{ah}^T A_{ah}\right)^{-1} A_{ah}^T\right) a_j \right\| < \varepsilon \right\}$
$\boldsymbol{h} == \boldsymbol{h}_{old}$?
$t = t + 1$
$t = 0$
$t < t_{\max}$
OR
$\mathrm{rank}(\mathbf{A}_{\boldsymbol{ah}}) < \min(|\boldsymbol{h}|, |\boldsymbol{a}|)$
OR
$|\boldsymbol{J}| > 0$
NO
YES
$\boldsymbol{h}'$: a random subset of $\boldsymbol{h}$
$\boldsymbol{f} = \boldsymbol{h}' \cup \boldsymbol{j}$
$[\mathbf{r}, q]$: source routing path measurement covering $\boldsymbol{f}$
END
append $\boldsymbol{r}$ to $\mathbf{A}$
append q to $\boldsymbol{p}$

FIGURE 3.11. Adaptive fault localization algorithm.

where $\boldsymbol{x_s}$ is the vector of unknown link QoS values of links in $|bms$. The reduced measurement set (3.18) is extremely likely to be an under-determined set of equations, especially for large networks monitored with a few tomographic measurements. We seek a solution that explains

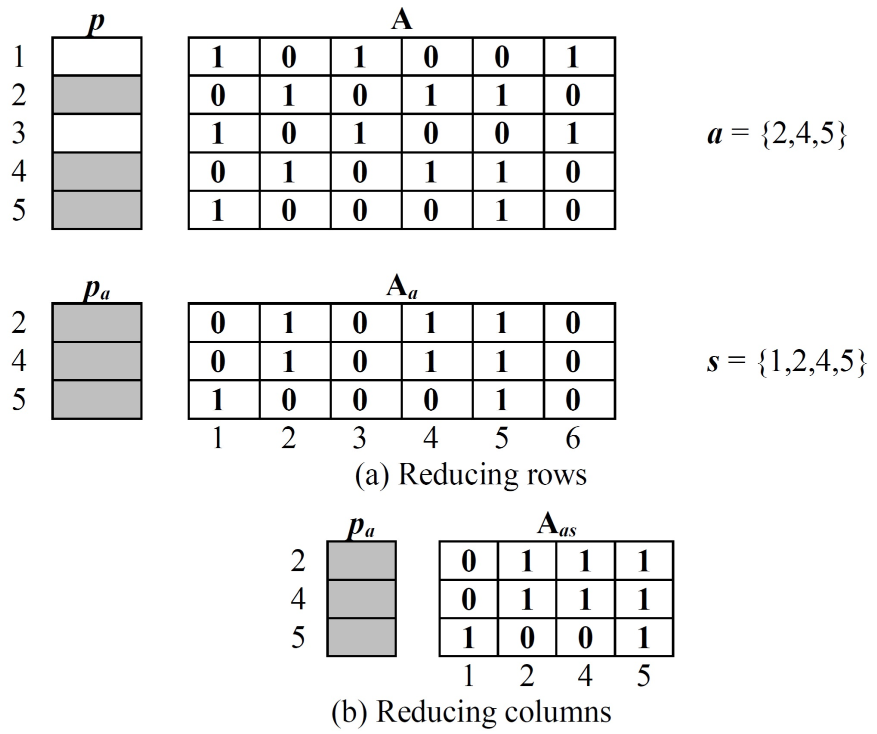


FIGURE 3.12. Reducing the measurement set. Anomalous path measurements are indicated by a shaded cell.

the path anomalies with a minimal number of faulty links. Thus, we solve the following CS problem as discussed in Section 3.3.2.

$$\text{minimize } \|\boldsymbol{x}\|_0 \quad \text{such that} \quad \boldsymbol{A_{as}x_s} = \boldsymbol{p_a} \tag{3.19}$$

where, $\|\cdot\|_0$ is the L_0 norm. In a traditional CS implementation, the process exits at this point. The faulty links are indicated by the non-zero elements in $\boldsymbol{x_s}$. Measurement matrices for networks are unlikely to satisfy properties required by traditional CS and solutions obtained will fail to indicate the faulty links. Thus, we develop an adaptive approach.

Unlike the existing adaptive sensing methods where the entire measurement matrix is rebuilt [43] or where the measurement matrices are impossible to be realized on networks, the proposed adaptive approach appends additional path measurements to the measurement set and iteratively solves until convergence. In each iteration, (3.19) is solved as a $\|\cdot\|_1$

minimization. Though publicly available solvers such as L1 magic [46] can be used, we developed a solver based on Doughlas-Rachford iterations [47] for stability. Should one seek to solve $\| \cdot \|_0$ minimization without the $\| \cdot \|_1$ relaxation, a number of algorithms such as the class of matching pursuit algorithms [48] can be used. It should be noted that the solution delivered from L_1 minimization may not necessarily be the sparsest solution we desire. Further, the solutions may require post-processing. For instance, if link delays were used, they are non-negative. On the other hand, if log of packet loss rates were used, they are non-positive. Thus, any out of range entries in the solution including other invalid entries such as NaNs (not a number) have to be removed via post-processing. The solution is then analyzed for termination criteria.

Termination criteria: Since individual link measurements are not available, a convergence criteria to terminate the iterations using only the available path measurements need to be found. Here we implement the three criteria discussed in Section 3.3.3. Violation of any of the three conditions implies an un-converged solution, and therefore the scheme continues with additional measurements.

The first criterion - stability - is implemented by keeping track of the support of the solution. At each iteration the support of the previous solution $\boldsymbol{h}_{\text{old}}$ is compared against the support of the current solution $\boldsymbol{h}$. If they are similar a counter t is incremented. If $\boldsymbol{h}$ is different, the counter is reset to zero. When the counter increase past a predetermined threshold $t_{\max}$, the solution is said to be stable. The solution is unstable when

$$t < t_{\max}. \tag{3.20}$$

The second criterion is the minimality. It implies that the solution cannot be further reduced to fewer links, i.e., every link marked as faulty is needed to describe the anomalies observed at path measurements. For this, a sub-matrix $\boldsymbol{A_{ah}}$ is formed by selecting columns corresponding to $\boldsymbol{h}$ from $\boldsymbol{A_{as}}$. If the solution can be further reduced, then a linear dependence between the links in $\boldsymbol{h}$ and the corresponding columns $\boldsymbol{A_{ah}}$ should exist. Therefore, if $\boldsymbol{A_{ah}}$ is rank deficient, the minimality condition is considered to be violated.

$$\operatorname{rank}(\boldsymbol{A_{ah}}) < \min(|\boldsymbol{h}|, |\boldsymbol{a}|) \tag{3.21}$$

The third criterion is uniqueness. This condition prevents alternative solutions. A projection operator $\mathbb{I} - \boldsymbol{A_{ah}}\left(\boldsymbol{A_{ah}^{T}}\boldsymbol{A_{ah}}\right)^{-1}\boldsymbol{A_{ah}^{T}}$ is built, first. Then a set $\boldsymbol{h}^c$ which is the complement of set $\boldsymbol{h}$ is formed. Columns of $\boldsymbol{A_{as}}$ corresponding to $\boldsymbol{h}^c$ are projected onto the above projector. The columns that vanishes when projected are listed in a set $\boldsymbol{j}$. Elements in $\boldsymbol{j}$ correspond to possible alternative links that may have caused the same faulty path measurements. Thus, a non-empty $\boldsymbol{j}$ indicates a violation of the uniqueness condition.

$$|\boldsymbol{j}| > 0 \tag{3.22}$$

If none of the above violations (3.20), (3.20), or (3.22) occur, then set $\boldsymbol{h}$ contains the minimal set of faulty links. The fault localization scheme terminates at this point. Otherwise, the process continues through the next steps.

Selecting additional adaptive measurements (AAMs): If the scheme did not terminate in the previous step, further path measurements are made. Termed as additional adaptive measurements (AAMs), these measurements are designed to remove any ambiguities and converge to the actual solution. As discussed in Section 3.3.3, two requirements are sought in the links selected for adaptive measurements: (1) selected links that break linear dependences between faulty links indicated in the solution, and (2) selected links to include possible alternatives links which are not identified as faulty in the solution. The first rule is implemented by selecting a subset of links in $\boldsymbol{h}$. The second rule is implemented by including the set $\boldsymbol{j}$. Thus, a set $\boldsymbol{f}$ is formed by taking the union of a subset of $\boldsymbol{h}$ and the entire set $\boldsymbol{j}$. Then a path measurement is made to cover links in $\boldsymbol{f}$.

Carry out AAMs: The goal here is to make AAMs that will route measurement packets through the links in f. It is to be noted that an AAM is not required to cover only or all the links in f. A path measurement may contain other links and even may skip some of the selected links due to routing limitations. If the path measurement did not achieve the anticipated coverage, and as a result convergence was not reached, the next iteration will account for the deficiencies.

Appending the measurement set: Each additional adaptive measurement is appended to the measurement set $\boldsymbol{Ax} = \boldsymbol{p}$. Each new AAM adds a new row to $\boldsymbol{A}$ corresponding to the path and the measurement value adds a new entry to $\boldsymbol{p}$. If the new AAM value is q, then the current path values vector $\boldsymbol{p}$ is updated as $\boldsymbol{p} = \left[\boldsymbol{p}^T q\right]^T$. Similarly, the new route indicated by row $\boldsymbol{r}$ is appended to measurement matrix $\boldsymbol{A}$ as $\boldsymbol{A} = \left[\boldsymbol{A}^T \boldsymbol{r}^T\right]^T$. Then adaptive fault localization scheme is repeated on the updated measurement set.

3.3.5. Experimental Setup. Here we review the experimental setup used to test the proposed scheme. Our goal is to test the proposed scheme on realistic large networks under real and interesting operating conditions. The details can also be found in .

3.3.5.1. *Realistic Topologies.* Realistic network topologies are generated using the *IGen* topology generator [49] simulating backbone networks on one continent, connected using *Delaunay triangulation.* Work in [60] shows that though the Internet grows in size, its features are relatively stable. Therefore we make a reasonable assumption that IGen provides a faithful representation of realistic networks of the scales we sought to test.

3.3.5.2. *Path Measurements.* We assume the network supports the IPv4 option *Loose Source Routing and Route Recording* (LSRR) [61] for test traffic. This option provides two functions. The first, it permits routing test packets along a path that includes the links we desire. The second, it records the path the test packets took, enabling a convenient construction of the measurement matrix. Measurement packets are generated at a network probing device attached to a network node, and the measurement path terminates again at a probing device which retrieves test packet information. All the simulated networks are assumed to be monitored by 10% of nodes with attached probes scattered across the network. However, it was noticed that the number of probes used has little impact on performance, long as more than 10 probes are used. This is because source routing enables routing to and from desired links, from and to the available probes. Two types of measurements are made in the proposed scheme: monitoring measurements and AAMs.

The goal of the monitoring measurements is to monitor the network with a minimum number of measurements. They have to adhere to two constraints: (1) monitoring measurement should cover the entire network, and (2) each monitoring path has to adhere to the path length limit. This is achieved by building a few longest possible paths that cover the network. Each path is forced to go thru as many unvisited links as possible. The algorithm used is summarized in Fig. 3.13.

For AAMs, two probes closest to the set of links $\boldsymbol{f}$ are selected and source routing measurement are made between the two probes over path that includes the links on $\boldsymbol{f}$. However, there is a possibility that path length may exceed the limit if the entire set $\boldsymbol{f}$ is included in a measurement. The set $\boldsymbol{f}$ contains links included for two purposes: (1) break linear dependence between the links identified as possibly faulty, and (2) alternative links which possibly be faulty. The set $\boldsymbol{f}$ is shuffled, so that the two types of links appear

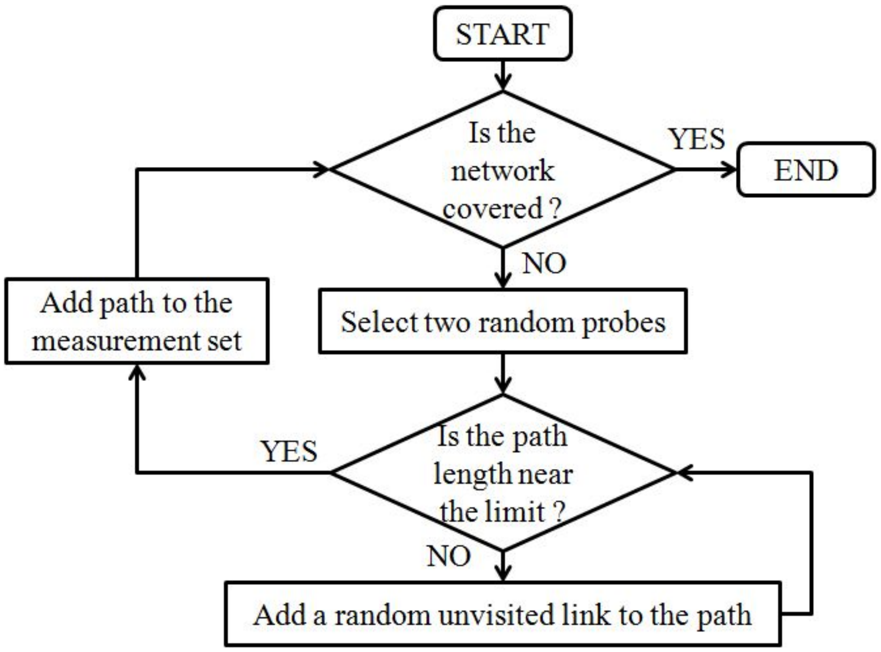

START
Is the network covered ?
YES
END
NO
Select two random probes
Add path to the measurement set
Is the path length near the limit ?
YES
NO
Add a random unvisited link to the path

Figure 3.13. Constructing monitoring path measurements.

in reasonably large subset of $\boldsymbol{f}$. Then a path measurement is developed by including the largest possible subset of $\boldsymbol{f}$ adhering to the path length limit.

3.3.5.3. *Data Models.* Network tomography relies on the fundamental concept of *separability* laid down in [8]. The data model proposed in [8] uses two states of QoS values: *High* and *Low*. The idea of separability and the High-Low data model states that when a link contain a fault, a path going through the link will show a noticeable shift in path QoS values and if a path shows a noticeable shift in QoS values, it must contain at least one faulty link. We test the proposed scheme under three data models: (1) binary fault model, (2) simple random fault model, and (3) realistic loss and delay data model.

The first data model we consider is the binary data model. Mostly used in earlier literature on network tomography such as in [62], binary data model is a highly simplified representation of data and faults, mostly suitable for proof of concept experiments. Under this model, links with no faults are assumed to carry a QoS value of zero and faulty links are assigned with a one. Each path measurement counts the number of one's along the path.

Finally, the fault localization scheme will identify the faulty links which carried a QoS value of one.

The second model we employ is called the simple random fault model. This model is more representative of realistic network data. Under the simple random model, links without a fault are assumed to have a random value in a range $[0, \gamma]$. For simplicity, we assume non-faulty QoS values are uniformly distributed over $[0, \gamma]$. When a link is faulty, its QoS value will escalate to an unbounded random value ζ, which is obviously beyond γ. We express the strength of the fault ξ as a fold of the range of non-faulty values, i.e, ζ/γ.

Finally we simulate realistic network data. We consider two QoS parameters: packet loss rate and link delay. The packet losses are simulated with *Gilbert-Elliott* model [50] which uses a two state Markov chain. This model emulates bursty packet losses prevalent in the Internet, using a faulty state where packets are continually dropped and a no-faults state where no packets are dropped. To obtain realistic parameters, we made measurements on the Planet-Lab infrastructure [51] for link losses and tuned the fault model. The Planet-Lab measurements suggested a probability of $1/\left(1.5 \times 10^6\right)$ to transition to faulty state and a probability of 0.05 to transition to no-faults state. Each measurement consisted of 1000 packets transmitted over a path. For convenience of implementation pass rate which is (1 - loss rate) is used. Pass rate of a path j denoted by d_j is the product of pass rates of all links on the path. This formulation is similar to the previous work [31, 32]. If the loss rate of a link i which is a member of path j is r_i then

$$d_j = \prod_i (1 - r - i) \tag{3.23}$$

$$\boldsymbol{p}_j = \log d_j = \sum_i \log (1 - r_i) \tag{3.24}$$

Network delays are simulated using an alpha-stable heavy-tail distribution [63]. Again we employ measurements made on Planet-Lab infrastructure to calibrate the model, so that the regenerated data is realistic. Based on the measurements we made, we used the following parameter settings for the network delay model. The *characteristic exponent* is set to 1.0. The *skewness* is bounded within [-1,1] and found to follow an exponential distribution with a rate of 0.25 distributed leftwards with an offset of 1.0. The *scale* is found to be distributed exponentially with a rate of 0.02. Finally, the *location* parameter is also found to be distributed exponentially with a rate of 0.2 and an offset of 2.0.

3.3.5.4. *Performance Metrics.* The performance is analyzed with two parameters: detection rate (DR) and false positive rate (FPR). Links that have a loss rate over 5% are treated as faulty. Let the set of links that actually are faulty be $\boldsymbol{t}$ and the set of links the scheme identified as faulty be $\boldsymbol{s}$, then

$$\text{DR} = \frac{|\boldsymbol{t} \cap \boldsymbol{s}|}{\boldsymbol{t}} \times 100\% \tag{3.25}$$

$$\text{FPR} = \frac{|\boldsymbol{t} \cap \boldsymbol{s}^c|}{\boldsymbol{s}} \times 100\% \tag{3.26}$$

3.3.6. Results. In this section we present the performance and cost results of the proposed scheme under the data models discussed in Section 3.3.5. Performance is quantified with detection and false positive rates. Cost is quantified with the number of measurements needed for monitoring and fault localization. All results are average values over 100 realization under each parameter setting.

First, we test the scheme under varying sparsities of binary fault model. The corresponding results are shown in Fig. 3.14. Here we use networks of 300 nodes and 1772 links in average. Faulty links carry a value one and the other links carry a zero. As can be noted,

TABLE 3.1. Ratio Between Actual and Theoretical Samples Needed

Sparsity	$k \log(n/k)$	m (Actual Number of Samples)	$m\,(k \log{(n/k)})^{-1}$
2	13.6	26.2	1.93
4	24.4	37.1	1.52
6	34.1	51.6	1.51
8	43.2	61.6	1.42
10	51.8	78.0	1.51

the proposed scheme achieves near 100% detections and near 0% false positives for the range of sparsities tested.

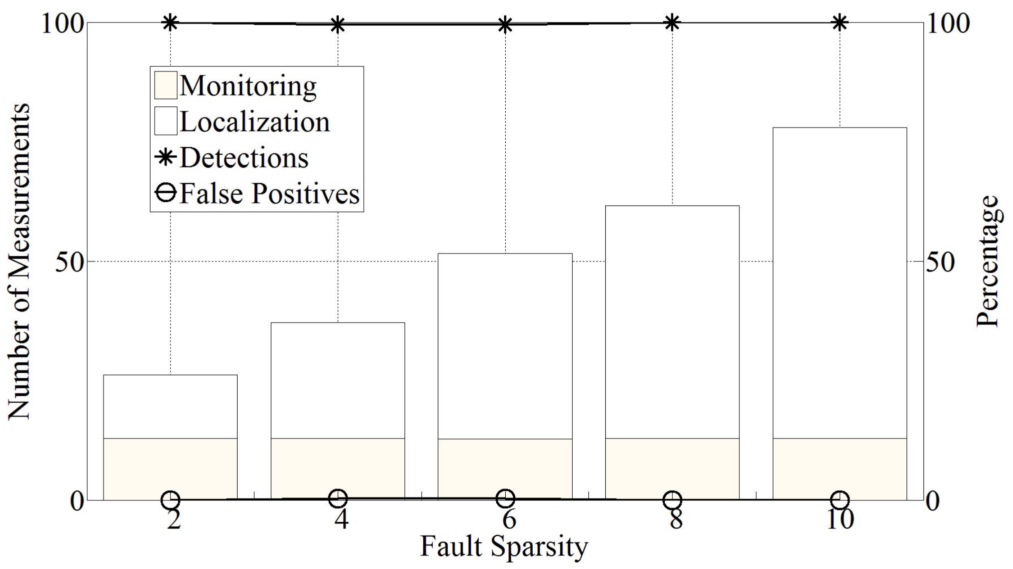


FIGURE 3.14. Effect of sparsity under the binary data model.

We compare the above results against the theoretical number of measurements needed $m = \mathcal{O}\,(k \log{(n/k)})$. For this we calculate $m\,(k \log{(n/k)})^{-1}$. Under a typical compressive sensing setup, this number is expected to be around 3.0. But as listed in Table 3.1, the proposed algorithm achieves a relatively smaller factor.

Next we consider the scalability of the scheme under the binary data model, as we increase the network size. Here we simulate five faulty links on each network. Fig. 3.15 shows the results, as the network size is increased. The figure shows the sizes of the networks in terms of the number of nodes and the average number of links. As the results indicate, the scheme continues to performs with near 100% detection and near 0% false positives throughout all the network sizes test. Notably even when the network size is grown exponentially, the number of measurements needed grew much slower.

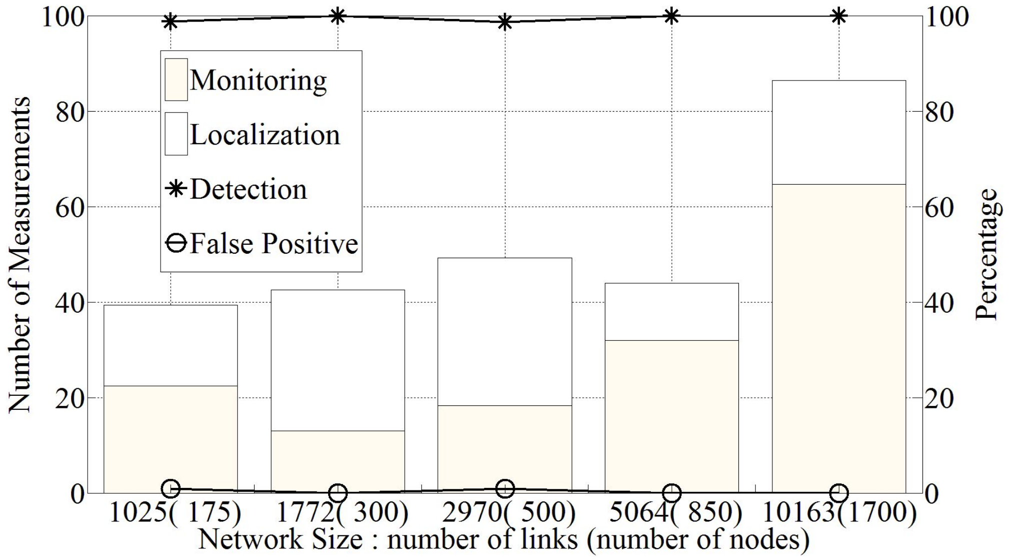


FIGURE 3.15. Scalability under the binary data model.

Next we test the scheme under the simple random model. As before, we begin by testing the performance of the scheme with varying fault sparsity. The networks tested in these experiments are of 300 nodes and 1772 links average. The faults had a strength ξ of 30dB. Performance and cost results are shown in Fig. 3.16. The results indicate a perfect 100% detection for all the tested cases, and a low false positives rate.

The scalability of the scheme under the simple random model is tested in the experiment whose results are shown in Fig. 3.17. In this experiment five faults with strength of 30dB is injected to the test networks. The network sizes are varied exponentially as shown in the figure. Although, as can be seen, the number of measurements needed grew slowly. The detection rate continues to remain at 100% while maintaining a low false positives rate.

We also test the effect of fault strength on the performance and cost. For this, we test the detection of five faulty links on a network of 300 nodes and 1772 links in average. The fault

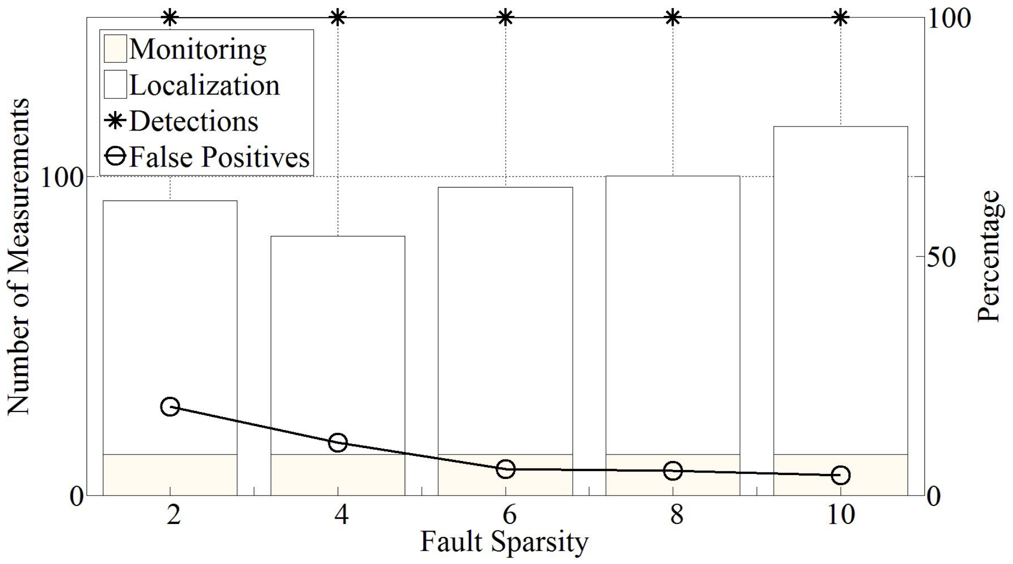


FIGURE 3.16. Effect of sparsity under the simple random model.

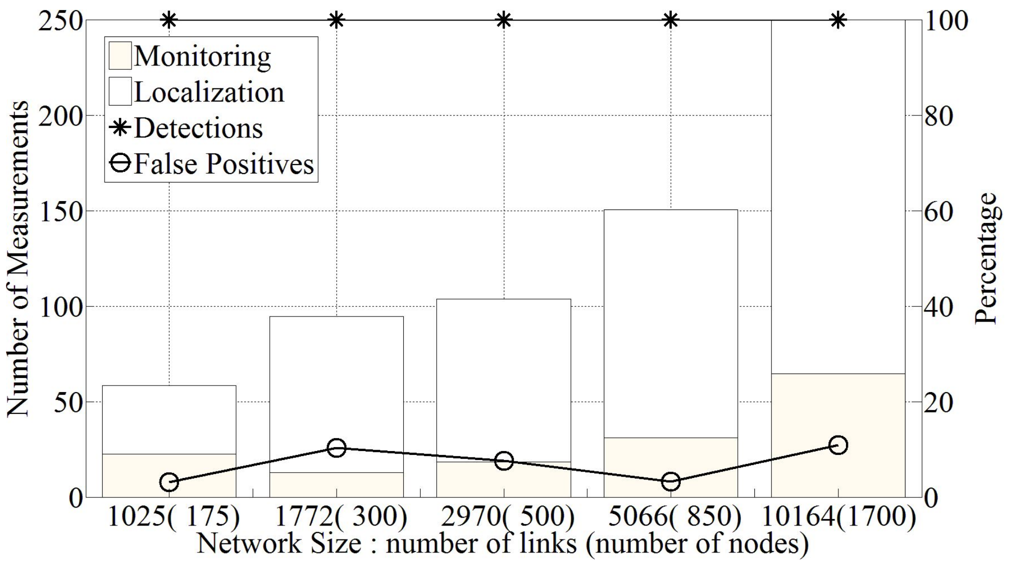


FIGURE 3.17. Scalability under the simple random model.

strength ξ is varied over the range 25dB to 45dB. The results shown in Fig. 3.18 indicate that the scheme achieves a near 100% detection rate and a very low false positives rate. Further, it can also be noted that the number of measurements needed to localizes faults slightly decreases as the strength of the faults increases.

Then we test the performance of the scheme under realistic data models. We begin with the loss data and test for the scalability of the proposed scheme. As discussed before network

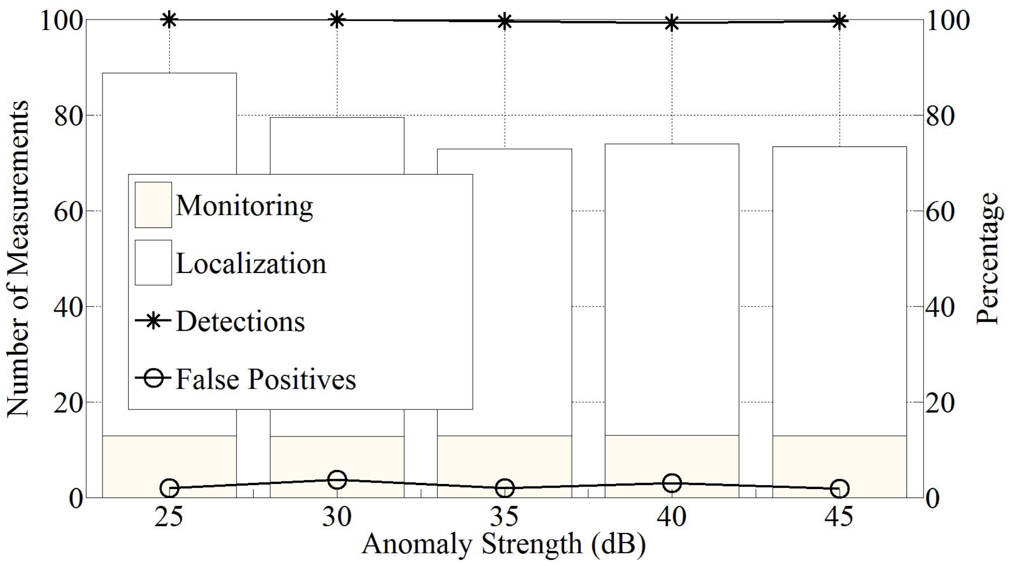


FIGURE 3.18. Effect of the strength of the faults.

losses are simulated with Gilbert-Elliott model calibrated with real measurements made on Planet-Lab infrastructure. As the results shown in Fig. 3.19 indicate, the proposed model achieves a high detection rate and low false positives rate. More importantly, even when the network size is grown exponentially, the number of measurements needed grew much slower indicating a high scalability.

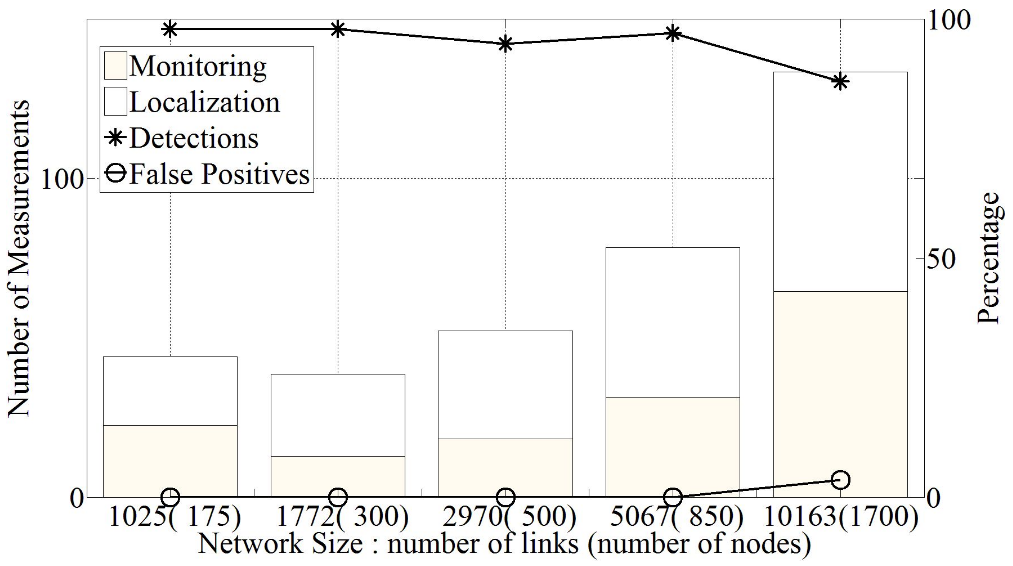


FIGURE 3.19. Scalability under the loss data model.

Then we repeat the experiment for the network link delay model. As discussed above, link delays are simulated using heavy tail distributions calibrated with measurements made on Planet-lab infrastructure. We continue to observe a high detection rate and a low false positives rate in the link delay model as well. Moreover, the number of measurements needed grew much slower, even when the network size is grown exponentially.

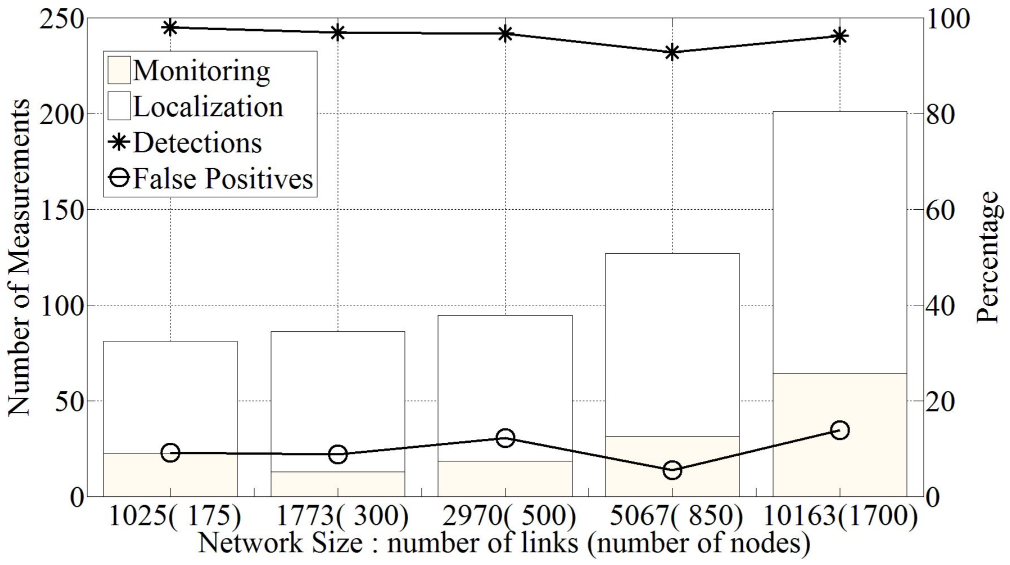


FIGURE 3.20. Scalability under link delay model.

Finally we take a closer look at the scalability of the proposed scheme under each data model. As the results in Fig. 3.21 shows, the number of measurements needed to localize faults grows nearly proportional to the logarithm of the number of links in the network. This indicates that the proposed scheme achieves the $m = \mathcal{O}\left(k \log\left(n/k\right)\right)$ scaling so far was not achieved with CS in a network setup. Such kind of a scaling promises significant cost savings for large networks.

3.3.7. CONCLUSIONS. An adaptive compressive sensing scheme for network tomography based fault localization was proposed. The scheme is tested on realistic network topologies using a few different data models. The scheme achieves a very high detection rate and a

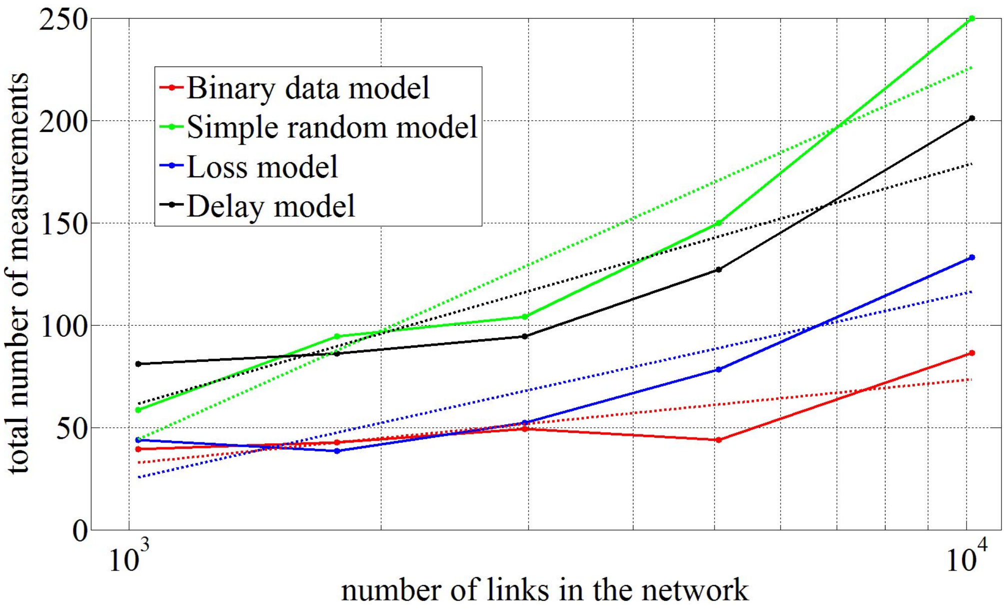


FIGURE 3.21. Scalability under link delay model.

very low false positive rate under all the tested data models. More importantly, the number of measurements the scheme required scale logarithmically with the number of links in the network. Further, the proposed three criteria for termination and the two rules for selecting the links for additional adaptive measurements, lead to a fault localization with a minimal number of additional measurements and assures a rapid localization process. Thus, the proposed scheme is efficient for monitoring and localizing faulty links of large networks in terms of accuracy, speed, instrumentation cost, and measurement traffic load.

3.3.8. LIMIT ON PATH LENGTH. Let σ^2 denote variance of path QoS values and σ_l^2 denote variance of link QoS values. Then for a path of h links $\sigma^2 = \sigma_l^2 h$. For a fault of magnitude ζ to cause a deviation beyond $n\sigma$ for a selected n

$$(\zeta - \gamma) > n\sigma = n\sigma_l\sqrt{h} \tag{3.27}$$

$$\therefore h < \left(\frac{\zeta - \gamma}{n\sigma_l}\right)^2 \tag{3.28}$$

3.3.9. Limit on Path Length Under Simple Random Data Model. Let the typical link level QoS values under simple random data model be distributed uniformly over the range $[0, \gamma]$. Then link level variance $\sigma_l^2 = \gamma^2/12$.

$$h < \frac{12(\zeta - \gamma)^2}{n^2\gamma^2} \tag{3.29}$$

$$h < \frac{12}{n^2}\left(\frac{\zeta - \gamma}{\gamma}\right)^2 = \frac{12}{n^2}(\xi - 1)^2 \tag{3.30}$$

3.3.10. Detection and False Positive Rates. Area under a normalized Gaussian bell over a range $[-x, +x]$ around the mode is given by $\mathrm{erf}(x/\sqrt{2})$. Therefore the detection rate for Fig. 3.7 is

$$\mathrm{erf}\left(\frac{1}{\sqrt{2}}\frac{\beta - \alpha}{\sigma}\right) + \frac{1}{2}\left(1 - \mathrm{erf}\left(\frac{1}{\sqrt{2}}\frac{\beta - \alpha}{\sigma}\right)\right) = \left(1 + \mathrm{erf}\left(\frac{1}{\sqrt{2}}\frac{\beta - \alpha}{\sigma}\right)\right)/2 \tag{3.31}$$

Similarly, the false positive rate is

$$\left(1 - \mathrm{erf}\left(\frac{1}{\sqrt{2}}\frac{\alpha}{\sigma}\right)\right)/2 \tag{3.32}$$

3.4. Additional Algorithms Developed

Here we present a few additional algorithms developed under this work.

3.4.1. Optimum Sampling Paths. This section presents an approach to select optimum sampling paths. This scheme is an alternative measurement scheme to the random path monitoring used in the prior sections. The presented scheme supports prioritized recovery. That is, the measurements are constructed such that faults localization occurs more rapidly over marked high priority regions. Further, the algorithm is designed to operate with realistic hub and spoke cum mesh core network structures. The monitoring probes are assumes to be placed in the leaf nodes.

Figure 3.22 shows an example map of three priority levels. The network resembles a typical communication network where the core is a mesh network and the service network has a hub and spoke structure. We use this map as a running example.

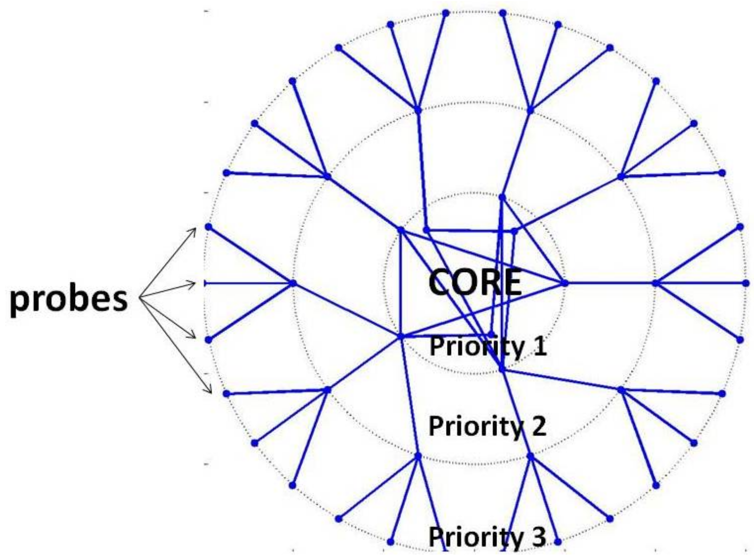


FIGURE 3.22. Sample network

Algorithm 3.23 shows the algorithm used herein. It begins by listing all end to end path measurements. The next step is to assign a score to each path. The score is based on the importance of the links covered. Note that it follows the priorities assigned in the map. Then the algorithm selects the paths. It can be terminated when the maximum number of paths desired is reached.

In order to compare performance of the scheme, three other path selection schemes are used.

(1) All core paths

(2) Weighted random paths

(3) Random paths

```
Obtain all end-to-end paths
Assign a score to paths based on the links it go thru
    Max for core links
    Min for edge links
Start with a path with maximal score
Find a most different path
    Hamming distance / incoherence
    Compare group-wise
```

FIGURE 3.23. Algorithm for optimum sampling paths

In "All core path" selection, all possible paths that go through the core network are selected. However, this does not lead to superior performance, as shown later. In "Weighted random paths" paths are selected at random, but a higher probability is assigned to paths with a higher priority. In the "Random paths" case paths are selected at random.

Next we proceed to compare performance. Figure 3.24 shows the detection of a single fault using each path selection scheme at each priority level. In this experiment the number of paths selected are varied. As can be noted, the proposed scheme shows the most superior performance. We also study the false positive rate of the proposed scheme in detecting 1-

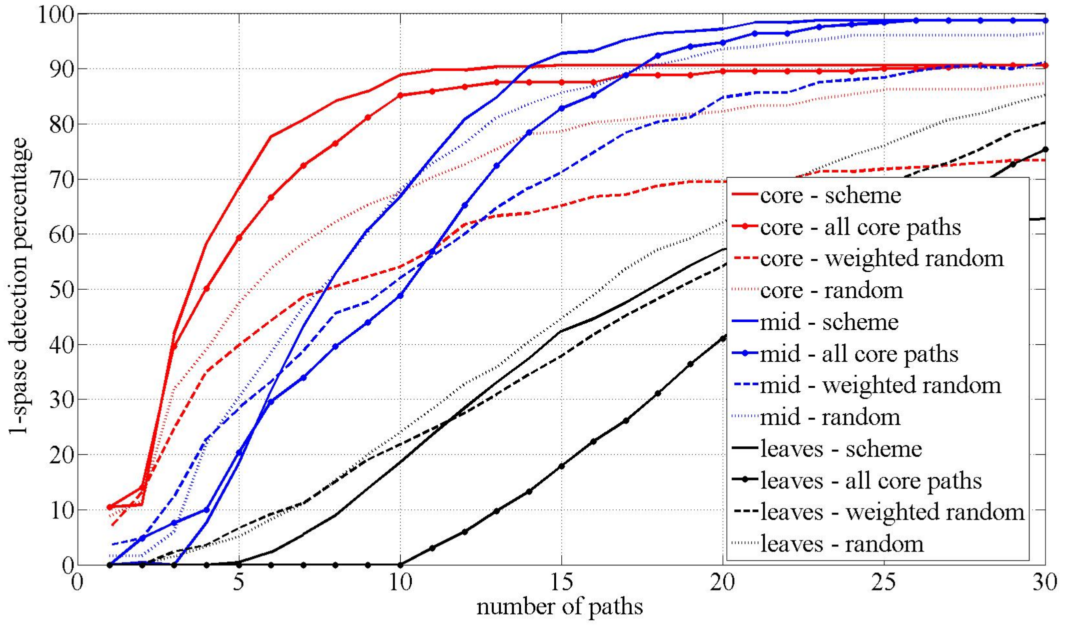


FIGURE 3.24. Detecting 1-sparse faults

sparse faults. The results are shown in Fig. 3.25. The results indicate that the false positives caused by the algorithm is in the same neighborhood as the other schemes. We also test

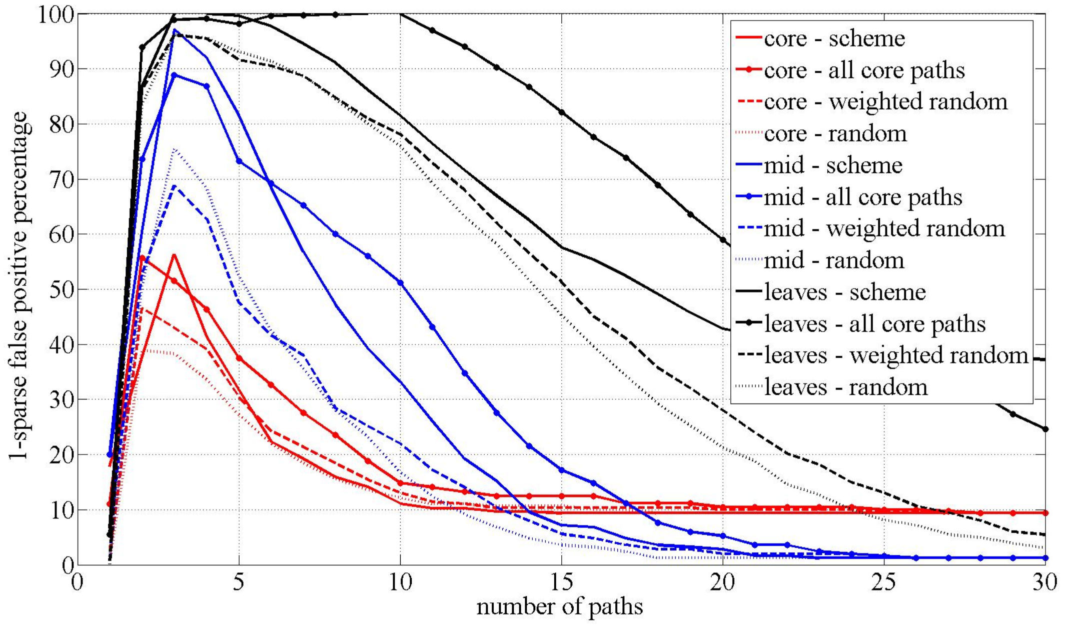


FIGURE 3.25. False positives in detecting 1-sparse

the performance of the algorithm in detecting realistic losses. The corresponding results are presented in Fig. 3.26. These results are comparable to the detection results obtained earlier.

In essence, the presented optimum sampling scheme selects the best set of measurements to optimize fault localization of a prioritized network.

3.4.2. EVOLUTION OF THE FAULT LOCALIZATION ALGORITHM. Here we briefly review the evolution of the fault localization algorithm presented in prior sections.

The first algorithm reduces the measurement systems and applies compressive sensing to localize faults. But it does not perform iterative solving. Thus, it will require a good measurement systems such as build in Section 3.4.1. Otherwise it requires a higher number of measurements compared to algorithms presented here forth.

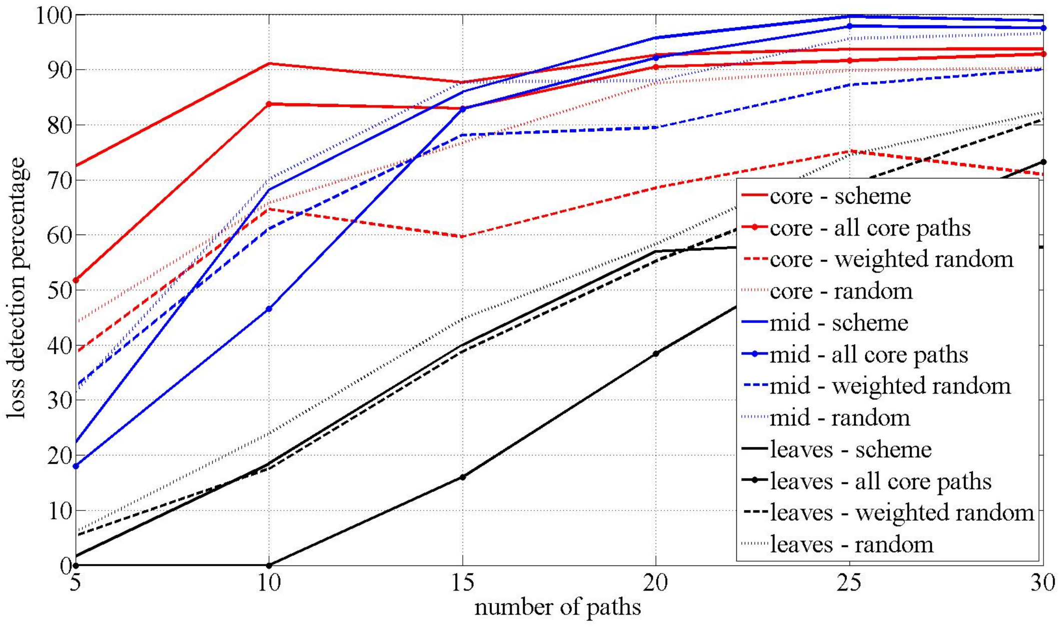


FIGURE 3.26. Loss detection

When access to any network link is available fault localizations can be immediately verified. Algorithm 3.28 is designed for such scenarios. Notably this algorithm always yields zero fault positives. Also the number of measurements needed for this algorithms is far less. However, to realize the algorithm on a network require a significant amount of instrumentation.

When access to a limited set of links are available, Algorithm 3.29 can be used. Although it require less instrumentation than the previous algorithm, it has degraded performance.

Algorithm 3.30 is designed for tomographic measurement environments. This algorithm forms the foundation of the localization algorithms presented in prior sections. This algorithm is endowed with high accuracy and also require less instrumentations. However it is vulnerable to noise. But the next generations of this algorithm became resilient to noise.

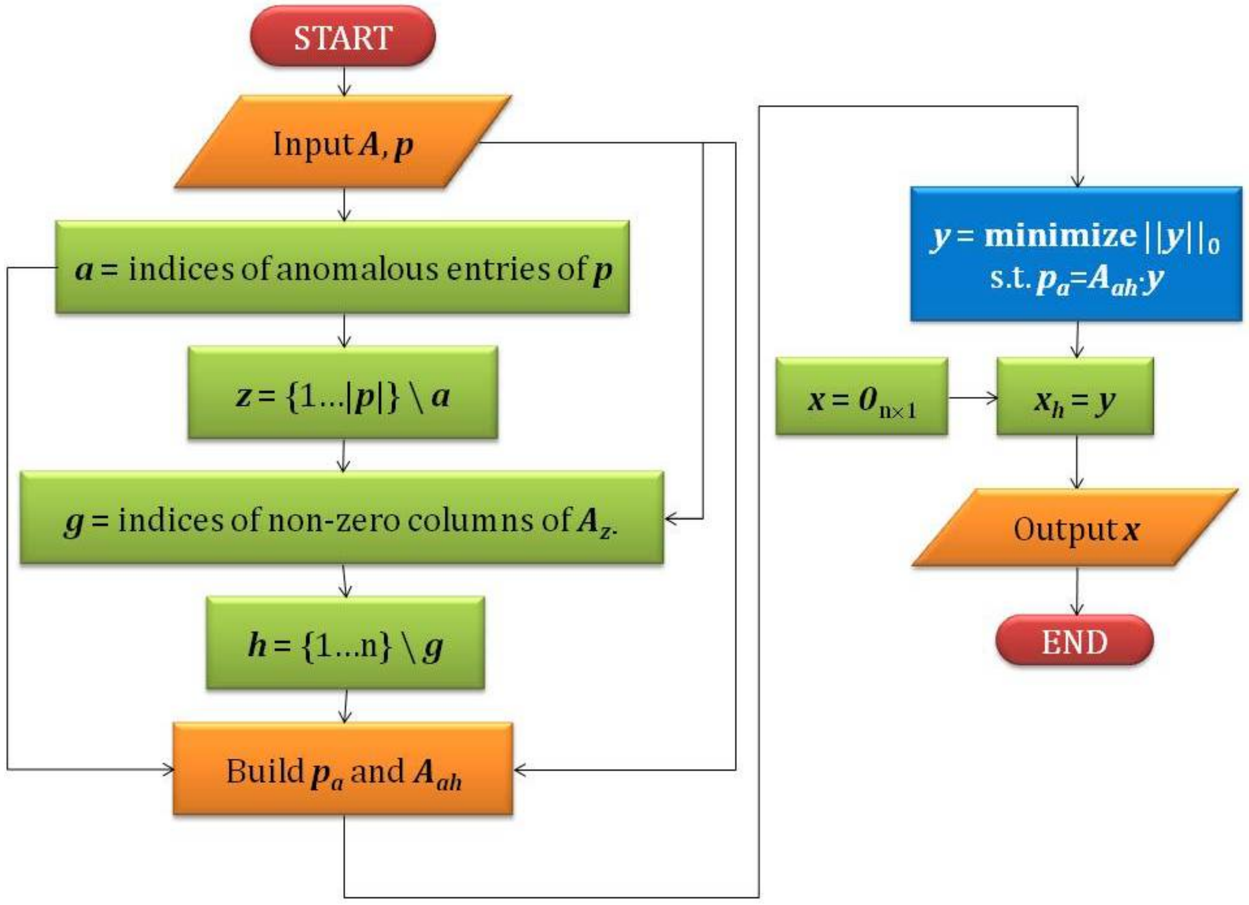

START
Input A, p
a = indices of anomalous entries of p
$z = \{1 \ldots |p|\} \setminus a$
g = indices of non-zero columns of $A_{z\cdot}$
$h = \{1 \ldots n\} \setminus g$
Build p_a and A_{ah}
y = minimize $||y||_0$ s.t. $p_a = A_{ah} \cdot y$
$x = 0_{n \times 1}$
$x_h = y$
Output x
END

FIGURE 3.27. Fault localization algorithm - version 1

3.5. Conclusions

In this chapter we presented our work on network monitoring and fault localization. We used compressive sensing to resolve network measurements. To achieve a high detection accuracy from a minimal number of measurements, we developed adaptive algorithms that quickly converge on the faulty links. Further, these algorithms relied only on the available information unlike CS algorithms found in literature. We tested our algorithms on realistic network topologies with realistic data models for QoS parameters such as link delays and losses, for accuracy, scalability and cost.

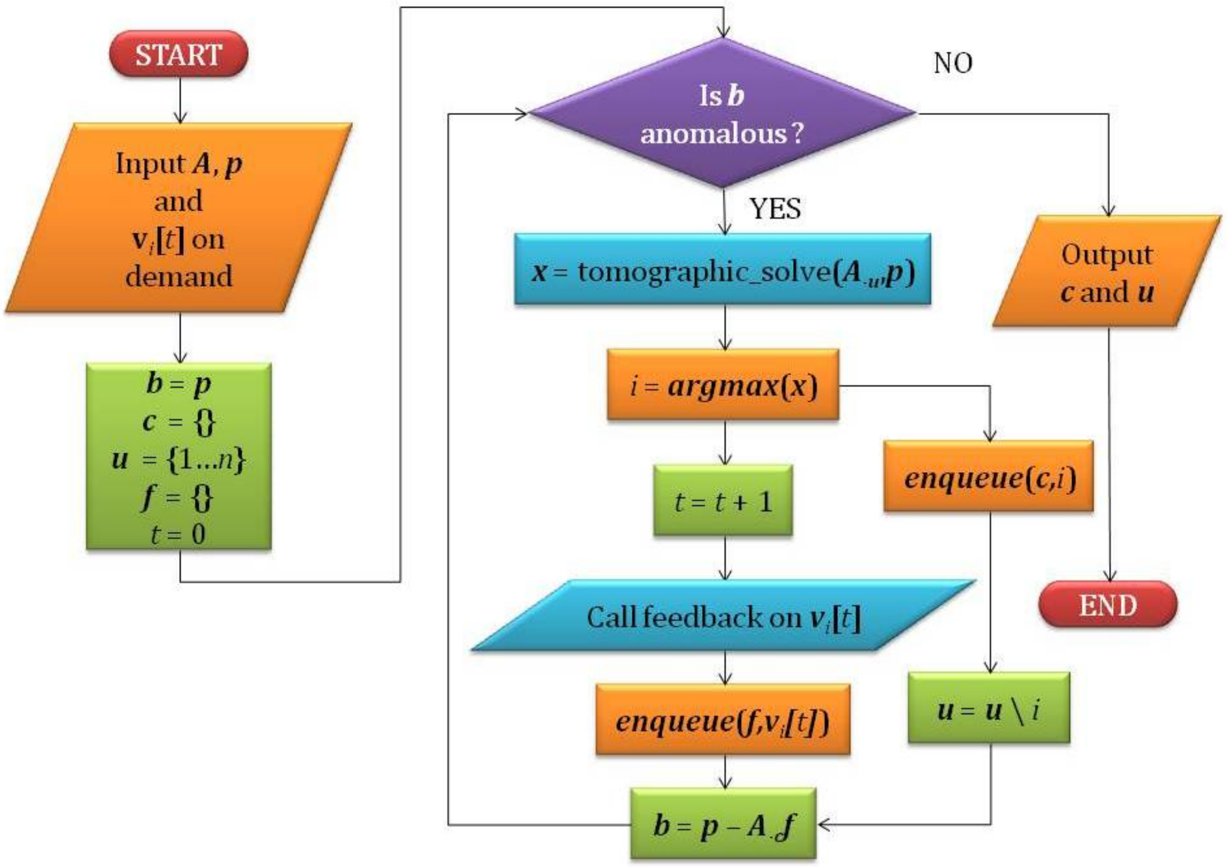


FIGURE 3.28. Fault localization algorithm - version 2

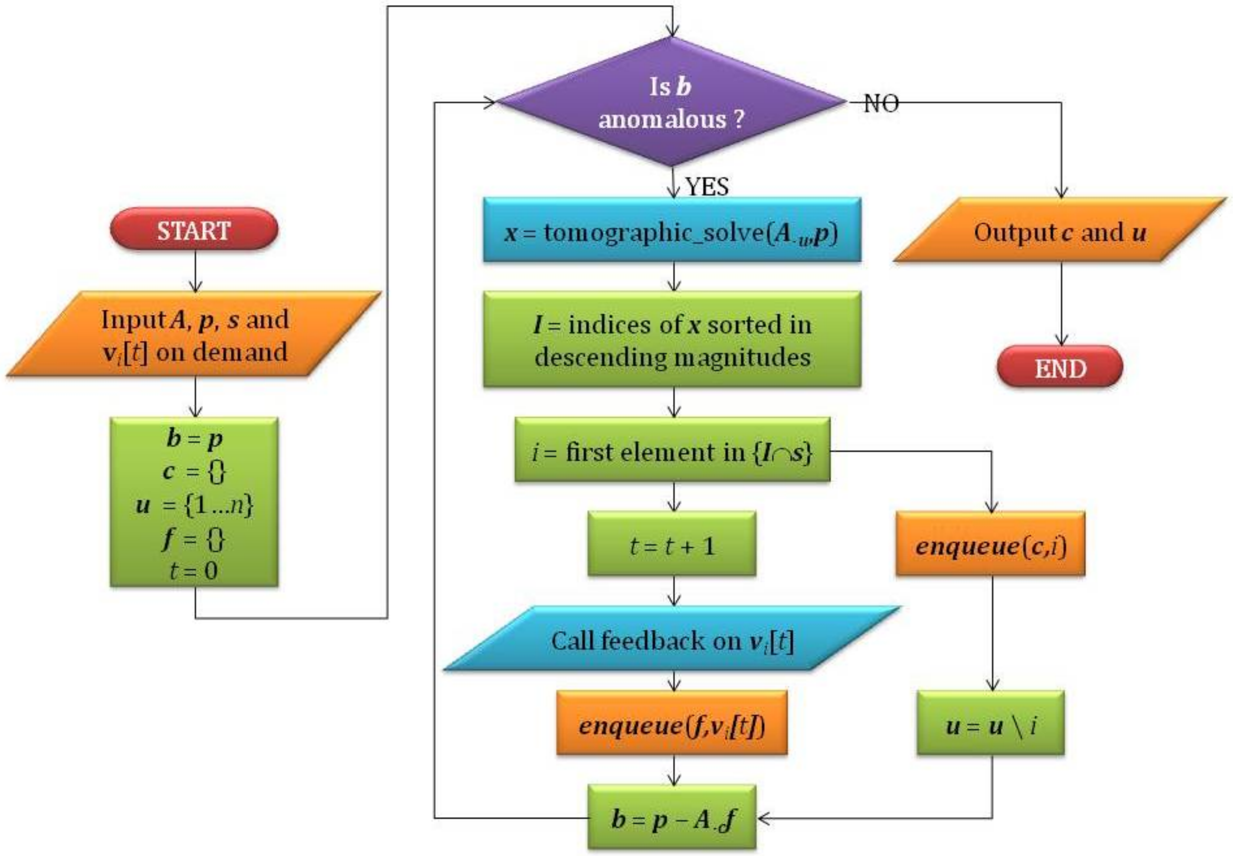


FIGURE 3.29. Fault localization algorithm - version 3

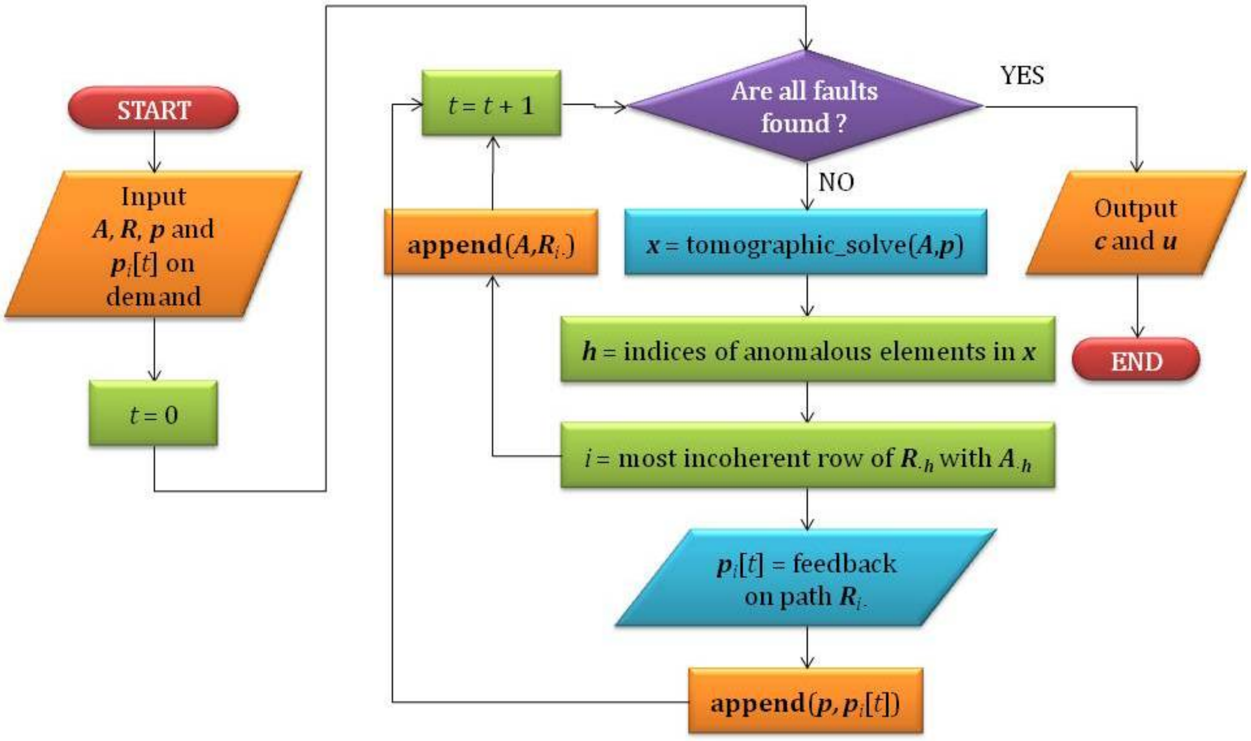


FIGURE 3.30. Fault localization algorithm - version 4

CHAPTER 4

Data Recovery

4.1. Introduction

In this chapter we consider data recovery techniques. We begin with compressive sensing. Authors of [21] established recovery bounds for compressive sensing when the sampling measure is orthogonalizational. We extend this work to any sampling measure and quantify the effect of mismatch therein. We also review the implications of the generalizations. This generalization is important as it cannot be guaranteed that a practical sampling scheme would actually be orthogonalizational. Then we apply compressive sensing for phenomena discovery in Wireless Sensor Networks (WSNs). The findings here show a notable amount of cost savings that can achieved via compressive sensing in WSNs.

Next we consider wavelet based data recovery. We develop a distributed compression scheme that reduce the communication cost of WSNs to deliver information. This scheme uses two-dimensional wavelet transform and only require a limited set of sensors scattered at random locations. We demonstrate the performance of the scheme on tracking a chemical plume with this limited number of sensors. In an extension of the same work, matrix completion is used to recover data when the sampling is extremely low (5%). To overcome the lack of smoothness in matrix completion, compressive sensing is used over matrix completed data with Discrete Cosine basis as the sparsity domain.

4.2. Performance Bounds for Sparse Signal Recovery from Random Samples

We consider the problem of reconstructing a signal from a small number of its samples. The signal is assumed to have a sparse representation in a known basis and the sampling points are selected at random according to a probability measure. However, unlike previous work, the basis elements are not orthogonal with respect to the measure from which the sample points are selected. In other words, there is mismatch between the sampling measure and the orthogonality measure. We consider two cases: (1) *non-uniform recovery*: Given each fixed s-sparse support, we establish conditions that guarantee the recovery of the signal from a random realization of sample points with high probability, and (2) *uniform recovery*: Given a fixed realization of sample points, we establish conditions under which recovery is guaranteed for any s-sparse signal with high probability. In each case, lower bound on the number of measurements is a monotonically increasing function of the extent of mismatch. We specifically bound the extent of mismatch in the case where sparse signals in the Fourier basis are sampled at random according to distributions from the natural exponential family.

4.2.1. Introduction. Let $\{\psi_k(t)\}_{k=1}^N$ be an orthonormal system of complex functions on $\mathcal{D} \subset \mathbb{R}$ with respect to measure ν that is

$$\int_{\mathcal{D}} \psi_j(t)\overline{\psi_k(t)}\,\mathrm{d}\nu(t) = \delta_{jk} \quad j,k \in \{1,\dots,N\}, \tag{4.1}$$

where δ_{jk} is the Kronecker delta and $\mathcal{D}$ is endowed with measure ν, and $\nu(\mathcal{D}) = 1$. Assume basis elements $\psi_k(t), k = 1,\dots,N$ are bounded as

$$\|\psi_k\|_\infty = \sup_{t\in\mathcal{D}} |\psi_k(t)| \leq K,\ K \in \mathbb{R}^+. \tag{4.2}$$

Let $y(t)$ be a generic signal that has an s-sparse representation in $\{\psi_k(t)\}_{k=1}^N$. That is,

$$y(t) = \sum_{k=1}^{N} \psi_k(t) b_k = \mathbf{\Psi}(t)\mathbf{b}, \tag{4.3}$$

where $\mathbf{\Psi}(t) = [\psi_1(t),\dots,\psi_N(t)]$, and $\mathbf{b} = [b_1,\dots,b_N]^T \in \mathbb{C}^N$ is s-sparse. Let $t_1,\dots,t_m$ be a sequence of i.i.d. realizations w.r.t. a probability measure $\bar{\nu}$. Let $\mathbf{y} = [y(t_1),\dots,y(t_m)]^T$

and $\mathbf{A}$ be an $m \times N$ matrix whose $(l,k)^{\text{th}}$ entry $\mathbf{A}_{l,k} = \psi_k(t_l)$. Then, we can express $\mathbf{y}$ as

$$\mathbf{y} = \mathbf{A}\mathbf{b}. \tag{4.4}$$

We are interested in deriving bounds on the number of samples m required to guarantee the recovery of $y(t)$ from a random set of samples $\mathbf{y} = [y(t_1), \ldots, y(t_m)]^T$, with high probability.

In [21], the authors have investigated this question in the case where the sampling measure $\bar{\nu}$ is identical to the orthogonality measure ν. They have studied two scenarios: *non-uniform recovery* and *uniform recovery*. Given each fixed s-sparse support, non-uniform recovery guarantees the recovery of the signal from a random realization of sample points with high probability. Given a fixed realization of sample points, uniform recovery guarantees the recovery of any s-sparse signal with high probability. In [64], the authors present improved guarantee bounds for the uniform recovery case considered in [21].

In this paper, we study the case where there is a mismatch between the sampling measure $\bar{\nu}$ and the orthogonality measure ν, and derive sufficient conditions for uniform and non-uniform recoveries. The lower bound on the number of measurements needed for recovery is a monotonically increasing function of the extent of mismatch. We specifically bound the extent of mismatch in case where sparse signals in the *Fourier* basis are sampled at random according to distributions from the natural exponential family. We report explicit bounds for three distributions: *Exponential*, *Normal*, and *Gamma*. As discussed in [65] many scenarios exist where sampling measures are not orthogonalizational. This work provides the theoretical foundation for recovery guarantees for such instances.

4.2.2. Non-Uniform Recovery. Here we study the recovery of sparse signals with a fixed but unknown support.

Theorem 4.2.1. *Let* $\mathbf{b} \in \mathbb{C}^N$ *be an arbitrary* s*-sparse signal with fixed but unknown support* S $(|S| = s)$*. Let* $\{t_i\}_{i=1}^m$ *be a sequence of i.i.d. draws w.r.t. the probability measure* $\bar{\nu}$ *and* $\mathbf{A} \in \mathbb{C}^{m\times N}$ *be a matrix whose* $(l,k)^{th}$ *entry* $\mathbf{A}_{l,k} = \psi_k(t_l)$*. Then,* $\mathbf{b}$ *can be successfully recovered from* $\mathbf{y} = \mathbf{A}\mathbf{b}$ *with probability at least* $(1-\epsilon)$ *if*

$$m \geq 8\sqrt[4]{2}sK^2\kappa^2 \log\left(\frac{3(N-s)2^{3/4}}{\epsilon}\right)\log\left(\frac{3\left(N^2+s\right)}{\sqrt[4]{2}\epsilon}\right). \tag{4.5}$$

where $\kappa = \sqrt{\frac{D^2}{4}+1+Q} + \frac{Q}{D} + \frac{D}{2}$, $D = 2^{15/8}\sqrt{s/me}K$, $Q = \|\Delta\|_2$ *and*

$$\Delta = \begin{pmatrix} 0 & \rho_{12} & \cdots & \rho_{1N} \\ \rho_{2,1} & 0 & \cdots & \rho_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{N1} & \rho_{N2} & \cdots & 0 \end{pmatrix} \tag{4.6}$$

where

$$\rho_{ij} = \int_D \psi_j(t)\overline{\psi_i(t)}\,\mathrm{d}\bar{\nu}(t) \tag{4.7}$$

for $i = 1,\ldots,N, j = 1,\ldots,N,$ *and* $i \neq j$.

Proof. Steps of the proof is similar to those in the proof of [21, Theorem 4.2]. In Appendix 4.2.6.1, the steps that are required to incorporate the effects of mismatch into the proof are presented. □

Remark 1. *The lower bound in* (4.44) *is similar to the bound derived in [21], except in* κ*. The effect of mismatch between orthogonalization measure* ν *and the sampling measure*

$\bar{\nu}$ appears in κ via Q. The value of Q depends on ρ_{ij}s which capture the extent of non-orthogonality of the basis elements $\psi_j(t)$ w.r.t. the sampling measure $\bar{\nu}$, as shown in (4.7). *In the mismatch-free case, $\kappa = \sqrt{D^2/4+1}+D/2$ and for us $\kappa = \sqrt{D^2/4+1+Q}+Q/D+D/2$. The extent of mismatch will in turn determine the overhead in the number of measurements to guarantee perfect recovery. In the special case where ν and $\bar{\nu}$ coincide, ρ_{ij} and subsequently Q vanish, and the bound in* (4.44) *reduces to that obtained in [21]. Later in Section 4.2.4, we present a case study where the mismatch is quantified and explicitly bounded.*

4.2.3. Uniform Recovery. We now study the recovery of any sparse signal with a sparsity limit.

Restricted Isometry Property (*RIP*) of $\mathbf{A}$ is a commonly used sufficient condition for $\mathbf{A}$ to recover $\mathbf{b}$ from $\mathbf{y} = \mathbf{A}\mathbf{b}$ (see, e.g. [11]). We say $\mathbf{A}$ satisfies RIP with *Restricted Isometry Constant* (*RIC*) γ_s, or in short it satisfies γ_s-RIP, if

$$(1-\gamma_s)\|\mathbf{b}\|_2^2 \le \|\mathbf{A}\mathbf{b}\|_2^2 \le (1+\gamma_s)\|\mathbf{b}\|_2^2,\ \forall \mathbf{b} \in \Gamma_s \tag{4.8}$$

where Γ_s is the set of all s-sparse vectors in $\mathbb{C}^N$. The RIP can be written as

$$|||\tilde{\mathbf{A}}^*\tilde{\mathbf{A}} - \mathbb{I}|||_s \le \gamma_s \tag{4.9}$$

where $\tilde{A} = \frac{1}{\sqrt{m}}\mathbf{A}$, and the operator semi-norm $|||\cdot|||_s$ for s-sparse unit vectors is defined as

$$|||\mathbf{B}|||_s = \sup_{\|\mathbf{z}\|_0 \le s, \|\mathbf{z}\|_2 = 1} |\langle \mathbf{B}\mathbf{z}, \mathbf{z}\rangle| \tag{4.10}$$

for $\mathbf{B} \in \mathbb{R}^{m\times N}$.

If A satisfies γ_{2s}-RIP with $\gamma_{2s} < 1/3$, then every s-sparse $\mathbf{b}$ is recoverable from $\mathbf{y} = \mathbf{A}\mathbf{b}$ using a linear program [11]. If RIP is satisfied probabilistically, then recovery is guaranteed only probabilistically. The probability ϵ with which γ_s-RIP is not satisfied is given by

$$\epsilon = \mathbb{P}\left[|||\tilde{\mathbf{A}}^*\tilde{\mathbf{A}} - \mathbb{I}|||_s \geq \gamma_s\right]. \tag{4.11}$$

THEOREM 4.2.2. *Let $\mathbf{b} \in \mathbb{C}^N$ be an arbitrary s-sparse signal. Let $\{t_i\}_{i=1}^{\tilde{m}}$ be a sequence of i.i.d. draws w.r.t. the probability measure $\bar{\nu}$ and $\mathbf{A} \in \mathbb{C}^{\tilde{m}\times N}$ be a matrix whose $(l,k)^{th}$ entry $\mathbf{A}_{l,k} = \psi_k(t_l)$. Then, $\mathbf{b}$ is recoverable from $\mathbf{y} = \mathbf{Ab}$ with a probability at least $(1-\epsilon)$ if*

$$\frac{\tilde{m}}{\ln(10\tilde{m})} \geq \frac{c_1\tilde{\kappa}^2}{\gamma_s^2} s \ln^2(100s)\ln(4N)\ln\left(\frac{7}{\epsilon}\right) \tag{4.12}$$

where

$$\tilde{\kappa} = \sqrt{1 + \frac{D^2}{\tilde{m}} + \tilde{Q}} + \frac{\tilde{Q}\sqrt{\tilde{m}}}{2D} + \frac{D}{\tilde{m}} \tag{4.13}$$

for $D = C\sqrt{2\beta}K\sqrt{s}\ln(100s)\sqrt{\ln(4N)\ln(10\tilde{m})}$ for some constants C and β, and $\tilde{Q} = |||\Delta|||_s$.

Before we prove *Theorem* 4.3.3, two remarks are in order.

REMARK 2. *The bound in* (4.95) *is similar to the bound obtained in [21], except in $\tilde{\kappa}$. The effect of mismatch between the sampling measure $\bar{\nu}$ and the orthogonalization measure ν appears in $\tilde{\kappa}$ via $\tilde{Q}$. In the mismatch-free case, $\tilde{\kappa} = \sqrt{D^2/\tilde{m} + 1} + D/\tilde{m}$ instead of $\tilde{\kappa} = \sqrt{D^2/\tilde{m} + 1 + \tilde{Q}} + D/\tilde{m} + \tilde{Q}\sqrt{\tilde{m}}/(2D)$. The extent of mismatch affects the number of measurements needed to guarantee recovery with a certain probability. When ν and $\bar{\nu}$ coincide, $\tilde{Q}$ vanishes and* (4.95) *reduces to that in [21]. Later in Section 4.2.4, we will bound the extent of mismatch for the case where Fourier sparse signals are sampled at random according to distributions from the natural exponential family.*

REMARK 3. *Under the additional condition* $\mathbb{E}|||\mathbf{A} * \mathbf{A} - \mathbb{I}|||_s \leq 8\gamma_s/9$, *the bound in* (4.95) *can be tightened to*

$$\tilde{m} \geq \frac{c_2 D_1}{\gamma_s^2} s K^2 \ln\left(\frac{1}{\epsilon}\right) \tag{4.14}$$

where $c_2 \leq 456$ *and* $D_1 = \frac{1+\tilde{Q}/76}{1+9\tilde{Q}/\gamma_s^2}$.

PROOF. See Appendix 4.2.6.2. □

4.2.4. CANONICAL EXAMPLE. We consider a signal $y(t)$ that has a sparse representation in *Fourier* domain, i.e., $\psi_k(t) = e^{\imath k \omega_0 t}$, for some fundamental frequency ω_0, and $\|\mathbf{b}\|_0 \leq s$. We assume the signal $y(t)$ is sampled in time at random according to a probability measure $\bar{\nu}$ corresponding to the natural exponential family with *characteristic function*

$$\varphi_{\bar{\nu}}(t) = \exp\{\mathcal{A}(\eta + t) - \mathcal{A}(\eta)\} \tag{4.15}$$

where $\mathcal{A}(\eta)$ is the so-called *log-partition function* and η is the so-called *natural parameter.*

We quantify the extent of mismatch between the sampling measure $\bar{\nu}$ and the orthogonality measure ν for the Fourier basis by $Q = \|\Delta\|_2$, with Δ defined in (4.6). We have the following theorem.

THEOREM 4.2.3 (Bounding mismatch for non-uniform recovery). *Let* $Q = \|\Delta\|_2$ *with* Δ *as in* (4.6) *and* $\rho_{jk} = \int_{\mathcal{D}} \psi_j(t)\overline{\psi_k(t)} \mathrm{d}\bar{\nu}(t)$ *with* $\bar{\nu}$ *being a measure of the natural exponential family with characteristic function* (4.15), *and* $\psi_k(t) = e^{\imath k \omega_0 t}$. *Then*

$$\exp\{\mathcal{A}(\eta + \imath\omega_0) - \mathcal{A}(\eta)\} \leq Q$$
$$\leq \sum_{k \neq N/2} \exp\left\{\mathcal{A}\left(\eta + \imath(\frac{N}{2} - k)\omega_0\right) - \mathcal{A}(\eta)\right\} \tag{4.16}$$

for $k = 1, \ldots, N$.

PROOF. Using *Gershgorin circle theorem*, and norm equivalence, we have

$$\max_{j,k} |\rho_{jk}| \leq Q \leq \max_j \sum_{k \neq j} |\rho_{jk}| \tag{4.17}$$

where

$$\rho_{jk} = \int_{\mathcal{D}} e^{\imath(j-k)\omega_0 t} \mathrm{d}\bar{\nu}(t). \tag{4.18}$$

Note that (4.18) is the characteristic function of $\bar{\nu}(t)$ at $(j-k)\omega_0$. Therefore

$$\rho_{jk} = \exp\left\{\mathcal{A}\left(\eta + \imath(j-k)\omega_0\right) - \mathcal{A}(\eta)\right\}. \tag{4.19}$$

Since $\mathcal{A}(\eta)$ is a logarithmic function, $\max_{j,k} |\rho_{jk}|$ occurs when $|j-k| = 1$ and $\max_j \sum_{k \neq j} |\rho_{jk}|$ occurs at $j = \lceil N/2 \rceil$. This completes the proof. □

REMARK 4. *For uniform recovery, we quantify the extent of mismatch by* $Q = |||\Delta|||_s$. *Then, similar bounds as in* (4.16) *for* Q *hold with* $k = 1, \ldots, s/2, N - s/2, \ldots, N$.

We now explicitly work out the bounds in (4.16) for three distributions from the natural exponential family. For the *exponential* distribution, the natural parameter $\eta = -\lambda$ and log-partition function $\mathcal{A}(\eta) = -\ln -\eta$, where λ is the *rate*. Then,

$$\frac{\lambda}{\sqrt{\lambda^2 + \omega_0^2}} \leq Q \leq \sum_{k \neq \frac{N}{2}} \frac{\lambda}{\sqrt{\lambda^2 + (\frac{N}{2} - k)^2 \omega_0^2}}. \tag{4.20}$$

For the *Normal* distribution with *mean* μ and *variance* σ^2, we have

$$\eta = \left[\frac{\mu}{\sigma^2}, -\frac{1}{2\sigma^2}\right]$$

$$\mathcal{A}(\eta) = -\frac{\eta_1^2}{4\eta_2^2} - \frac{1}{2}\ln(-2\eta_2)$$

and,

$$\exp\left\{-\frac{1}{2}\sigma^2\omega_0^2\right\} \le Q \le \sum_{k\neq\frac{N}{2}} \exp\left\{-\frac{1}{2}\sigma^2\left(\frac{N}{2}-k\right)^2\omega_0^2\right\}. \tag{4.21}$$

Similarly, for the *Gamma* distribution, we have

$$\eta = \left[\alpha - 1, -\frac{1}{\theta}\right] \quad \text{and}$$

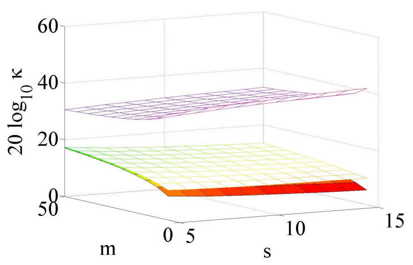
$$\mathcal{A}(\eta) = \ln\Gamma\left(\eta_1+1\right) - \left(\eta_1+1\right)\ln\left(-\eta_2\right)$$

where α is the *shape* parameter and θ is the *scale* parameter, and (4.16) becomes

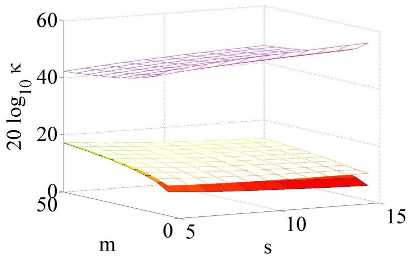
$$\left(1+\omega_0^2\theta^2\right)^{-\alpha/2} \le Q \le \sum_{k\neq\frac{N}{2}} \left(1+\left(\frac{N}{2}-k\right)^2\omega_0^2\theta^2\right)^{-\alpha/2}. \tag{4.22}$$

Figure 4.1 show the effect of sampling mismatch on κ in non-uniform and uniform recovery for Exponential distribution in Figs. 4.1(a) and 4.1(d), Normal distribution in Figs. 4.1(b) and 4.1(e), and Gamma distribution in Figs. 4.1(c) and 4.1(f). In each figure, the upper and lower bound on κ in the presence of mismatch are plotted (meshed surface), along with the κ in the mismatch-free case (solid surface) obtained in [21]. These plots show the increase of κ due to the mismatch between the sampling measure and the orthogonality measure for the sparsity basis. An increase in κ means more measurements may be needed in the mismatched case, compared to the mismatch-free case, to guarantee signal recovery with the same probability.

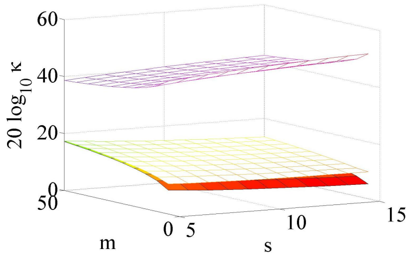
4.2.5. Conclusion. In this paper we derived sufficient conditions for uniform and non-uniform recovery of a sparse signal from its random samples for the case where there is a mismatch between the sampling measure and the orthogonality measure for the sparsity



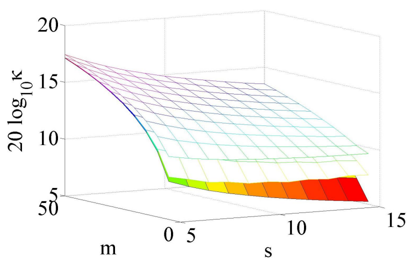
(A) Exponential with $\lambda = 0.5$ (non-uniform recovery)



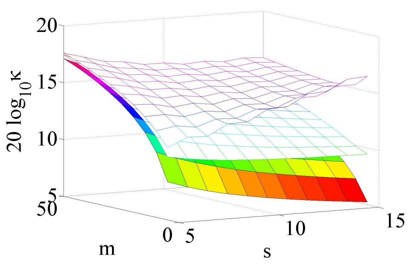
(B) Normal with $\sigma = 0.1$ (non-uniform recovery)



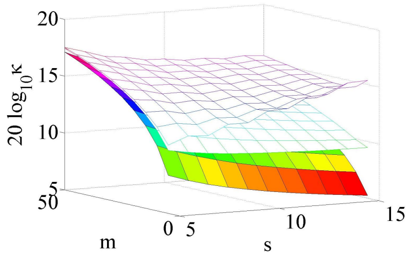
(C) Gamma with $\alpha = 0.5, \theta = 1.0$ (non-uniform recovery)



(D) Exponential with $\lambda = 0.5$ (uniform recovery)



(E) Normal with $\sigma = 0.1$ (uniform recovery)



(F) Gamma with $\alpha = 0.5, \theta = 1.0$ (uniform recovery)

FIGURE 4.1. Upper and lower bounds for κ (in dB), for different values of s and m, in non-uniform recovery (subplots (a)–(c)) and uniform recovery (subplots (d)–(f)). The bounds are plotted for three different random sampling distributions: Exponential, Normal, and Gamma. The plot of κ in the mismatch-free case is also shown for comparison (bottom most surface in all subplots). In all cases the signal dimension $N = 1000$.

basis. We derive explicit bounds for the extent of mismatch when Fourier sparse signals are sampled at random according to distributions from the natural exponential family.

4.2.6. Appendices.

4.2.6.1. *Proof of Theorem 4.3.1.* Let a_l denote the l^{th} column of $\mathbf{A}$. Let $\mathbf{A}_S$ denote the submatrix of $\mathbf{A}$, whose column indices are in S that is, $\mathbf{A}_S = [a_{l_1}, a_{l_2}, \ldots, a_{l_s}]$, where $l_i \in S$ for $i = 1, 2, \ldots, s$. Further, let $\mathbf{b}_S = [b_{l_1}, b_{l_2}, \ldots, b_{l_s}]^T$.

From [21, Corollary 2.8] (also see [58, 59]) if $\mathbf{b}_S$ satisfies

$$\left|\left\langle \mathbf{A}_S^{\dagger} a_l, sgn\left(\mathbf{b}_S\right)\right\rangle\right| < 1, \ \forall l \notin S \tag{4.23}$$

where $sgn(\cdot)$ denotes the signum function and $\mathbf{A}_S^{\dagger}$ denotes the Moore-Penrose pseudo inverse of $\mathbf{A}_S$, then $\mathbf{b}$ can be recovered from $\mathbf{y}$ with probability at least $1-\epsilon$, using a linear program. Using union bound and [21, Corollary 6.10], the lower bound ϵ on the failure probability in recovering $\mathbf{b}$ can be bounded as

$$\begin{aligned}\epsilon &\leq \mathbb{P}\left[\max_{l\notin S}\left|\left\langle \mathbf{A}_S^{\dagger}a_l, sgn\left(\mathbf{b}_S\right)\right\rangle\right| \geq 1\right] \qquad (4.24)\\ &\leq (N-s)2^{3/4}\exp\left(-\frac{1}{2}\left(\frac{\sqrt{s}\tau}{1-\gamma}\right)^{-2}\right)\\ &+\mathbb{P}\left[\left\|\tilde{A}_S^*\tilde{A}_S-\mathbb{I}\right\|_2 > \gamma\right]\\ &+\mathbb{P}\left[\max_{l\notin S}\sqrt{\sum_{j\in S}\left|\langle\tilde{a}_l,\tilde{a}_j\rangle\right|^2} > \sqrt{s}\tau\right] \qquad (4.25)\end{aligned}$$

where $\tau,\gamma\in(0,1/2]$, $\tilde{A}_S=\frac{1}{\sqrt{m}}\mathbf{A}_S$, $\tilde{A}_S^*$ is the Hermitian transposed of $\tilde{A}_S$, and $\tilde{a}_l=\frac{1}{\sqrt{m}}a_l$ for $l=1,\ldots,s$.

From Markov inequality, the second term of R.H.S. of (4.25) is bounded as

$$\mathbb{P}\left[\left\|\tilde{A}_S^*\tilde{A}_S-\mathbb{I}\right\|_2 > \gamma\right] \leq \frac{E}{\gamma} \qquad (4.26)$$

where $E=\mathbb{E}_{\bar{\nu}}\|\tilde{A}_S^*\tilde{A}_S-\mathbb{I}\|_2$, and the expectation is taken w.r.t. the probability measure $\bar{\nu}$. Then we write E as

$$E=\mathbb{E}_{\bar{\nu}}\left\|\frac{1}{m}\sum_{l=1}^{m}X_l^*X_l-\mathbb{I}\right\|_2 \qquad (4.27)$$

where X_l is the l^{th} column of $\tilde{A}_S^*$. Now, let's look at $\mathbb{E}_{\bar{\nu}}\left(X_lX_l^*\right)$. The $(i,j)^{\text{th}}$ entry of $\mathbb{E}_{\bar{\nu}}\left(X_lX_l^*\right)$ is

$$\mathbb{E}_{\bar{\nu}}\left(X_lX_l^*\right)_{ij}=\frac{1}{m}\sum_{l=1}^{m}\mathbb{E}_{\bar{\nu}}\left[\overline{\psi_i(t_l)}\psi_j(t_l)\right]. \qquad (4.28)$$

Then, we have $\mathbb{E}_{\bar{\nu}}\left(X_l X_l^*\right) = \mathbb{I} + \Delta$ or alternatively

$$\mathbb{I} = \mathbb{E}_{\bar{\nu}}\left(X_l X_l^*\right) - \Delta \tag{4.29}$$

where Δ is given in (4.6) and ρ_{ij} is given in (4.7) for $i = 1, \ldots, N, j = 1, \ldots, N,$ and $i \neq j$. Using (4.29), we write

$$\begin{aligned} E &= \mathbb{E}_{\bar{\nu}} \left\| \frac{1}{m} \sum_{l=1}^{m} \left(X_l^* X_l - \mathbb{E}_{\bar{\nu}}\left(X_l X_l^*\right) + \Delta\right) \right\|_2 \\ &\leq \mathbb{E}_{\bar{\nu}} \left\| \frac{1}{m} \sum_{l=1}^{m} \left(X_l^* X_l - \mathbb{E}_{\bar{\nu}}\left(X_l X_l^*\right)\right) \right\|_2 + Q \end{aligned} \tag{4.30}$$

where $Q = \|\Delta\|_2$.

From the symmetrization lemma [21, Lemma 6.7], we have

$$\mathbb{E}_{\bar{\nu}} \left\| \sum_{l=1}^{m} X_l X_l^* - \mathbb{E}_{\bar{\nu}} X_l X_l^* \right\|_2 \leq 2\mathbb{E}_{\bar{\nu}} \left\| \sum_{l=1}^{m} \xi_l X_l X_l^* \right\|_2 \tag{4.31}$$

where $\{\xi_l\}_{l=1}^{m}$ is a Radermacher sequence. Therefore,

$$E < \frac{2}{m} \mathbb{E}_{\bar{\nu}} \left\| \sum_{l=1}^{m} \xi_l X_l X_l^* \right\|_2 + Q \tag{4.32}$$

From Rudelson's lemma [21, Lemma 6.18], we have

$$\mathbb{E}_{\bar{\nu}} \left\| \sum_{l=1}^{m} \xi_l X_l X_l^* \right\|_2 \leq 2^{7/8} \sqrt{\frac{sm}{e}} K \sqrt{\mathbb{E}_{\bar{\nu}} \left\| \tilde{A}_S^* \tilde{A}_S - \mathbb{I} + \mathbb{I} \right\|_2} \tag{4.33}$$

From (4.31)–(4.33) we obatin

$$E \leq 2^{15/8} \sqrt{\frac{s}{me}} K \sqrt{E+1} + Q. \tag{4.34}$$

Let $D = 2^{15/8}\sqrt{s/me}K$, then

$$E \leq D\sqrt{E+1} + Q \leq \kappa D \tag{4.35}$$

where

$$\kappa = \sqrt{\frac{D^2}{4} + 1 + Q} + \frac{Q}{D} + \frac{D}{2}. \tag{4.36}$$

From [21, Proposition 6.5], we have

$$\mathbb{P}\left[\|\tilde{\mathbf{A}}_S^*\tilde{\mathbf{A}}_S - \mathbb{I}\|_2 \geq \gamma\right] \leq 2^{3/4} s \exp\left(-\frac{ms\gamma^2}{8\sqrt[4]{2}K^2\kappa^2}\right). \tag{4.37}$$

This bound is similar in form to the probability bound in [21, Theorem 7.3], except that κ in (4.67) now contains the effect of mismatch as captured by (4.36). From here on, we can follow similar steps as in [21, Section 7.3] to obtain (4.44). We omit these steps for brevity.

4.2.6.2. *Proof of Theorem 4.3.3.* From Markov inequality, we have

$$\mathbb{P}\left[|||\tilde{\mathbf{A}}^*\tilde{\mathbf{A}} - \mathbb{I}|||_s\right] \leq \frac{\tilde{E}}{\gamma_s} \tag{4.38}$$

where $\tilde{E} = |||\tilde{\mathbf{A}}^*\tilde{\mathbf{A}} - \mathbb{I}|||_s$. Following similar steps as (4.27)–(4.30), with $\|\cdot\|_2$ replaced by $|||\cdot|||_s$, we obtain

$$E \leq \frac{1}{\tilde{m}}\mathbb{E}_{\bar{\nu}}|||\sum_{l=1}^{\tilde{m}}\left(X_l X_l^* - \mathbb{E}_{\bar{\nu}} X_l X_l^*\right)|||_s + \tilde{Q} \tag{4.39}$$

where $\tilde{Q} = |||\Delta|||_s$. From this point on the proof follows that of [21, Section 8.5], except that we carry $\tilde{Q}$ in our derivations quantify the effects of mismatch.

4.3. Complete Proof of Generalized Recovery Bounds for Compressive Sensing

Let $\{\psi_k(t)\}_{k=1}^N$ be an orthonormal system of complex functions with respect to measure ν that is

$$\int_{\mathbb{R}} \psi_j(t)\overline{\psi_k(t)}\,\mathrm{d}\nu(t) = \delta_{jk} \quad j,k \in \{1,\dots,N\}, \tag{4.40}$$

where δ_{jk} is the Kronecker delta and measure space $\mathbb{R}$ is endowed with measure ν, and $\nu(\mathbb{R}) = 1$. It should be noted that the results derived here are valid for any measure space $\mathcal{D} \subset \mathbb{R}^d$. Assume basis elements $\psi_k(t), k = 1,\dots,N$ are bounded as

$$\|\psi_k\|_\infty = \sup_{t\in\mathcal{D}} |\psi_k(t)| \le K,\ K \in \mathbb{R}^+. \tag{4.41}$$

Let $y(t)$ be a generic signal that has an s-sparse representation in $\{\psi_k(t)\}_{k=1}^N$. That is,

$$y(t) = \mathbf{\Psi}(t)\mathbf{b} = \sum_{k=1}^N \psi_k(t) b_k, \tag{4.42}$$

where $\mathbf{\Psi}(t) = [\psi_1(t),\dots,\psi_N(t)]$, and $\mathbf{b} = [b_1,\dots,b_N]^T \in \mathbb{R}^N$ is s-sparse. Let $t_1,\dots,t_m$ be a sequence of i.i.d. realizations w.r.t. a probability measure $\bar{\nu}$. Let $\mathbf{y} = [y(t_1),\dots,y(t_m)]^T$ and $\mathbf{A}$ be an $m\times N$ matrix whose $(l,k)^{\text{th}}$ entry $\mathbf{A}_{l,k} = \psi_k(t_l)$. Then,

$$\mathbf{y} = \mathbf{A}\mathbf{b}. \tag{4.43}$$

We are interested in deriving bounds on the number of samples m required to guarantee the recovery of $y(t)$ from a random set of samples, with high probability.

In [21], the authors have investigated this question in the case where the sampling measure $\bar{\nu}$ is identical to the orthogonality measure ν. They have studied two scenarios: *non-uniform* *recovery* and *uniform recovery.* Given each fixed s-sparse support, non-uniform recovery guarantees the recovery of the signal from a random realization of sample points with high probability. Given a fixed realization of sample points, uniform recovery guarantees the recovery of any s-sparse signal with high probability. In [64], the authors present improved guarantee bounds for the uniform recovery case considered in [21].

In this dissertation, we study the case where there is a mismatch between the sampling measure $\bar{\nu}$ and the orthogonality measure ν, and derive sufficient conditions for uniform and non-uniform recoveries.

4.3.1. Non-Uniform Recovery. Here we study the recovery of signals with a fixed support.

Theorem 4.3.1. *Let $\mathbf{b} \in \mathbb{R}^N$ be an arbitrary s-sparse signal with fixed but unknown support S. Let $\{t_i\}_{i=1}^m$ be a sequence of i.i.d. draws w.r.t. the probability measure $\bar{\nu}$ and $\mathbf{A} \in \mathbb{R}^{m\times N}$ be a matrix whose $(l,k)^{th}$ entry $\mathbf{A}_{l,k} = \psi_k(t_l)$. Then, $\mathbf{b}$ can be successfully recovered from $\mathbf{y} = \mathbf{A}\mathbf{b}$ with probability at least $(1-\epsilon)$ if*

$$m \geq 8\sqrt[4]{2}sK^2\kappa^2 \log\left(\frac{3(N-s)2^{3/4}}{\epsilon}\right)\log\left(\frac{3\left(N^2+s\right)}{\sqrt[4]{2}\epsilon}\right). \tag{4.44}$$

Proof. According to [58, 59] and [21, Corollary 2.8], if $\mathbf{b}$ with fixed support $S(|S| = s)$ satisfies

$$\left|\left\langle \mathbf{A}_S^{\dagger}a_l, sgn\left(\mathbf{b}_S\right)\right\rangle\right| < 1 \;\forall l \notin S \tag{4.45}$$

then $\mathbf{b}$ can be recovered from $\mathbf{y} = \mathbf{A}\mathbf{b}$ using a linear program. Here $\mathbf{A}_S$ denotes the submatrix formed with the columns of $\mathbf{A}$ on S, $\mathbf{b}_S$ denotes the sub-vector formed with the elements of $\mathbf{b}$ on S, a_l is the l^{th} column of $\mathbf{A}$, $sgn(\cdot)$ is the signum function, and $\mathbf{A}_S^{\dagger}$ is the Moore-Penrose inverse of $\mathbf{A}_S$. Then the probability ϵ of failure to recover $\mathbf{b}$ is bounded as

$$\epsilon \leq \mathbb{P}\left[\max_{l \notin S}\left|\left\langle \mathbf{A}_S^{\dagger} a_l, sgn\left(\mathbf{b}_S\right)\right\rangle\right| \geq 1\right]. \tag{4.46}$$

Next we consider two cases:

(1) $\left\|\mathbf{A}_S^{\dagger} a_l\right\|_2 \leq \alpha \; \forall l \notin S$

(2) $\left\|\mathbf{A}_S^{\dagger} a_l\right\|_2 > \alpha \; \forall l \notin S.$

We will set the value of α later on.

Under the first case

$$\mathbb{P}\left[\max_{l \notin S}\left|\left\langle \mathbf{A}_S^{\dagger} a_l, sgn\left(\mathbf{b}_S\right)\right\rangle\right| \geq 1\right] \leq \mathbb{P}\left[\max_{l \notin S}\left|\left\langle \mathbf{A}_S^{\dagger} a_l, sgn\left(\mathbf{b}_S\right)\right\rangle\right| \geq \left\|\mathbf{A}_S^{\dagger} a_l\right\|_2 \alpha^{-1}\right] \tag{4.47}$$

Then by applying union bound

$$\mathbb{P}\left[\max_{l \notin S}\left|\left\langle \mathbf{A}_S^{\dagger} a_l, sgn\left(\mathbf{b}_S\right)\right\rangle\right| \geq 1\right] \leq \sum_{l \notin S} \mathbb{P}\left[\left|\left\langle \mathbf{A}_S^{\dagger} a_l, sgn\left(\mathbf{b}_S\right)\right\rangle\right| \geq \left\|\mathbf{A}_S^{\dagger} a_l\right\|_2 \alpha^{-1}\right] \tag{4.48}$$

Reference [21, Corollary 6.10] states

$$\mathbb{P}\left[\left|\left\langle \mathbf{A}_S^{\dagger} a_l, sgn\left(\mathbf{b}_S\right)\right\rangle\right| \geq \left\|\mathbf{A}_S^{\dagger} a_l\right\|_2 \alpha^{-1}\right] \leq 2^{3/4} \exp\left(-\alpha^{-2}/2\right) \tag{4.49}$$

Therefore

$$\mathbb{P}\left[\max_{l\notin S}\left|\left\langle \mathbf{A}_S^{\dagger}a_l, sgn\left(\mathbf{b}_S\right)\right\rangle\right| \geq 1\right] \leq (N-s)2^{3/4}\exp\left(-\alpha^{-2}/2\right) \quad (4.50)$$

Using the above two cases, (4.46) can be re-written as

$$\begin{aligned}\mathbb{P}\left[\max_{l\notin S}\left|\left\langle \mathbf{A}_S^{\dagger}a_l, sgn\left(\mathbf{b}_S\right)\right\rangle\right| \geq 1\right] = &\\ &\mathbb{P}\left[\left\|\mathbf{A}_S^{\dagger}a_l\right\|_2 \leq \alpha\right]\\ &\mathbb{P}\left[\max_{l\notin S}\left|\left\langle \mathbf{A}_S^{\dagger}a_l, sgn\left(\mathbf{b}_S\right)\right\rangle\right| \geq 1 \;\middle|\; \left\|\mathbf{A}_S^{\dagger}a_l\right\|_2 \leq \alpha\right] +\\ &\mathbb{P}\left[\left\|\mathbf{A}_S^{\dagger}a_l\right\|_2 > \alpha\right]\\ &\mathbb{P}\left[\max_{l\notin S}\left|\left\langle \mathbf{A}_S^{\dagger}a_l, sgn\left(\mathbf{b}_S\right)\right\rangle\right| \geq 1 \;\middle|\; \left\|\mathbf{A}_S^{\dagger}a_l\right\|_2 > \alpha\right]. \quad (4.51)\end{aligned}$$

Then can be bounded as

$$\begin{aligned}\mathbb{P}\left[\max_{l\notin S}\left|\left\langle \mathbf{A}_S^{\dagger}a_l, sgn\left(\mathbf{b}_S\right)\right\rangle\right| \geq 1\right] &\\ &\leq (N-s)2^{3/4}\exp\left(-\alpha^{-2}/2\right) +\\ &\mathbb{P}\left[\max_{l\notin S}\left\|\mathbf{A}_S^{\dagger}a_l\right\|_2 > \alpha\right]. \quad (4.52)\end{aligned}$$

Now we focus on the second term of the R.H.S. of (4.52). By the definition of Moore-Penrose inverse

$$\mathbb{P}\left[\max_{l\notin S}\left\|\mathbf{A}_S^{\dagger}a_l\right\|_2>\alpha\right]=\mathbb{P}\left[\max_{l\notin S}\left\|\left(\mathbf{A}_S^*\mathbf{A}_S\right)^{-1}\mathbf{A}_S^*a_l\right\|_2>\alpha\right]. \quad (4.53)$$

Following [21] we set $\alpha=\frac{\sqrt{s}t}{1-\gamma}$ for some $t,\gamma\in(0,1/2]$

$$\begin{aligned}\mathbb{P}\left[\max_{l\notin S}\left\|\mathbf{A}_S^{\dagger}a_l\right\|_2>\alpha\right]&\leq\\ &\mathbb{P}\left[\max_{l\notin S}\left\|\left(\mathbf{A}_S^*\mathbf{A}_S\right)^{-1}\right\|_2\left\|\mathbf{A}_S^*a_l\right\|_2>\frac{\sqrt{s}t}{1-\gamma}\right]\\ &\leq\mathbb{P}\left[\left\|\left(\mathbf{A}_S^*\mathbf{A}_S\right)^{-1}\right\|_2>\frac{1}{1-\gamma}\right]+\\ &\qquad\mathbb{P}\left[\max_{l\notin S}\left\|\mathbf{A}_S^*a_l\right\|_2>\sqrt{s}t\right]. \quad (4.54)\end{aligned}$$

By applying the normalization $\tilde{A} = \frac{1}{\sqrt{m}}\mathbf{A}$, Neumann series, and using the definition of spectral norm $\|B\|_2 = \max_{\|z\|_2=1} \|Bz\|_2 = \sup_{\|z\|_2=1} | < Bz, z > |$

$$\begin{aligned}
&\mathbb{P}\left[\max_{l\notin S}\left\|\mathbf{A}_S^{\dagger}a_l\right\|_2 > \alpha\right] \leq \\
&\qquad \leq \mathbb{P}\left[\sum_{k=1}^{\infty}\left\|\tilde{A}_S^{*}\tilde{A}_S - \mathbb{I}\right\|_2^{k} > \sum_{k=1}^{\infty}\gamma^{k}\right] + \\
&\qquad \mathbb{P}\left[\max_{l\notin S}\sqrt{\sum_{j\in S}\left|\langle a_l, a_j\rangle\right|^2} > \sqrt{s}t\right] \\
&\qquad \leq \mathbb{P}\left[\left\|\tilde{A}_S^{*}\tilde{A}_S - \mathbb{I}\right\|_2 > \gamma\right] + \\
&\qquad \mathbb{P}\left[\max_{l\notin S}\sqrt{\sum_{j\in S}\left|\langle \tilde{a}_l, \tilde{a}_j\rangle\right|^2} > \sqrt{s}t\right]. \qquad (4.55)
\end{aligned}$$

Next we focus on the first term on the R.H.S. of (4.55). From Markov inequality,

$$\mathbb{P}\left[\|\tilde{\mathbf{A}}_S^{*}\tilde{\mathbf{A}}_S - \mathbb{I}\|_2 \geq \gamma\right] \leq \frac{E}{\gamma} \qquad (4.56)$$

where $E = \mathbb{E}\|\tilde{\mathbf{A}}_S^{*}\tilde{\mathbf{A}}_S - \mathbb{I}\|_2$, and the expectation is taken w.r.t. probability measure ν. Let $X_l = \bar{\Psi}_S(t_l) \in \mathbb{C}^s$ be the l^{th} column of $\tilde{\mathbf{A}}_S^{*}$. Next we take the expectation of $X_l X_l^{*}$ w.r.t. $\bar{\nu}$ - the sampling measure.

$$\mathbb{E}_{\bar{\nu}} X_l X_l^{*} = \frac{1}{m}\sum_{l=1}^{m} X_l X_l^{*} = \frac{1}{m}\sum_{l=1}^{m}\bar{\Psi}_S(t_l)\Psi_S(t_l). \qquad (4.57)$$

The $(i,j)^{\text{th}}$ entry of $\mathbb{E}_{\bar{\nu}} X_l X_l^*$ can be written as

$$
\begin{aligned}
(\mathbb{E}_{\bar{\nu}} X_l X_l^*)_{ij} &= \frac{1}{m} \sum_{l=1}^{m} \overline{\psi_i(t_l)} \psi_j(t_l) \\
&= \int_{\mathbb{R}} \psi_j(t) \overline{\psi_i(t)} \, \mathrm{d}\bar{\nu}(t) = \rho_{ij} \qquad (4.58) \\
\therefore \mathbb{E}_{\bar{\nu}} X_l X_l^* &= \mathbb{I} + \Delta \qquad (4.59)
\end{aligned}
$$

where Δ is a matrix whose diagonal entries are zero, and whose $(i,j)^{\text{th}}$ entry (for $i \neq j$) is ρ_{ij}.

Now we can write E as

$$
\begin{aligned}
E &= \mathbb{E} \| \frac{1}{m} \sum_{l=1}^{m} \left(X_l^* X_l - \mathbb{E}_{\bar{\nu}} X_l X_l^* + \Delta \right) \|_2 \\
&\leq \mathbb{E} \| \frac{1}{m} \sum_{l=1}^{m} \left(X_l^* X_l - \mathbb{E}_{\bar{\nu}} X_l X_l^* \right) \|_2 + \|\Delta\|_2. \qquad (4.60)
\end{aligned}
$$

The symmetrization lemma [21, Lemma 6.7] states

$$
\left(\mathbb{E} \left\| \sum_{l=1}^{m} X_l X_l^* - \mathbb{E}_{\bar{\nu}} X_l X_l^* \right\|_2^p \right)^{1/p} \leq
2 \left(\mathbb{E} \left\| \sum_{l=1}^{m} \xi_l X_l X_l^* \right\|_2^p \right)^{1/p} \quad \text{for } 1 \leq p < \infty \qquad (4.61)
$$

where ξ_l is a Radermacher sequence.

$$
\therefore E < \frac{2}{m} \mathbb{E} \left\| \sum_{l=1}^{m} \xi_l X_l X_l^* \right\|_2 + Q \qquad (4.62)
$$

where $Q = \|\Delta\|_2$.

Rudelson's lemma [21, Lemma 6.18] states

$$\left(\mathbb{E}\left\|\sum_{l=1}^{m}\xi_l X_l X_l^*\right\|_2^p\right)^{1/p} \leq$$

$$2^{3/4p}s^{1/p}\sqrt{p}e^{-1/2}\mathbb{E}\left[\|A_S^*\|_2 \max_l \|X_l\|_2\right] \text{ for } 2 \leq p < \infty. \quad (4.63)$$

Assuming $\mathbb{E}\left\|\sum_{l=1}^{m}\xi_l X_l X_l^*\right\|_2 \geq 1$ and $\mathbb{E}\|A_S^*\|_2 \geq 1$

$$\begin{aligned}\mathbb{E}\left\|\sum_{l=1}^{m}\xi_l X_l X_l^*\right\|_2 &\leq \left(\mathbb{E}\left\|\sum_{l=1}^{m}\xi_l X_l X_l^*\right\|_2^2\right)^{1/2}\\ &\leq 2^{7/8}\sqrt{\frac{s}{e}}K\sqrt{\mathbb{E}\|A_S^*A_S\|_2}\\ &= 2^{7/8}\sqrt{\frac{sm}{e}}K\sqrt{\mathbb{E}\left\|\tilde{A}_S^*\tilde{A}_S - \mathbb{I} + \mathbb{I}\right\|_2} \quad (4.64)\\ \therefore E &\leq 2^{15/8}\sqrt{\frac{s}{me}}K\sqrt{E+1} + Q. \quad (4.65)\end{aligned}$$

Let $D = 2^{15/8}\sqrt{s/me}K$, then

$$\begin{aligned}E &\leq D\sqrt{E+1} + Q\\ &\leq \kappa D \quad (4.66)\end{aligned}$$

where $\kappa = \sqrt{\frac{D^2}{4} + 1 + Q} + \frac{Q}{D} + \frac{D}{2}$.

Reference [21, Proposition 6.5] states if $(\mathbb{E}|Z|^p)^{1/p} < \theta\beta^{1/p}p^{1/\eta}$ for some $\theta, \beta, \eta, p > 0$ then $\mathbb{P}\left[|Z| > e^{1/\eta}\theta u\right] < \beta e^{-u^\eta/\eta}$. By comparing variables of (4.56) and (4.66) we set $\beta = 2^{3/4}s$ and $u = \frac{8\sqrt{ms}}{2\sqrt[8]{2}K\kappa}$. Then

$$\mathbb{P}\left[\|\tilde{\mathbf{A}}_S^*\tilde{\mathbf{A}}_S - \mathbb{I}\|_2 \geq \gamma\right] \leq 2^{3/4}s\exp\left(-\frac{ms\gamma^2}{8\sqrt[4]{2}K^2\kappa^2}\right). \quad (4.67)$$

Next we discuss the second term on the R.H.S. of (4.55). Let's pick a pair of column indices i, j and define $T = \{i, j\}, |T| = 2$, set $\gamma = t$, and apply in (4.67).

$$\mathbb{P}\left[\|\tilde{\mathbf{A}}_T^*\tilde{\mathbf{A}}_T - \mathbb{I}\|_2 \geq t\right] \leq 2^{3/4} \cdot 2 \cdot \exp\left(-\frac{2mt^2}{8\sqrt[4]{2}K^2\kappa^2}\right). \tag{4.68}$$

There are $\binom{N}{2} = \frac{N(N-1)}{2}$ $(< \frac{N^2}{2})$ candidate sets for T. Using union bound, for any T

$$\begin{aligned}\mathbb{P}\left[\|\tilde{\mathbf{A}}_T^*\tilde{\mathbf{A}}_T - \mathbb{I}\|_2 \geq t\right] &\leq 2^{3/4} \cdot 2 \cdot \exp\left(-\frac{2mt^2}{8\sqrt[4]{2}K^2\kappa^2}\right) \cdot \frac{N^2}{2} \\ &= 2^{3/4} N^2 \exp\left(-\frac{mt^2}{4\sqrt[4]{2}K^2\kappa^2}\right)\end{aligned} \tag{4.69}$$

The second term on the R.H.S. of (4.55) can be bounded as

$$\mathbb{P}\left[\max_{l \notin S} \sqrt{\sum_{j \in S} |\langle \tilde{a}_l, \tilde{a}_j \rangle|^2} > \sqrt{s}t\right] \leq \mathbb{P}\left[\max_{i \neq j} |\langle \tilde{a}_i, \tilde{a}_j \rangle| > t\right] \tag{4.70}$$

Since the absolute value of any entry of a matrix is bounded by the spectral norm of the matrix $|\langle \tilde{a}_i, \tilde{a}_j \rangle| \leq \left\|\tilde{\mathbf{A}}_T^*\tilde{\mathbf{A}}_T - \mathbb{I}\right\|_2$.

$$\begin{aligned}\therefore \mathbb{P}&\left[\max_{l \notin S} \sqrt{\sum_{j \in S} |\langle \tilde{a}_l, \tilde{a}_j \rangle|^2} > \sqrt{s}t\right] \\ &\leq \mathbb{P}\left[\left\|\tilde{\mathbf{A}}_T^*\tilde{\mathbf{A}}_T - \mathbb{I}\right\|_2 \geq t\right] \\ &\leq 2^{3/4} N^2 \exp\left(-\frac{mt^2}{4\sqrt[4]{2}K^2\kappa^2}\right)\end{aligned} \tag{4.71}$$

Now we proceed to derive the non-uniform recovery bound. For this, we plug (4.67) and (4.71) in (4.55), then apply (4.55) in (4.52), and finally apply (4.52) in (4.46).

$$\epsilon \leq \mathbb{P}\left[\max_{l \notin S}\left|\left\langle \mathbf{A}_S^{\dagger} a_l, sgn\left(\mathbf{b}_S\right)\right\rangle\right| \geq 1\right]$$

$$\leq (N-s)2^{3/4}\exp\left(-\alpha^{-2}/2\right) +$$

$$2^{3/4}s\exp\left(-\frac{ms\gamma^2}{8\sqrt[4]{2}K^2\kappa^2}\right) +$$

$$2^{3/4}N^2\exp\left(-\frac{mt^2}{4\sqrt[4]{2}K^2\kappa^2}\right) \quad (4.72)$$

Following [21] we bound the first term by $\epsilon/3$ and set $t = \gamma\sqrt{s/2}$. From the first term we deduce

$$\frac{1}{\gamma^2} \geq s^2 \log\left(\frac{3(N-s)2^{3/4}}{\epsilon}\right). \quad (4.73)$$

From the latter two terms we deduce

$$m \leq \frac{1}{\gamma^2}\frac{8\sqrt[4]{2}K^2\kappa^2}{s}\log\left(\frac{3\left(N^2+s\right)}{\sqrt[4]{2}\epsilon}\right). \quad (4.74)$$

Therefore if

$$m \leq 8\sqrt[4]{2}sK^2\kappa^2 \log\left(\frac{3(N-s)2^{3/4}}{\epsilon}\right)\log\left(\frac{3\left(N^2+s\right)}{\sqrt[4]{2}\epsilon}\right) \quad (4.75)$$

signal $\mathbf{b}$ cannot be recovered from $\mathbf{y} = \mathbf{A}\mathbf{b}$ with probability $(1-\epsilon)$. $\square$

REMARK: Note that Q is the *noise* generated due to violating the orthogonalization measure condition (4.40). Q vanishes in the cases similar to [21] where the sampling is an orthogonalization measure.

4.3.1.1. *Example: Exponential and Gaussian.* We assume that $y(t)$ has a sparse representation in *Fourier* domain, i.e., $\psi_k(t) = e^{\imath\omega_k t}$ and $\|\mathbf{b}\|_0 \le s$. We also assume $\psi_k(t)$'s have the same fundamental frequency $\omega_k = k\omega_0$.

$$\text{Note that} \quad \Delta_{jk} = \begin{cases} 0 & \text{for } j = k \\ \rho_{jk} & \text{for } j \neq k \end{cases} \quad \text{and } Q = \mathbb{E}\|\Delta\|_2^2.$$

If $\nu(t)$ is exponential with rate λ, from (4.58) for $j \neq k$

$$\rho_{jk} = \int_0^{+\infty} e^{\imath(\omega_j-\omega_k)t} \cdot \lambda e^{-\lambda t}\, \mathrm{d}t.$$

$$\therefore |\rho_{jk}| = \frac{\lambda}{\sqrt{\lambda^2 + (\omega_j - \omega_k)^2}}. \tag{4.76}$$

Using *Gershgorin circle theorem*

$$\|\Delta\|_2 = \sigma\text{max}(\Delta) \le \max_j \sum_{k\neq j} |\rho_{jk}|\,. \tag{4.77}$$

$$Q \le 4\lambda^2 \check{N}^2 \left(\sum_{k=1}^{\check{N}} \frac{\lambda}{\sqrt{\lambda^2 + \omega_0^2 k^2}} \right)^2, \; \check{N} = \lceil \frac{N}{2} - 1 \rceil. \tag{4.78}$$

For Gaussian $\nu(t)$ with mean $\check{\mu}$ and standard deviation $\check{\sigma}$,

$$\rho_{jk} = \int_{-\infty}^{+\infty} e^{\imath(\omega_j-\omega_k)t} \cdot \frac{1}{\check{\sigma}\sqrt{2\pi}} e^{-\frac{(t-\check{\mu})^2}{2\check{\sigma}^2}}\, \mathrm{d}t. \tag{4.79}$$

$$Q \le 4 \left(\sum_{k=1}^{\check{N}} \exp\left(-\frac{1}{2}\omega_0^2\sigma^2 k^2\right) \right)^2, \; \check{N} = \lceil \frac{N}{2} - 1 \rceil. \tag{4.80}$$

4.3.2. Uniform Recovery. *Restricted Isometric Property* (*RIP*) of $\mathbf{A}$ is a commonly used sufficient condition for $\mathbf{A}$ recovering of $\mathbf{b}$ from $\mathbf{y}$ (see, e.g. [46]). The RIP means that

A satisfies

$$(1-\gamma_s)\|\mathbf{b}\|_2^2 \leq \|\mathbf{Ab}\|_2^2 \leq (1+\gamma_s)\|\mathbf{b}\|_2^2,\ \forall \mathbf{b} \in \Gamma_s \tag{4.81}$$

where Γ_s is the set of all s-sparse vectors in $\mathbb{R}^N$ and γ_s is a positive constant called *Restricted Isometric Constant* (*RIC*). When **A** satisfies (4.81) we say **A** is γ_s-RIP. If A has γ_{2s}-RIP with $\gamma_{2s} < 1/3$, then every s-sparse **b** is recoverable from $\mathbf{y} = \mathbf{Ab}$ using a linear program [46]. If RIP is satisfied only probabilistically, then recovery is guaranteed only probabilistically as well.

The following operator semi norm $|||\cdot|||_s$ is defined for s-sparse unit vectors.

$$|||\mathbf{B}|||_s = \sup_{\|\mathbf{z}\|_0 \leq s, \|\mathbf{z}\|_2 = 1} |\langle \mathbf{Bz}, \mathbf{z}\rangle| \tag{4.82}$$

Then RIP can be rearranged as

$$|||\tilde{\mathbf{A}}^*\tilde{\mathbf{A}} - \mathbb{I}|||_s \leq \gamma_s \tag{4.83}$$

where $\tilde{A} = \frac{1}{\sqrt{m}}\mathbf{A}$, as before. Then the probability ϵ with which γ_s-RIP is not satisfied is give by

$$\epsilon = \mathbb{P}\left[|||\tilde{\mathbf{A}}^*\tilde{\mathbf{A}} - \mathbb{I}|||_s \geq \gamma_s\right]. \tag{4.84}$$

4.3.2.1. *Recovery Probability Guarantee.*

THEOREM 4.3.2. *Let* $\mathbf{b} \in \mathbb{R}^N$ *be an arbitrary s-sparse signal. Let* $\{t_i\}_{i=1}^{\tilde{m}}$ *be a sequence of i.i.d. draws w.r.t. the probability measure* $\bar{\nu}$ *and* $\mathbf{A} \in \mathbb{R}^{\tilde{m}\times N}$ *be a matrix whose* $(l,k)^{th}$ *entry* $\mathbf{A}_{l,k} = \psi_k(t_l)$*. Then,* **b** *can be successfully recovered from* $\mathbf{y} = \mathbf{Ab}$ *with probability at least* $(1-\epsilon)$ *if*

$$\epsilon \leq 7\exp\left(-\frac{\gamma_s^2}{c_1\tilde{\kappa}^2 s \ln^2(100s)\ln(4N)}\frac{\tilde{m}}{\ln(10\tilde{m})}\right). \tag{4.85}$$

Proof. By noting

$$\tilde{\mathbf{A}}^*\tilde{\mathbf{A}} = \frac{1}{\tilde{m}}\sum_{l=1}^{\tilde{m}} X_l X_l^* \quad \text{and} \quad \mathbb{E}_{\bar{\nu}} X_l X_l^* = \mathbb{I} + \Delta$$

we redefine the quantity E as

$$E = \mathbb{E}|||\tilde{\mathbf{A}}^*\tilde{\mathbf{A}} - \mathbb{I}|||_s. \tag{4.86}$$

restate (4.86) as

$$E = \frac{1}{\tilde{m}}\mathbb{E}|||\sum_{l=1}^{\tilde{m}} \left(X_l X_l^* - \mathbb{E}_{\bar{\nu}} X_l X_l^* + \Delta\right)|||_s. \tag{4.87}$$

Then use *Jensen's inequality* to separate Δ as

$$E \leq \frac{1}{\tilde{m}}\mathbb{E}|||\sum_{l=1}^{\tilde{m}} \left(X_l X_l^* - \mathbb{E}_{\bar{\nu}} X_l X_l^*\right)|||_s + \mathbb{E}|||\Delta|||_s. \tag{4.88}$$

By applying the *Symmetrization Lemma* [21, Lemma 6.7]

$$E \leq \frac{2}{\tilde{m}}\mathbb{E}|||\sum_{l=1}^{\tilde{m}} \varepsilon_l X_l X_l^*|||_s + \mathbb{E}|||\Delta|||_s \tag{4.89}$$

where $\{\varepsilon_1, \ldots, \varepsilon_{\tilde{m}}\}$ is a *Rademarcher sequence.* Using the *Crucial Lemma* [21, Lemma 8.2] on (4.89) and setting $\tilde{Q} = \mathbb{E}|||\Delta|||_s$

$$E \leq \frac{2}{\tilde{m}} D|||\sum_{l=1}^{\tilde{m}} X_l X_l^*|||_s^{1/2} + \tilde{Q} \tag{4.90}$$

where $D = C\sqrt{2\beta}K\sqrt{s}\ln(100s)\sqrt{\ln(4N)\ln(10\tilde{m})}$ with $C \approx 67.97$ and $\beta = 6.028$ from *Dudley's inequality* ([21, Theorem 6.42]). Following [21] we use *triangle inequality* to derive

$$E \leq \frac{2D}{\sqrt{\tilde{m}}}\sqrt{E+1} + \tilde{Q}. \tag{4.91}$$

Then by *completing the squares* yields

$$E \leq \frac{2D}{\sqrt{\tilde{m}}}\left(\sqrt{1+\frac{D^2}{\tilde{m}}+\tilde{Q}}+\frac{\tilde{Q}\sqrt{\tilde{m}}}{2D}+\frac{D}{\tilde{m}}\right). \tag{4.92}$$

$$\text{Let}\quad \tilde{\kappa} = \sqrt{1+\frac{D^2}{\tilde{m}}+\tilde{Q}}+\frac{\tilde{Q}\sqrt{\tilde{m}}}{2D}+\frac{D}{\tilde{m}}. \tag{4.93}$$

Then following the same procedure as for (4.67) we obtain

$$\begin{aligned}\epsilon &= \mathbb{P}\left[|||\tilde{\mathbf{A}}^*\tilde{\mathbf{A}}-\mathbb{I}|||_s \geq \gamma_s\right]\\ &\leq 7\exp\left(-\frac{\gamma_s^2}{c_1\tilde{\kappa}^2 s\ln^2(100s)\ln(4N)}\frac{\tilde{m}}{\ln(10\tilde{m})}\right)\end{aligned} \tag{4.94}$$

where $c_1 = 8eC$. □

4.3.2.2. *Minimum Number of Samples Required.*

THEOREM 4.3.3. *Let $\mathbf{b} \in \mathbb{R}^N$ be an arbitrary s-sparse signal. Let $\{t_i\}_{i=1}^{\tilde{m}}$ be a sequence of i.i.d. draws w.r.t. the probability measure $\bar{\nu}$ and $\mathbf{A} \in \mathbb{R}^{\tilde{m}\times N}$ be a matrix whose $(l,k)^{th}$ entry $\mathbf{A}_{l,k} = \psi_k(t_l)$. Then, to successfully recover $\mathbf{b}$ from $\mathbf{y} = \mathbf{A}\mathbf{b}$ with a probability at least $(1-\epsilon)$ will require at least $\tilde{m}$ samples satisfying*

$$\frac{\tilde{m}}{\ln(10\tilde{m})} \geq \frac{c_1\tilde{\kappa}^2}{\gamma_s^2} s\ln^2(100s)\ln(4N)\ln\left(\frac{7}{\epsilon}\right) \tag{4.95}$$

and when $\mathbb{E}|||\sum_{l=1}^{\tilde{m}} X_l X_l^* - \mathbb{I}|||_s \leq 8\gamma_s/9$

$$\tilde{m} \geq \frac{c_2 D_1}{\gamma_s^2} sK^2 \ln\left(\frac{1}{\epsilon}\right) \tag{4.96}$$

PROOF. Re-arrange Theorem 4.3.2 to obtain (4.95).

When $\mathbb{E}|||\sum_{l=1}^{\tilde{m}} X_l X_l^* - \mathbb{I}|||_s \leq 8\gamma_s/9$, as per [21, Theorem 8.4], ϵ can be strengthened as

$$\epsilon = \mathbb{P}\left[|||\sum_{l=1}^{\tilde{m}} X_l X_l^* - \mathbb{I}|||_s \geq \mathbb{E}|||\sum_{l=1}^{\tilde{m}} X_l X_l^* - \mathbb{I}|||_s + \tilde{Q}\tilde{m} + \frac{\gamma_s \tilde{m}}{9}\right] \quad (4.97)$$

Since $\gamma_s/sK^2 < 1$, using [21, Theorem 6.25]

$$\begin{aligned} \epsilon &\leq \exp\left(-\frac{\left(\tilde{Q} + \frac{\gamma_s}{9}\right)^2 \tilde{m}^2}{2\tilde{m}sK^2 + 4\left(\frac{8\gamma_s}{9}\right)\tilde{m} + \frac{2}{3}\left(\tilde{Q} + \frac{\gamma_s}{9}\right)\tilde{m}}\right) \\ &= \exp\left(-\frac{\tilde{m}\gamma_s}{c_2 D_1 sK^2}\right) \end{aligned} \quad (4.98)$$

where $c_2 \leq 456$ and $D_1 = \frac{1+\tilde{Q}/76}{1+9\tilde{Q}/\gamma_s^2}$.

Rearrange (4.98) into (4.96). □

4.4. Phenomena Discovery in WSNs: A Compressive Sensing Based Approach

A Compressive Sensing (CS) based solution is proposed for centralized and distributed discovery of physical phenomena in large scale Wireless Sensor Networks (WSNs). WSNs monitoring environmental phenomena over large geographic areas collect measurements from a large number of distributed sensors. Compressive Sensing provides an effective means of discovery and reconstruction of functions with only a subset of samples. Traditional CS relies on uniformly distributed samples which limits practicality of CS based recovery. To enhance the flexibility of sampling and implementation, the proposed approach uses random walk based samples. Unlike uniform sampling, random walk based sampling enables individual nodes achieve phenomenon awareness, i.e., the physical distribution of the phenomenon. We also derive a theoretical upper bound for the reconstruction failure probability. Simulation results on the number of samples required and error show that random walk based sampling is comparable to uniform sampling but with superior energy efficiency. More importantly, the proposed scheme provides a practical solution for a range of applications where uniform sampling is less economical or even infeasible.

4.4.1. Introduction. Future Wireless Sensor Networks (WSNs) can be envisioned as large information ecosystems of millions of sensors embedded in the environment. Apart from the complexities posed by the enormous scale, factors such as lack of direct connectivity [66], coverage and delay intolerance [67], make data dissemination and fusion much challenging. Therefore, schemes for data dissemination capable of handling vast amount of data, which are also resilient to intermittent connections and lack of connectivity are in demand.

Compressive Sensing (CS) [11] is an attractive approach to estimate functions from a minimal set of samples. Employing a domain where a signal is represented with a minimal support, CS can recover a high dimensional signal with a small number of random projections or samples of the signal. The distribution of the samples has a major effect on the recovery [21]. Among the distributions identified as feasible, uniformly at random sampling is predominant in literature. However, gathering uniformly scattered samples is expensive in practice. The goal of the presented work is to investigate function recovery of natural physical phenomena using practical sampling schemes and domains providing sparse representations.

Phenomenon, in this dissertation, refers to a distribution or some other profile, e.g., a chemical plume, being monitored by a sensor network. Current CS based phenomena discovery approaches are implemented at a BS instead of at individual nodes [68], as they employ uniform sampling. Since collecting a set of uniformly scattered samples is costly to realize, data are only gathered at BSs, not at individual nodes. We define phenomena awareness as nodes being conscious of the phenomena the network is observing and call the perception process as phenomena discovery. Phenomena awareness ranges from a gross estimation to the exact recovery of the phenomena. This awareness at individual nodes can dramatically improve the capability of the network to efficiently track and react to the changes of the phenomenon. In this dissertation, we demonstrate achieving phenomenon awareness at individual nodes efficiently in a distributed manner. In the presence of a BS, phenomena awareness may be achieved at BS as well. Phenomenon awareness at the node level facilitates smarter and adaptive sensing strategies and provides localized decision making ability to sensor-actuator applications, with no involvement of a base-station. An upper bound is provided for the probability of recovery failure of CS based recovery under a given basis and a sampling scheme. This upper bound provides an estimate for the number of samples required to reconstruct a function within a desired error margin.

We present several motivating examples next. The hydrologic study by USDA-ARS Great Plains Systems Research (Fort Collins, Colorado) has 110ha of a winter wheat and fallow strip cropping system [69], where soil measurements are collected using sensors mounted on a pickup truck. A traditional CS realization needs samples scattered uniformly over the field, which is difficult to achieve with a truck. However, the truck could make random walks (RWs) to collect samples with much ease. We are interested in knowing whether CS reconstruction is possible with RW samples instead of uniform samples. Another application is Intel's Wireless Vineyard [70] which uses ubiquitous computing for agricultural monitoring. Here, the network is expected to not only collect and interpret data, but also to use such data to make decisions related to detecting parasites and using appropriate insecticides. In this delay tolerant network application [71], data collection relies on data mules [72] - small devices carried by people/dogs/robots that communicate with the nodes and collect data.

Here as well the collected samples may not be uniformly scattered. The results presented in this dissertation are applicable under such scenarios.

Distributed phenomena discovery has many emerging applications. If the sensor nodes in Intel's Wireless Vineyard are phenomena aware, then the sensed information can be accessed via any node using a mobile phone, and alerts can be sent out via automated emails or text messages. Vehicular Ad-hoc Networks (VANETs) - networking vehicles with one another to build an infrastructure that provide drivers information beyond their field of vision and warn them about accidents or traffic jams [73] is another example, where it is necessary for individual nodes to be aware of phenomena being monitored.

The rest of the section is organized as follows: Section 4.4.2 presents related work. Proposed novel algorithm for phenomena discovery using random walk and a mathematical bound for the reconstruction performance is discussed in Section 4.4.3. Section 4.4.4 delivers performance evaluation. Finally, Section 4.4.5 concludes the section.

4.4.2. Related Work. Single dimension function recovery in underwater sensor networks is discussed in [74], where function is assumed to be sparse in Fourier domain and sensors send their information directly to the base station in a uniformly at random manner. Discovery of binary sparse events using Bayesian detection is addressed in [75]. However, the performance of their scheme decays as the signal to noise ratio (SNR) approaches 20dB. Minimizing the network energy consumption through joint routing and compressed aggregation is the goal in [76], with uniformly at random samples routed to a sink through a tree based structure. A scheme to efficiently exchange features in VANETs is developed in [77]. An energy efficient compressed sensing scheme for wireless sensor networks using spatially-localized sparse projections is proposed in [78] by using measurements from clusters of adjacent sensors in order to reduce transmission cost. Differing from the above, [79] proposes spatial domain sparse function recovery at a sink using RW based linear combination of the sensed values. However this scheme has scalability issues related to RW samples required in larger networks.

A CS for manifold learning protocol (CSML) is proposed in [80] for localization in wireless sensor networks. Here, each sensor transmits a subset of distance measurements to a central node. Then the central node reconstructs the full pair wise distance matrix through an L1-minimization algorithm. A CS based approach for sparse target counting and positioning scheme is proposed in [81]. The proposed greedy matching pursuit algorithm (GMP) in [81] complements the well-known signal recovery algorithms in CS theory and proves that GMP can accurately recover a sparse signal with a high probability.

All the successful implementations discussed above share two common factors: uniformly at random sampling and recovery at a base/central station. The focus of this research is on smooth function discovery in a suitable domain, using a pragmatic sampling scheme. Uniformly sampled sensor values require many nodes to participate in propagating sensed values of a limited number of nodes to the BS. We are tempted to ask "why not make use of the sensed information lying on the paths leading to the BS?" Collecting uniformly at random samples also require sensor nodes to be placed and activated uniformly in the sensor field, which is less or even not practical in many of the applications. We propose RW based sampling, and demonstrate centralized and distributed phenomena discovery.

4.4.3. Compressive Sensing Under Random Walk based Sampling in Discrete Cosine Domain. Compressive Sensing [11] is posed as recovering an n-dimensional signal $X(\in \mathbb{R}^n)$ that is k-sparse in its sparse representation$x(\in \mathbb{R}^n$, with $m(\ll n)$ number of samples $y(\in \mathbb{R}^m)$ given by $y = Ax$. If y is a subset of samples of X and X has a sparse representation in a domain whose inverse transform is Ψ, the problem can be re-written as in [17]:

$$y = RX = R(\Psi x) = (R\Psi)x \tag{4.99}$$

where $A = R\Psi$ is called the measurement matrix and R is a subset of rows of an $n \times n$ Identity matrix selected by some probability mass function (pmf). Most of the theoretical recovery bounds in CS are derived from the Restricted Isometric Property (RIP) of the measurement matrix A. RIP requires every combination of support of x many columns of A to be well conditioned as [11]:

$$(1-\delta)\|x\|_2^p \leq \|Ax\|_2^p \leq (1+\delta)\|x\|_2^p \tag{4.100}$$

where δ is called Restricted Isometric Constant (RIC) and is specific for the support of x. Authors of [57] provide a summary of solvers that can be used to solve the CS problem when the RIP is satisfied. The work presented in this dissertation uses L_1 minimization [11, 46] to recover the signal vector. Given y and A, the under-determined system (4.99) is solved for x as:

$$\textbf{minimize } \|x\|_1 \text{ s.t. } Ax = y \tag{4.101}$$

where $\|\cdot\|_1$ is the L_1-norm and $A(\in \mathbb{R}^{m\times n})$ is the sensing/measurement matrix.

The goal is to recover the sparse transformed domain representation of the function/signal. Here, two main design criteria emerge: (1) the choice of the basis/frame, and (2) the row selection scheme. The basis/frame is chosen to provide a sparsest possible representation of the signal. The row selection scheme essentially is the sampling scheme. The probability of failure and the minimal number of samples required, when the measurement matrix is constructed by drawing rows from an orthonormal basis is derived in [21] according to an orthogonalization measure. For example, a measurement matrix constructed by uniformly sampling the Fourier basis meets the above requirements. Deviating from the traditional approach, in this dissertation we sample the basis based on a pragmatic sampling schemes that can be used in a sensor field. Next, we discuss the basis and the sampling scheme selected.

4.4.3.1. *Why Discrete Cosine Basis?* According to [21], operating on a basis where the signal is sparsest, provides highest recovery probability. Therefore, in a WSN deployment to monitor real-world physical phenomena, sensing the Discrete Cosine Transform (DCT) of the phenomenon is rather promising. As reference [82] points out, the DCT of natural signals achieves nearly optimal energy compression - comparable to Karhunen-Loève transform, yielding the fewest coefficients, i.e., the sparsest representation.

4.4.3.2. *Why Random Walk as the Sampling Scheme?* Random routing is based on Random Walk or Brownian motion models and is the basis for a large number of routing algorithms for WSNs [83]. In random routing, each node randomly selects a neighbor and forwards the received message. Rumor routing [83] is an example RW routing protocol, in which messages such as agents and queries, also called rumors, randomly traverse the network. Even when the network is structured and deterministic routing is possible, random routing schemes play a crucial role in WSNs in discovery of resources and disseminating information, especially in the absence of a base station that acts as a global moderator. Moreover, RW motion models are applicable for the case where samples are collected by a carrier. Thus, random routing is highly desirable in WSN applications. However, using RW routing to gather a set of uniformly scattered measurements from a sensor field is rather inefficient. Making the messages traverse in a RW manner, while collecting measurements along the path it traverse is more practical. But, such will not result in a uniform selection of measurements. Instead we receive a set of RW collected samples.

4.4.3.3. *Implementation - Random Walk based Phenomena Discovery.* This section discusses the proposed RW based phenomena discovery algorithm for centralized and distributed realizations.

Centralized Realization of Phenomena Discovery. In a centralized implementation, the network has a base station (BS) with a higher computational capacity. There are many scenarios where a centralized implementation is feasible or even preferable [70]. In this setup, we assume there is a carrier - a robot/vehicle/animal, collecting sensed information while traversing the network on a RW. At the end, the carrier either returns or transmits the collected data to the BS. Then BS will form and solve the CS problem to recover the phenomenon. Under similar conditions, forcing the carrier to collect samples uniformly at random is not pragmatic.

Distributed Realization of Phenomena Discovery. Nodes becoming phenomena aware with distributed schemes without the involvement of a base-station is crucial for many future ubiquitous sensor/actuator network applications. This phenomena awareness may be achieved using messages that continuously disseminate in the network for event/destination discovery or other management purposes. Let X be the vectorized 2D sensed phenomena. Then each node has a corresponding entry in X. For simplicity, we assume nodes are numbered in ascending order from the top left corner to bottom right corner (see Fig. 4.2), which is used as the node ID as well the index in X. In a localized network, physical position information of nodes can be used to organize X. If the network is not localized, a hash function can be used to map some identification of the node to an index in the range of X. Consider the example grid network in Fig. 4.2, where a message generated by node-8 traverses the network in a RW while disseminating information it gathered so far from the nodes it visited. When a node receives a message, it stores the content. Then the node piggybacks its node ID and the measurement to the message and forwards to a randomly selected neighbor. For instance, node-15 may receive the message $[ID_8, T_8, ID_9, T_9]$ from node-9. Node-15 then stores the message and appends its node identification ID_15 and measurement T_15 and transmits to a neighbor. After visits from multiple packets, a node may accumulate a sufficient number of samples for recovery and construct the entire phenomena using the algorithm in Fig. 4.3.

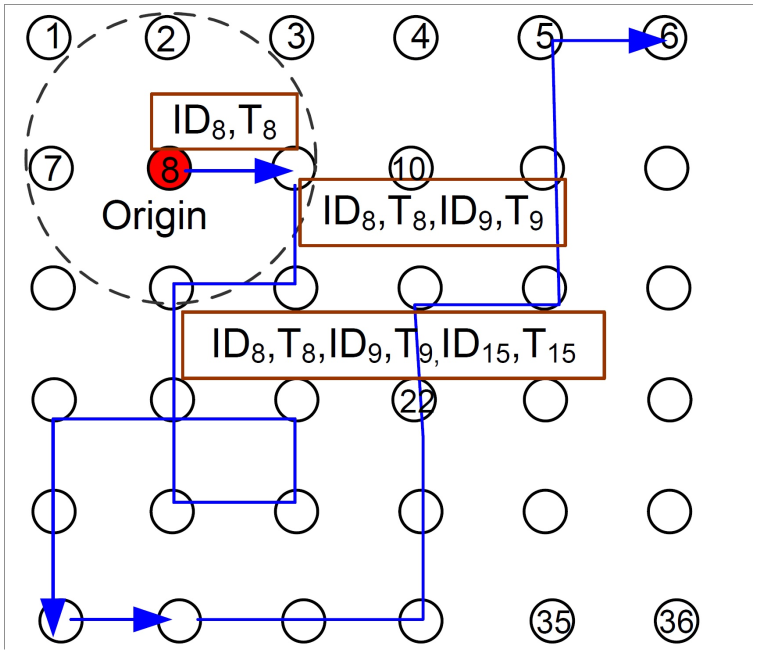


FIGURE 4.2. RW based sample collection on an example grid

In next section we evaluate the probability of recovery failure of this process. The proposed mathematical bound provides a bound on the minimum number of measurements needed to recover the function within a desired error margin.

Reconstruction Failure Probability. The support of the signal vector is the set of indices of non-zero elements. Let x with support S be the signal to be recovered and $|S| = s$ the number of non-zero elements, i.e., sparsity of x. Failure of recovery of a signal with support S is viewed as A being unable to satisfy RIP. In this case, RIP implies that a sub-matrix

```
function PHENOMENA DISCOVERY(Collected samples of the phenomena)
    $\bar{m} \leftarrow$ number of collected samples
    $y \leftarrow T_i, i = 1, \ldots, \bar{m}$
    $Z \leftarrow ID_i, i = 1, \ldots, \bar{m}$
    if $\bar{m} \geq m_T$ then
        for $i = 1, \ldots, \bar{m}$ do
            for $j = 1, \ldots, N_T$ do
                if $j == 1$ then
                    $\Psi(i,j) \leftarrow \sqrt{1/N_T}$
                else
                    $k \leftarrow ID_i$
                    $\Psi(i,j) \leftarrow \sqrt{1/N_T} \cos\left(\frac{\pi(2k+1)j}{2N_T}\right)$
                end if
            end for
        end for
        SOLVE(min. $\|x\|_1$ s.t. $\Psi x = y$)
        $X \leftarrow$ IDCT(x)                                    ▷ Inverse DCT of x
    else                                                          ▷ more sample required
        for $j = 1, \ldots, \bar{m}$ do                           ▷ check whether $i$ is a new sample
            if $ID_i == ID_j$ then
                flag $\leftarrow 1$
            end if
        end for
        flag $\leftarrow 0$
        if flag $== 0$ then
            $ID_{m+1} \leftarrow ID_i$
            $T_{m+1} \leftarrow T_i$
        else
            flag $\leftarrow 1$
        end if
            Forward the packet to a neighbor to which which the packet has not been previously
forwarded.
    end if
return Reconstructed phenomena
end function
```

FIGURE 4.3. Distributed phenomena discovery algorithm implemented at a node.

formed by columns of A over S referred as A_S being nearly orthonormal, i.e.,

$$\|\tilde{A}_S^* \tilde{A}_S - \mathbb{I}\|_2 \leq \delta. \tag{4.102}$$

Here $\tilde{A}_S$ denotes column wise normalized A_S and δ is the RIC. The probability with which the above is unable to be satisfied is defined as the probability of failure ϵ.

$$\epsilon = \mathbb{P}\left[\|\tilde{A}_S^* \tilde{A}_S - \mathbb{I}\|_2 \geq \delta\right] \tag{4.103}$$

Let $X_l = \left(\bar{\Psi}_j\left(t_l\right)\right)_{j\in S}$, then, expected value of $X_l^* X_l$

$$\mathbb{E}\left(X_l^* X_l\right)_{j,k} = \phi_{j,k} \tag{4.104}$$

$$\mathbb{E}X_l^* X_l = \mathbb{I} + \eta \tag{4.105}$$

where $\mathbb{E}$ is expectation, Ψ is an orthonormal basis, t_l is a set of arbitrary indices and $\mathbb{I}$ is the identity. η is the off-diagonal elements of $\mathbb{E}X_l^* X_l$. According to [21] in scenarios such as when the basis is Fourier and the sampling scheme is uniform and η is null; thus, Fourier basis with uniform sampling is widely accepted to provide the optimal performance. Since we are using a different sampling scheme, recovery performance is degraded due to nonzero η, thus, requiring additional number of samples to achieve similar recovery properties such as probability of failure, error in recovered function, etc., compared to those when uniform sampling is used. The general form of the RIP condition given in (4.100) can be re-arranged to:

$$\delta = \max_{S\subset\{1,\ldots,n\},|S|\leq s} \left\|\tilde{A}_S^* \tilde{A}_S - \mathbb{I}\right\|_2 \tag{4.106}$$

From Markov Inequality the probability of failure is bounded above by;

$$\mathbb{P}\left[\|\tilde{A}_S^* \tilde{A}_S - \mathbb{I}\|_2 \geq \delta\right] \leq \frac{E_p}{\delta} \tag{4.107}$$

where E_p is $\mathbb{E}\left\|\tilde{A}_S^*\tilde{A}_S - \mathbb{I}\right\|_2^p$. Substituting for $\mathbb{I}$ from (4.106)

$$E_p = \mathbb{E}\left\|\tilde{A}_S^*\tilde{A}_S - \mathbb{I}\right\|_2^p \leq \mathbb{E}\left\|\frac{1}{m}\sum_{l=1}^{\bar{m}} X_l^* X_l - \mathbb{E}X_l X_l^*\right\|_2^p + \mathbb{E}\left\|\eta\right\|_2^p \qquad (4.108)$$

Note that $\mathbb{E}\|\eta\|_2^p$ is the noise generated due to the deviation from uniform sampling. By solving for E_p

$$E_p^{1/p} \leq D\left[\sqrt{1 + \frac{D^2}{4} + \frac{Q}{D^2}} + \frac{D}{2}\right] \qquad (4.109)$$

where $D = \left(\left(\frac{2}{\sqrt{\bar{m}}}\right)^p 2^{3/4} s p^{p/2} \exp\left(-\frac{p}{2}\right)\right)^{1/p}$. Let $\bar{\kappa} = \sqrt{1 + \frac{D^2}{4} + \frac{Q}{D^2}} + \frac{D}{2}$. Substituting E_p (4.109) in (4.107) and rearranging terms, we obtain the failure probability ϵ

$$\mathbb{P}\left[\|\tilde{A}_S^*\tilde{A}_S - \mathbb{I}\|_2\right] = \epsilon \leq 2^{\frac{3}{4}} s \exp\left(-\frac{\delta^2 \bar{m}}{8K^2\bar{\kappa}^2 s}\right) \qquad (4.110)$$

where $\bar{m}$ is the number of measurements needed under any sampling scheme and K is the upper bound of $\|X_l\|_2 \leq \sqrt{s}$. Complete proof is available on [65].

The theoretical number of samples required for uniform sampling as in [78] is evaluated and shown in Fig. 4.4 under failure probabilities 0.1%; 0.5%; 1%, against sparsity. δ and K were set to 0.5 and 1 respectively. The number of samples required under same probabilities of failure (ϵ), by RW based sampling evaluated by simply solving (4.110) for $\bar{m}$, which is $\bar{m} \geq \frac{K^2\bar{\kappa}^2 s}{\delta^2} \ln\left(\frac{2^{2/3}s}{\epsilon}\right)$, for the same sparsity range is also plotted in Fig. 4.4. Monte Carlo simulation of RW is used to estimate the probability distribution of a message visiting a node and $\bar{\kappa}$ is estimated.

4.4.4. Performance Evaluation. Temperature distribution map of State of Alabama [84] during August averaged over 1951 - 2006 years was used to demonstrate the effectiveness of the RW based phenomena discovery (See Fig. 4.5(a)). There are a total of 7653 data

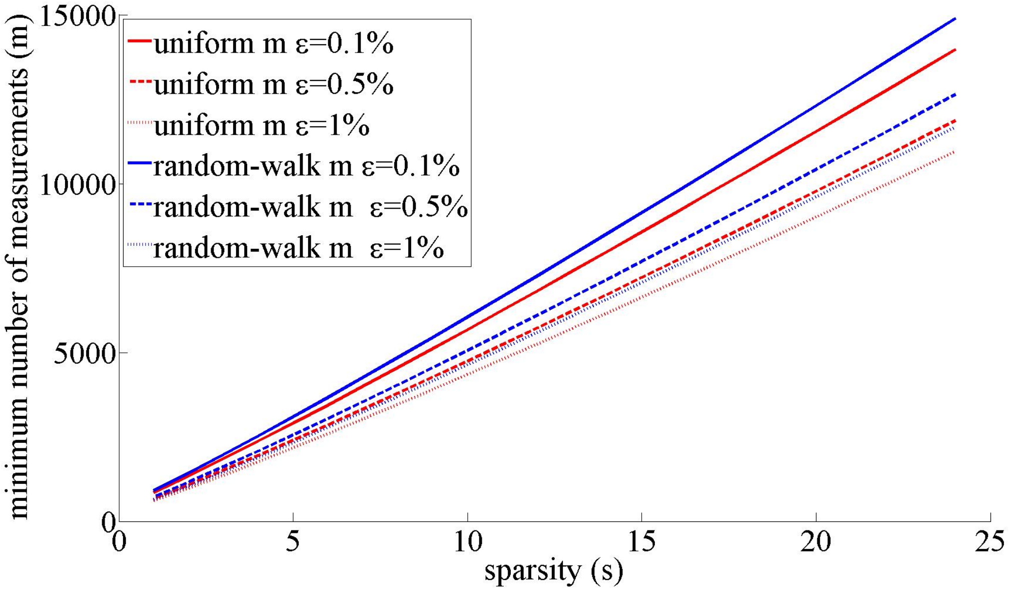


FIGURE 4.4. Variation of m and $\bar{m}$ with sparsity under different failure probabilities (ϵ) for a 100×100 grid network

points in a grid structure where we assume 7653 sensors are deployed. Each node is capable of communicating with its immediate four neighbors, i.e. communication range is one grid segment.

The presented experiments look into the cost of implementing centralized and distributed phenomena awareness with random walk sampling using a suitable basis. Experiments are carried out on MATLAB 2011a and L1 magic [46] is used as CS solver. Performance evaluation metric is the percentage reconstruction error (E_r) of the recovered function defined as:

$$E_r = \frac{1}{N_T} \sum_{k=1}^{N_T} \left(\left| X_k - \bar{X}_k \right| / X_k \right) \times 100 \tag{4.111}$$

where N_T is the total number of samples in the function, which is the same as the total number of sensors in the network. X_k and $\bar{X}_k$ are the k^{th} sample of the original function and the reconstructed function respectively.

First, a pre analysis of temperature distribution in Fig. 4.5(a) was performed to identify its sparsity level. The number of significant coefficients defines the sparsity level. If the function to be reconstructed is sparse in DCT domain, we expect only a few significant DCT coefficients. It was found that to approximate the temperature function in Fig. 4.5(a) within a 0.1% error, 3183 DCT coefficient out of 7653 were required. This implied that even in DCT domain the selected phenomenon is not as sparse as expected.

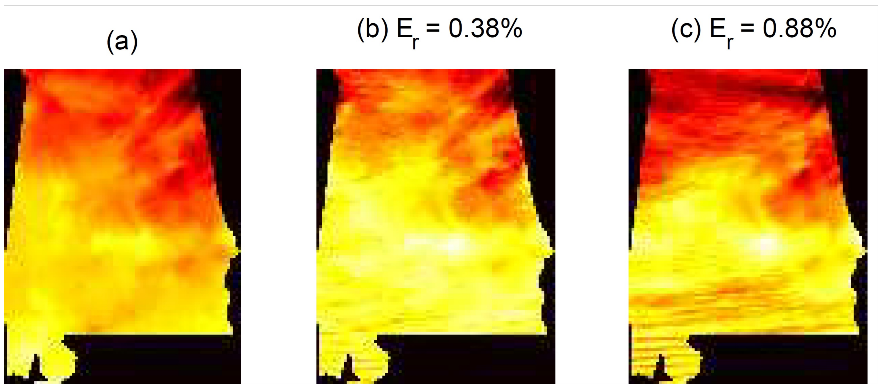
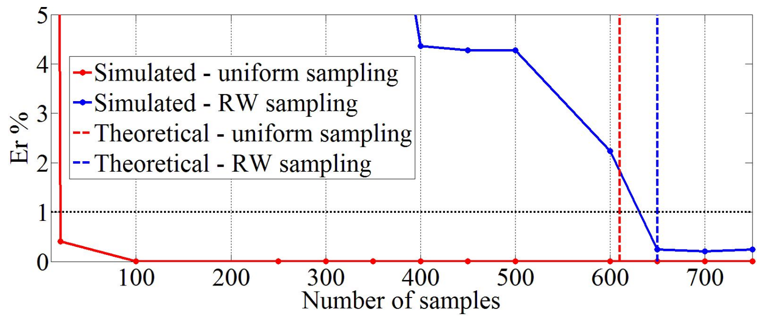
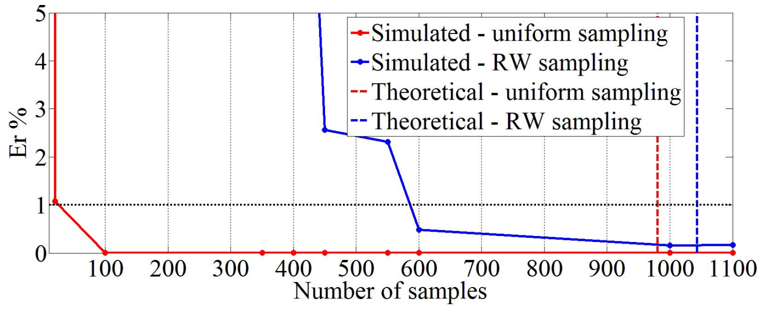


FIGURE 4.5. (a)Average temperature distribution of State of Alabama in August. 7653 sensors in total are available. (b) Reconstructed image based on 2583 samples collected at the BS by a single carrier (c) Reconstructed image at randomly selected node when 1056 samples were collected in that node

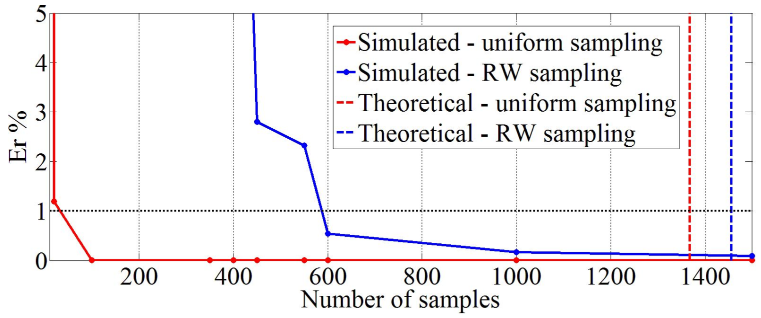
In order to compare the theoretical prediction with the simulated, we begin our performance evaluation by finding the number of samples needed to recover the approximated version of phenomena in Fig. 4.5(a) based on the most significant s DCT coefficients. The number of coefficients to be recovered is considered as the sparsity (s) of the function. Here a message with a predefined TTL (Time To Live) is disseminated into the network. Note that due to the possibility of revisiting to the same node the message will not be able to collect TTL many unique samples. Recovery error of reconstruction against the samples collected by the message is plotted as in Fig. 4.6, for three sparsity cases: 2, 3 and 4. As Fig. 4.6 indicates, the empirical values for the number of measurements needed to obtain an E_r of 1%, for sparsities 2, 3, 4 is less than 600. Dashed lines in Fig. 4.6 indicate the theoretical



(A) $s = 2$



(B) $s = 3$



(C) $s = 4$

FIGURE 4.6. Variation of E_r with the empirical number of samples used when (a) s=2 (b) s=3 (c) s=4. Dashed lines show the theoretical number of samples needed and the dotted line show the Er=1% level.

estimates for the number of samples required under each sampling scheme. The theoretical values are an over-estimate as they are based on satisfying the sufficient conditions of recovery.

Although such natural phenomena can be approximated by a sparse representation with only a few non-zero coefficients, in next simulation results, we aim to recover the original dataset - not an approximation of the original with low sparsity.

4.4.4.1. *Performance of Phenomena Discovery at the Base Station.* Here, the base station is assumed to be at (0,0). A carrier (robot) collects sensed information and node IDs while moving from one node to another in a RW of step size one. The maximum number of steps that the carrier will take is set to 4000 when there is a single carrier. Since revisiting to the same node twice is allowed, the carrier may collect less than 4000 samples. Fig. 4.5(b) shows an example recovery under RW sampling at a BS when 2583 unique samples were available. In reality, the true sparsity of the phenomenon is unknown. Therefore, the theoretical number of samples needed is undetermined. In Fig. 4.7 we demonstrate the variation of E_r with the number of samples used. The variation of error as two carriers collectively gather the same number of samples is also plotted in the same Figure. In the experiment with two carriers, each carrier has a TTL of 2000. As can be seen, performance in terms of E_r depends more on the number of samples than the number of carriers used. Even though collecting samples under uniform distribution is difficult in practice, we show the recovery error under simulated uniform sampling as a comparison. Error performance under RW sampling is only about 0.2% away that of under uniform sampling, when 2000 samples are available.

4.4.4.2. *Performance of Distributed Phenomena Discovery.* To the best of our knowledge, CS based distributed phenomena discovery in WSNs is proposed here for the first time. We envision that future WSNs will evolve over their lifespan and become increasingly aware of the sensed phenomena. Thus, the proposed scheme will provide a cost effective infrastructure.

We use the same temperature dataset for the distributed phenomena awareness implementation and evaluation. In a WSN where there is no fixed BS, random routing is used for event and sink discovery [67]. Those messages can be used to make network learn the phenomena being observed. Note that while phenomena discovery at a BS used carriers incurring

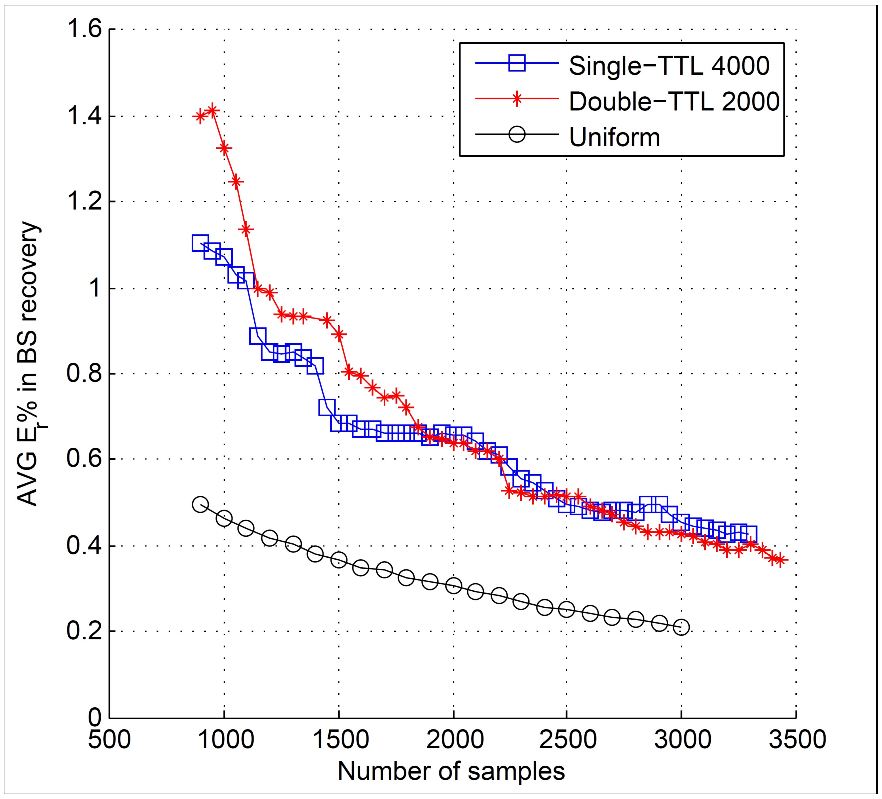


FIGURE 4.7. Variation of average error with number of samples used for reconstruction in centralized recovery. Single carrier with 4000 steps in total and two carriers with 2000 steps per each were used.

cost, the distributed implementation uses packets already disseminated in the network incurring no additional cost. In the simulation, 1000 messages with TTL 300 are generated at a randomly selected node and traversed on a RW. Messages may revisit the same node but message will carry only one sample per visited node. A view of the phenomena at a randomly selected node, which has collected 1056 samples from the messages that passed through it, is given in Fig. 4.5(c). Figure 4.8 shows the average error E_r in the recovered phenomena at different nodes. The mean E_r is calculated over nodes with the same number of samples collected.

Next, we consider the convergence of the entire network achieving phenomena awareness in a distributed implementation. Figure 4.9 shows the mean rate of nodes achieving phenomena awareness under two different TTL values. From Fig. 4.8 we deduce that a node

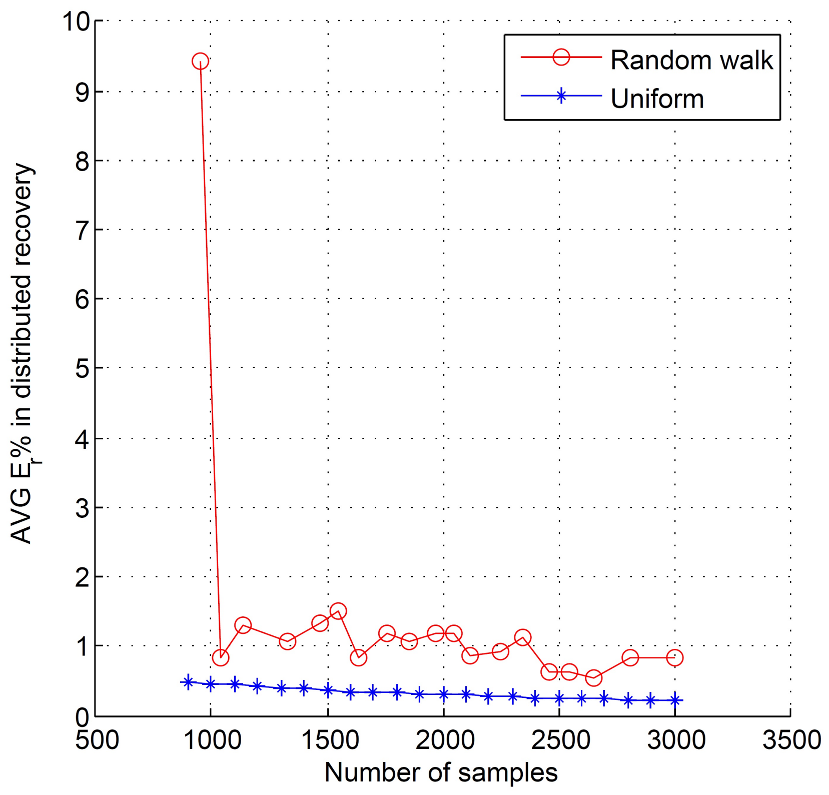


FIGURE 4.8. Variation of average error with number of samples used for reconstruction in a distributed manner. Evaluation is after 1000 messages with 300 TTL disseminated in the network.

needs at least 1000 samples to become aware of the sensed phenomena with an E_r of less than 2%. When TTL is 300, at least 1200 messages need to be disseminated in the network, while when TTL is doubled the required number of messages reduces to less than 400. Note that the network considered has 7653 nodes. If the traditional uniform sampling was used, for entire network to become aware of the phenomena, 1000 randomly selected nodes need to flood the network which leads to at least 7,653,000 transmissions. But the proposed approach achieves a map with a similar E_r with at least 240,000 transmissions with TTL 600, providing approximately 96% reduction in the number of transmissions.

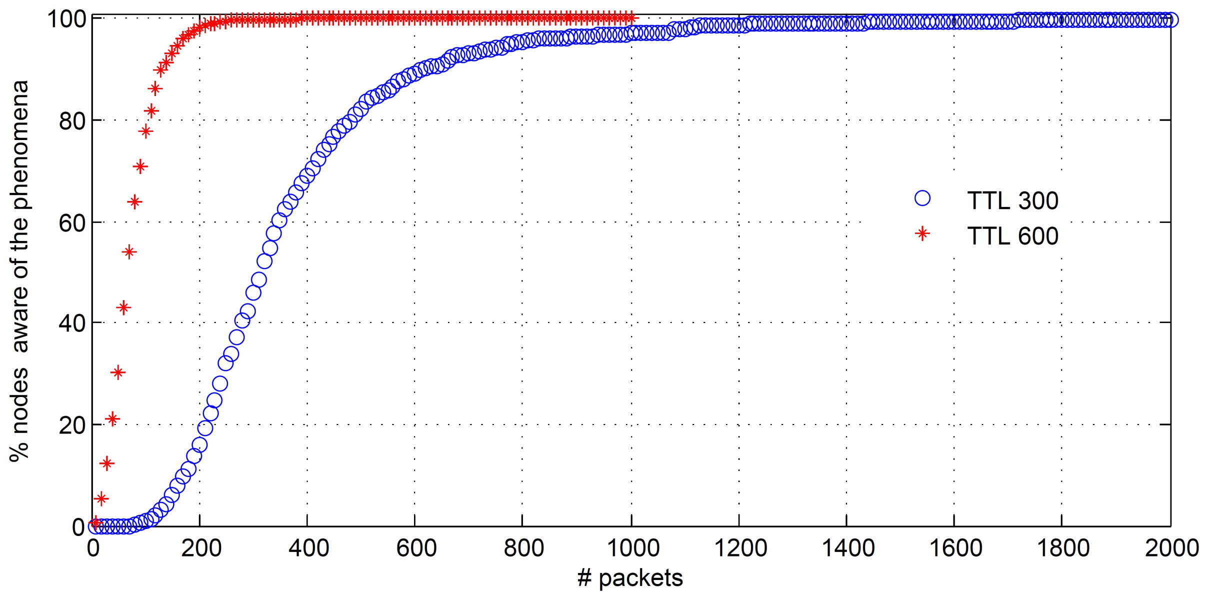


FIGURE 4.9. Convergence rate of nodes achieving phenomena awareness in a distributed implementation when messages has TTL 300 and 600.

4.4.5. CONCLUSIONS. A novel implementation using compressive sensing for phenomena awareness is proposed. The proposed algorithm provides nodes with network-wide knowledge of the events observed. Moving beyond the traditional approach of uniform sampling based CS for function recovery, we illustrate that RW based sampling can practically and successfully be used for phenomena awareness at base-stations and at each sensor without a BS, with minimal additional samples. An upper bound for the probability of successful recovery with a given error percentage is also derived. The derived bound provides an approximate number of samples required to recover a function under a selected basis and a sampling scheme.

We considered random walk as the mobility model due its wide spread usage in WSNs. But other mobility models as random waypoint model, Markovian model etc. can also be used in the same manner. Performance bounds for CS based phenomena discovery using a frame - an over complete basis, instead of an orthogonal basis and other practical sampling schemes that accurately captured by motion models are under investigation.

4.5. Subsurface Plume Tracking Using Sparse Wireless Sensor Networks

Deployment of sensors for tracking chemical plumes such as those in the subsurface can be quite expensive. A compressed data gathering scheme for chemical plume tracking is presented. With only a fraction (about 25%) of the measurement points required to achieve a given spatial resolution this novel application of compressed sensing can track the plume within 7% accuracy compared to the case of a full sensor array. The scheme also gathers and re-distributes information of the sensors to the entire network, compressing with Discrete Wavelet Transform. The scheme can be used to disseminate global tracking information to sensors as well, with savings in communications by a factor of 5 in average, if such capability is required. The scheme is capable of interpolating randomly missing sensor points with significant accuracy. It also supports data fusion as a simple addition of coefficients requiring no changes to the message length.

4.5.1. Introduction.

In a variety of situations in environmental science and engineering, some of them related to national security, it is necessary to track and monitor chemical plumes, to make predictions on their future behaviors, and to evaluate potential risk to humans and ecological environments. While it is desirable to have data on chemical concentrations collected at high spatial and temporal resolutions to facilitate reliable predictions, the cost and other logistical factors associated with installing sampling wells limit the monitoring accuracy and the resolution achievable. Specific situations dealing with tracking dissolved contaminant plumes in flowing groundwater require the collection of water samples from sparsely distributed monitoring wells. With current technology, these samples are delivered from field sites to testing laboratories to conduct chemical analysis to determine dissolved concentrations. For accurate tracking, this tedious and expensive process has to be repeated frequently. Recent advances in Wireless Sensor Network (WSNs) have the potential to alleviate this labor intensive and time consuming task of data gathering. With wireless sensor nodes (motes) that measure and transmit the concentrations in situ in sampling wells in real-time, the need for manual collection of samples can be avoided. However, installation and maintenance of a large number of WSN nodes (containing sensors interfaced with motes) required for large-scale and evolving plumes can also be expensive. Minimizing the number of sensors will allow for this real-time monitoring technology to be a viable option. Sampling at regular spatial intervals (e.g., with sensors arranged in a rectangular grid) can be challenging at field sites, wherein an unstructured deployment of motes would be more realistic. The distribution of such nodes will be determined by other factors associated with the geography and accessibility of deployment locations (e.g., to avoid buildings and other land infrastructure features). Effectively using a random deployment of sensors to obtain satisfactory results is also of interest.

Many interesting phenomenon in chemical plume tracking, seismic activity monitoring, animal migration tracking, etc., results in data in the forms of configurations with fairly regular boundaries and smooth gradients over the sensor field. Image processing algorithms often deals with similar regular features. A number of transforms such as Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT) are known to effectively compress images with regular shapes. Moreover these transforms are realizable as linear transforms and can easily be implemented on sensor motes with limited processing capacity. The goal of this work is to use such image processing techniques to reduce the amount of information transferred back and forth on sensor networks, while improving the resolution for a given set of sensors. The technique also enables the redistribution of the state of a part or the entire network back to each node efficiently. This feature could be used to develop future smart sensing schemes that operate more intelligently.

4.5.1.1. *Related work.* WSNs are widely used in applications related to environmental and habitat monitoring, reconnaissance and building automation [85–89]. A WSN node, also called a mote, consists of a processor and a wireless transceiver interfaced to a sensing device to measure the interested phenomenon, [90].

The feasibility of using WSNs for subsurface chemical plume tracking applications has been demonstrated in [91, 92]. Some of the challenges posed in traditional data gathering as described in [93] prevent the accurate tracking of the plume. Low maintenance, miniature sensing devices such as electrical conductivity sensors [94] are placed in wells at different depths for monitoring the plumes. The WSN has to be configured for efficient operation [95], and can coupled to numerical models to form a closed loop system that uses WSN readings to calibrate the model, while the model provides information for data collection and node activation [96].

The potential for sensor readings related to many applications to be compressible is well known. Most work exploits the local correlation of the readings. Reference [97] provides an information theoretic derivation, based on correlations of sensor readings, on savings possible for one- and two-dimensional networks. The Hierarchical Cooperation scheme presented in [97] achieves logarithmic scaling on traffic and schedule lengths. A data compression scheme based on wavelet decomposition and reconstruction is applied hierarchically at cluster heads in [98] with the goal of reducing waste in transmitting raw data to the data-center. Reference [99] points out the fact that wavelets approximate missing data on sensor readings when applied on a correlated structure. Their method - Data Correlation Compression (DCC) relies on Gaussian assumption of sensor data. A scheme utilizing historical data to reduce the amount of information needed to be transmitted is presented in [100]. A localizing scheme based on correlation is used to improve the accuracy on the multi-level clustering structure. Spatio-temporal correlation can be exploited to identify redundant sensors as indicated in [101], where wavelet transform and Fourier transform are used for compressing time series on individual sensors.

4.5.1.2. *Contribution.* A novel approach for reducing the number of sensors used and/or improving the resolution of the measurements in plume tracking applications is presented. With a fairly small fraction of sensor readings, the scheme is capable of approximating the status of the entire network to a significant accuracy. With the proposed approach, the energy spent gathering the status of the network and then redistributing that information back to the network is a magnitude less than with the conventional approach. Fusion has no effect to the message length, thus, it requires no additional bandwidth.

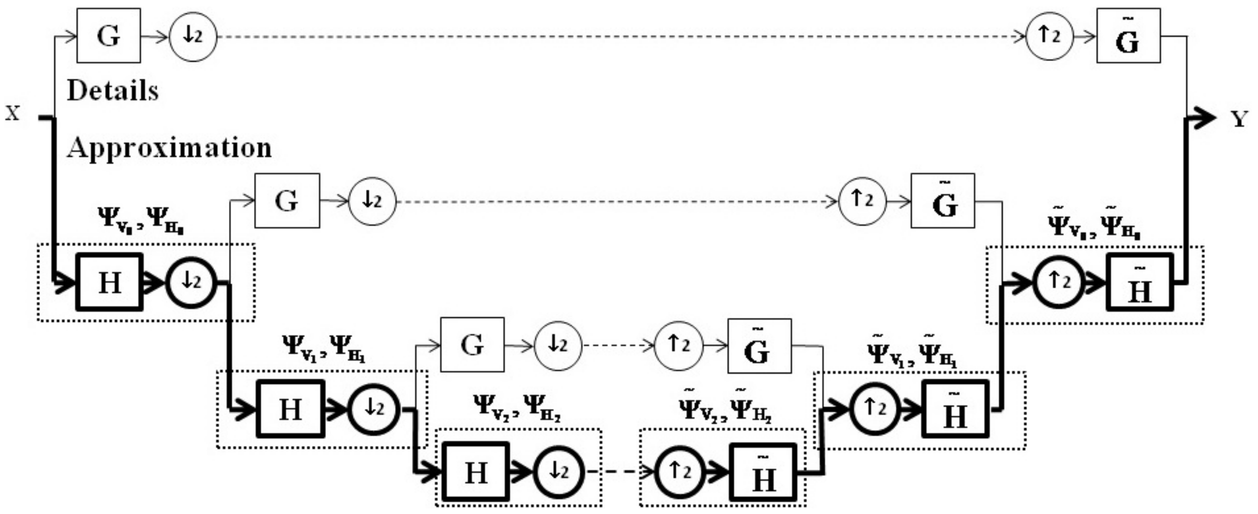
Section 4.5.2 discusses the theoretical background. An analysis on the communication cost is presented in Section 4.5.3. Sensor deployment is addressed in Section 4.5.5 while Section 4.5.6 presents the results of the work. Scheme and results are evaluated in Section 4.5.7, followed by conclusions in Section 4.5.8.

4.5.2. Discrete Wavelet Transform Based Compression.

We view the measurements collected by the sensors as pixels of an image. We assume that the underlying chemical concentration image has pixels on a fine grid. However, the wireless sensors are randomly deployed and only sparsely populate this grid. Our objective is to reconstruct the image on the fine grid from the sensor measurements. To minimize communication volume, we also wish to compress the data collected by each sensor before processing. We show in this dissertation that a simple DWT compression provides reasonable results. We start by reviewing two-dimensional DWT.

4.5.2.1. *DWT compression and reconstruction.* Most real images are compressible in the DWT domain. The DWT successively splits an image into an approximation component, which captures the smooth part of the image, and several detail components, as shown in Fig. 4.10. Roughly speaking, at each level, the H_i filter is a "low-pass" filter that passes the smoother part of the image and the G_i is a "high-pass" filter that passes detail components. As the DWT branches are traversed the size of the signal decreases diadically (down-sampling by 2). Since an image is two dimensional, each transformation is applied in two dimensions, the horizontal (row-wise) and vertical (column-wise) and as we proceed through successive branches the number pixels in the DWT image is reduced by a factor of 4.

To compress the image, we discard the detail components and only keep the coarsest approximation component produced by the bottom most branch in Fig. 4.10.



X
Y
Details
Approximation
G
↓2
↑2
G̃
H
H̃
Ψ_{V_0}, Ψ_{H_0}
Ψ_{V_1}, Ψ_{H_1}
Ψ_{V_2}, Ψ_{H_2}
$\tilde{\Psi}_{V_0}, \tilde{\Psi}_{H_0}$
$\tilde{\Psi}_{V_1}, \tilde{\Psi}_{H_1}$
$\tilde{\Psi}_{V_2}, \tilde{\Psi}_{H_2}$

path in bold - coarsest approximation and recovery
H_i - i^{th} level approximation filter
$\tilde{H}_i$ - i^{th} level reconstruction filter
Ψ_{V_i} - i^{th} level vertical approximation filter
$\tilde{\Psi}_{V_i}$ - i^{th} level vertical reconstruction filter
Ψ_{H_i} - i^{th} level horizontal approximation filter
$\tilde{\Psi}_{H_i}$ - i^{th} level horizontal reconstruction filter
X - original image
Y - reconstructed image

FIGURE 4.10. Block diagram of wavelet transform

The DWT however needs to be calculated in a distributed fashion, where each sensor computes the contribution of its own measurement to the coarse approximation term without having the knowledge of measurements from the other nodes in the network. To accomplish this, we work with point spread functions (PSF) associated with each sensor node as we now describe.

Consider the coarsest approximation branch in Fig. 4.10. The coarsest approximation A for this branch can be expressed in matrix form as:

$$A = \Psi_V X \Psi_H \tag{4.112}$$

where, X is the original image defined under the fine grid, Ψ_V is the DWT matrix in vertical direction and Ψ_H is the DWT matrix in horizontal direction.

The vertical DWT matrix Ψ_{V_i} (accounting for down-sampling) at the i^{th} level for an $n \times n$ image is given by ($j = 1, \ldots, n/2^i, k = 1, \ldots, n/2^{i-1}$):

$$\Psi_{V_i} = \begin{cases} n_{j-2(k-1)} & i \geq j - 2(k-1) \geq 1 \\ 0 & \text{otherwise} \end{cases} \tag{4.113}$$

where, l is the length of the filter and h_i's are the scaling coefficients of the filter. The horizontal transform Ψ_{H_i} is the transpose of Ψ_{V_i}. The coarse approximation component A_i at level i is given by

$$A_i = \Psi_{V_i} X_{i-1} \Psi_{H_i} \tag{4.114}$$

where, A_0 is equal to X. If an L-level DWT the coarse approximation component A can be calculated from (4.112) with:

$$\Psi_V = \Psi_{V_{L-1}} \Psi_{V_{L-2}} \cdots \Psi_{V_1} \Psi_{V_0} \tag{4.115}$$

$$\Psi_H = \Psi_{H_0} \Psi_{H_1} \cdots \Psi_{H_{L-2}} \Psi_{H_{L-1}} \tag{4.116}$$

We can write the coarse term approximation to A as:

$$A = \sum_{i,j} A_{(i,j)} = \sum_{i,j} \Psi_V(\cdot, i) X(i,j) \Psi_H(j, \cdot) \tag{4.117}$$

where $A_{(i,j)} = \Psi_V(\cdot, i) X(i,j) \Psi_H(j, \cdot)$ is the contribution of the (i,j) pixel $X(i,j$ to the coarse approximation A. The notations $(\cdot, i)$ and $(j, \cdot)$ respectively mean the i^{th} column and the j^{th} row of a matrix. Thus, we can think of PSF of pixel (i,j) as:

$$PSF(i,j) = \Psi_V(\cdot, i) \Psi_H(j, \cdot) \tag{4.118}$$

If all $X(i,j)$ on the fine grid were available we could obtain the coarse approximation component A by simply transmitting PSFs scaled by the corresponding pixel value. The advantage is that each PSF can be calculated locally without knowledge of other pixels. The size of each PSF matrix is $2/n^L \times n/2^L$.

In our case sensors sparsely populate the image grid and we only have access to a small number of pixels at random locations. Nonetheless, we show that by combining the PSFs associated with these sensor locations we can still obtain a reasonable reconstruction of the chemical plume concentration.

Coefficients of the PSF depend on the wavelet transform and the filter selected. They can be built into the sensor motes prior to deployment. Therefore computing the contribution of each sensor to the approximation can be done locally, which involves only scaling the PSF by the sensor reading.

Another advantage of this method is that computing the approximation of a part of or the entire sensor field becomes an addition of the contributions of each of the sensors. This allows sensors nodes to fuse their contributions to a single message conveniently and opportunely. For example, if the sensors are reporting to a base station over a tree, an intermediate node will add its coefficient matrix to the coefficient matrices it receives from its children nodes and transmits the result to its parent. Thus, there will be only one transmission per link in the tree carrying all the information of the subtree below. Regardless of the position of a link in the tree the size of the composite PSF matrix that the link needs to communicate stays the same.

Once the base-station receives the sum of all contributions the image can be approximated by applying the inverse-DWT, as shown in Fig. 4.10. The synthesis filter pair $(\tilde{H}, \tilde{G})$ and analysis filter pair (H, G) are quadrature mirror filters, satisfying the perfect reconstruction condition [102]. Note that in the synthesis tree all detail components are zero.

However, this reconstruction can also be done at each sensor as will be shortly explained. While it is not essential for a node to know about the plume spread in the entire flow region, we envision future intelligent plume tracking systems where sensing operations within a locality may benefit by having global information on plumes. A simple extension to the scheme provides the ability to re-distribute the global information to cluster heads or individual sensors efficiently.

Table 4.1. Daubechies D4 Scaling Coefficients

Coefficient	Value
h_1	$\left(1+\sqrt{3}\right)/4\sqrt{2}$
h_2	$\left(3+\sqrt{3}\right)/4\sqrt{2}$
h_3	$\left(3-\sqrt{3}\right)/4\sqrt{2}$
h_4	$\left(1-\sqrt{3}\right)/4\sqrt{2}$

Suppose the coarse approximation component A is broadcasted from the base-station, then the inverse-DWT approximation can be calculated as:

$$\tilde{X} = \tilde{\Psi}_V \tilde{A} \tilde{\Psi}_H \tag{4.119}$$

where, $\tilde{A}$ is the sensor network approximation to the coarse approximation component A. The matrices $\tilde{\Psi}_V$ and $\tilde{\Psi}_H$ are vertical and horizontal synthesis matrices. They are of the form (4.115) and (4.116) respectively, but constructed from elements of similar to (4.113).

4.5.2.2. *Implementation with Daubechies D4 wavelet.* We present a sample implementation of a single level compression using Daubechies D4 wavelet. The coefficients of the H filter are shown in Table 4.1. The single level vertical and horizontal DWT matrices are given by:

$$\Psi_V = \begin{pmatrix} h_1 & h_2 & h_3 & h_4 & 0 & 0 & 0 & 0 & \\ 0 & 0 & h_1 & h_2 & h_3 & h_4 & 0 & 0 & \\ 0 & 0 & 0 & 0 & h_1 & h_2 & h_3 & h_4 & \\ & & & & & & & & \ddots \end{pmatrix} \tag{4.120}$$

$$\Psi_H = \Psi_V^T \tag{4.121}$$

The PSF to be loaded to each node is computed using (4.118). The PSF for a node is a few non-zero elements often appear as a single patch on a mostly empty matrix. During reporting, each node will scale the pre-loaded PSF by its measurement. Details of the compression algorithm are summarized in Fig. 4.11.

```
function COMPRESSED REPORT(reading, reports from children)
    Initialize report ← []                    ▷ empty matrix of the size of PSF
    for $i = 1, \ldots,$ number of children do
        report $i$ ← report of child node $i$
        report += report $i$                  ▷ a matrix addition
    end for
    my report ← PSF × reading                 ▷ a matrix scaling
    report += my report                       ▷ a matrix addition
    Forward report to parent
end function
```

FIGURE 4.11. Pseudo code for the compression algorithm

As a simple example, let us consider the 10×10 matrix of sensor readings shown in Fig. 4.12.

0.00	0.00	1.76	2.47	2.67	2.47	1.76	0.00	0.00	0.00
0.00	0.00	2.20	2.80	2.98	2.80	2.20	0.00	0.00	0.00
0.00	1.05	2.47	3.02	3.18	3.02	2.47	1.05	0.00	0.00
0.00	1.36	2.62	3.14	3.30	3.14	2.62	1.36	0.00	0.00
0.00	1.45	2.67	3.18	3.33	3.18	2.67	1.45	0.00	0.00
0.00	1.36	2.62	3.14	3.30	3.14	2.62	1.36	0.00	0.00
0.00	1.05	2.47	3.02	3.18	3.02	2.47	1.05	0.00	0.00
0.00	0.00	2.20	2.80	2.98	2.80	2.20	0.00	0.00	0.00
0.00	0.00	1.76	2.47	2.67	2.47	1.76	0.00	0.00	0.00
0.00	0.00	0.93	1.96	2.20	1.96	0.93	0.00	0.00	0.00

FIGURE 4.12. A sample 10×10 measurement matrix

Each sensor is also assigned with a PSF calculated according to (4.118). For example node (7,4) would calculate its PSF using (4.118) for a single level compression by multiplying the 7th column of Ψ_V and 4th row of Ψ_H to produce:

$$\text{PSF}(7,4) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -0.029 & 0.188 & 0 & 0 & 0 \\ -0.062 & 0.404 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{4.122}$$

According to Fig. 4.12 the reading of the sensor node (7,4) is 3.02 . Therefore the contribution of the node (7,4) is obtained by scaling PSF(7,4) by 3.02 .

Similarly, all the nodes will calculate their PSF and then scale by their measurement. The approximation of the entire sensor field is obtained by summing up all the approximations generated by individual nodes.

$$A = \sum_{i=1}^{10} \sum_{j=1}^{10} A_{(i,j)} \tag{4.123}$$

$$A = \begin{pmatrix} 0.210 & 5.011 & 5.794 & 1.456 & -0.008 \\ 1.754 & 5.899 & 6.485 & 3.265 & -0.233 \\ 2.000 & 6.027 & 6.583 & 3.549 & -0.269 \\ 0.638 & 5.474 & 6.204 & 2.026 & -0.066 \\ -0.002 & 3.731 & 4.539 & 0.839 & 0.000 \end{pmatrix} \tag{4.124}$$

To obtain an approximation to the chemical plume image we inverse-DWT is applied on A and the resultant matrix is shown in Fig. 4.13.

4.5.3. Exchange of Sensor Data. a network in which the sensors are placed on an $n \times n$ grid as shown in Fig. 4.14(a) with a tree communication structure rooted at the center

0.05	0.08	1.60	2.71	2.57	2.77	1.29	0.25	0.20	-0.12
0.08	0.15	1.83	3.08	2.89	3.08	1.49	0.38	0.25	-0.15
0.41	0.76	2.12	3.21	3.03	3.19	1.91	1.02	0.37	-0.34
0.65	1.23	2.39	3.40	3.22	3.36	2.28	1.52	0.48	-0.49
0.62	1.16	2.35	3.37	3.19	3.33	2.22	1.45	0.46	-0.47
0.65	1.23	2.39	3.39	3.21	3.35	2.28	1.53	0.48	-0.49
0.34	0.64	2.10	3.24	3.06	3.22	1.86	0.91	0.35	-0.31
0.13	0.23	1.90	3.14	2.95	3.14	1.57	0.47	0.27	-0.18
0.07	0.12	1.49	2.52	2.41	2.61	1.22	0.26	0.19	-0.12
-0.04	-0.07	1.15	2.03	1.99	2.19	0.88	-0.02	0.10	-0.05

FIGURE 4.13. The reconstructed measurement matrix

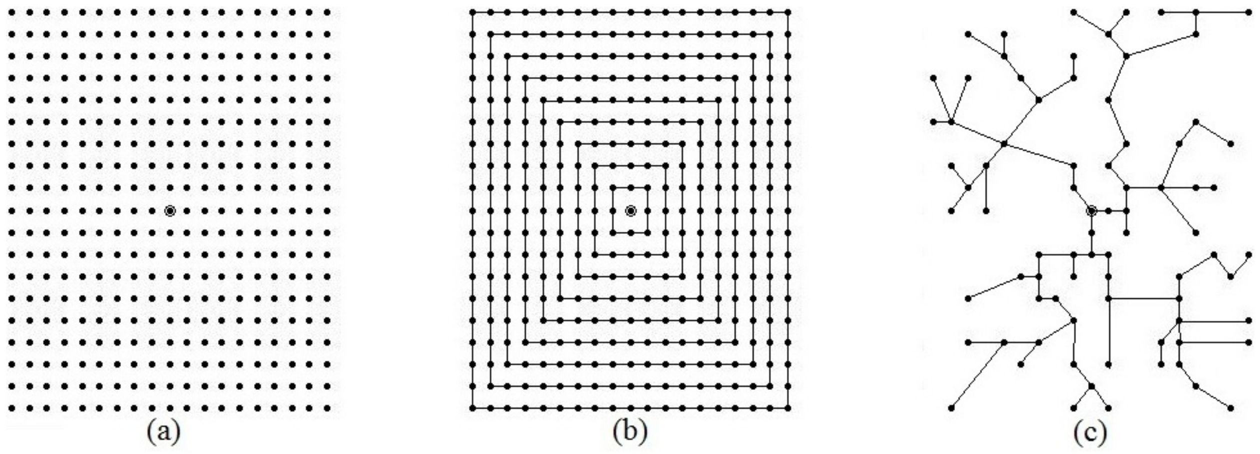


FIGURE 4.14. (a) Nodes placed in a grid with root at the center (b) Levels of nodes (c) A random node deployment with a tree communication structure.

of the grid. We assume that a node is capable of communicating with its eight immediate neighbors. The levels of the tree then form co-centric squares, with the maximum depth of the tree at $n/2$. The average depth of a node from the root is $n/6$.

Communication cost is two folds: reporting and re-distributing. During reporting, sensors report their readings to the root. Then, root informs the status of the network to each of the sensors during re-distributing. Reporting costs can be alleviated by making sensors not report, if the reading is null. However, the node still needs to take part in communication to relay reports from the subtree descending from itself.

4.5.3.1. *Conventional Reporting.* Under the conventional monitoring scheme, each node reports its measurement along with its ID or coordinates. This report has to be to the root, i.e. $n/6$ times on average. Since there are n^2 nodes in the network, the total reporting cost is $\mathcal{O}(n^3)$. If the null readings are not transmitted, then communication cost reduces to $\mathcal{O}(kn^2)$, where k is the number of nodes having a non-zero reading. Further, overhead in transmitting individual reports as separate packets can be saved by packing a few if not all reports received from the subtree to a single message along with its report. Such a fusion saves overhead cost, yet no savings are made on the amount of payload transmitted. It is to be noted that in the conventional scheme, reporting the location and the reading provides no loss of information.

4.5.3.2. *Compressed Reporting, Fusion and Recovering Missing Data.* Compressed reporting exploits the compressibility of data. Instead of reporting the reading and location information tuple, nodes report wavelet coefficients. Further, data is fused by adding coefficient matrices. As in conventional scheme, the nodes having a null reading do not contribute to the coefficient matrix.

Each contributing node will produce a coefficient matrix, which is a small patch of non-zero elements. By putting together these patches, the approximation for the entire matrix is formed. Patches are in fact added onto the coefficient matrix - which enable an effective fusion scheme, where the message length does not change. Under conventional reporting, reading of each node was stored in the message separately. Such would lengthen message length as the message arrives at the root. But with the compressed reporting, nodes keep adding their contributions onto the existing message. Therefore the length of the message is not affected.

The size of the coefficient matrix is $m \times m$, with m equal to $n/2^L$, where L is the number of levels of compression applied. When fewer nodes have readings, instead of transmitting the entire coefficient matrix, the patch and its location information can be transmitted to save cost.

Compressed reporting imposes a smoothing operation on the measurements. Thus, it automatically approximates readings of the locations which provided no input. If a malfunctioning node feeds in an abnormally large contribution (an outlier) that would be suppressed as well.

Let us demonstrate recovering missing data points using the matrix in Fig. 4.12 by randomly dropping some 10 measurements out of the 100. The resultant is shown in Fig. 4.15. This doesn't correspond to a sparse sensor network, but still demonstrates how missing data points are handled. The actual chemical plume example presented in Section 4.5.6 corresponds to a truly sparse network where the sensors populate only 25% of the grid points. beginfigure[!ht]

0.00	0.00	1.76	2.47	2.67	2.47		0.00	0.00	0.00
0.00	0.00	2.20	2.80	2.98	2.80	2.20		0.00	0.00
0.00	1.05	2.47	3.02	3.18	3.02	2.47	1.05	0.00	0.00
		2.62		3.30	3.14	2.62	1.36	0.00	0.00
0.00	1.45	2.67	3.18	3.33	3.18	2.67	1.45	0.00	0.00
0.00	1.36	2.62	3.14	3.30	3.14	2.62		0.00	0.00
0.00		2.47	3.02	3.18	3.02	2.47	1.05	0.00	0.00
	0.00	2.20	2.80	2.98	2.80	2.20	0.00	0.00	0.00
0.00	0.00	1.76	2.47	2.67		1.76	0.00	0.00	0.00
0.00	0.00	0.93		2.20	1.96	0.93	0.00	0.00	0.00

FIGURE 4.15. A sample 10×10 measurement matrix with 10 missing values

The approximation derived from the available nodes if shown in Fig. 4.16a and the reconstruction is shown in Fig. 4.16b. By comparing Fig. 4.16b with Fig. 4.12, it can be

0.305	5.351	5.603	1.045	-0.031
1.139	3.701	6.462	3.413	-0.085
1.802	6.027	6.730	2.595	-0.238
0.179	5.758	5.741	2.026	0.000
0.210	2.510	3.451	0.649	0.000

(A) Approximation for the sensor readings with missing nodes

0.09	0.16	1.56	2.61	2.39	2.50	1.09	0.09	0.14	-0.10
0.10	0.19	2.06	3.44	2.98	3.01	1.33	0.11	0.16	-0.12
0.29	0.52	1.58	2.42	2.78	3.27	1.89	1.01	0.40	-0.28
0.43	0.77	1.37	1.89	2.80	3.60	2.36	1.67	0.58	-0.41
0.51	0.96	2.06	2.98	3.11	3.45	2.03	1.08	0.39	-0.37
0.61	1.16	2.51	3.64	3.34	3.42	1.91	0.83	0.30	-0.38
0.22	0.42	2.11	3.39	2.99	3.05	1.71	0.75	0.32	-0.25
-0.03	-0.07	1.93	3.39	2.80	2.77	1.54	0.63	0.32	-0.15
0.07	0.12	1.24	2.08	1.99	2.15	1.03	0.26	0.17	-0.10
0.07	0.13	0.69	1.11	1.34	1.62	0.62	-0.04	0.06	-0.04

(B) Reconstruction

FIGURE 4.16. Approximation and reconstruction with missing data

noted missing points are approximated quite closely compared to the range of measurements.

4.5.3.3. *Hybrid reporting.* Compressed reporting captures the status of the (partial) network by an $m \times m$ matrix, instead of the number of nodes many tuples. However, at lower depths of the tree, where only a few nodes observe some reading, reporting the $m \times m$ matrix or the coefficients patch may be too costly. The hybrid scheme proposes to use conventional scheme until the number of nodes with a non-zero reading is below $\frac{1}{2}m \times m$. Once this threshold is reached, readings are to be transformed to the $m \times m$ coefficient matrix. Implementation of the hybrid scheme is explained in Fig. 4.17. Until transmitting the coefficient matrix is effective than reporting raw data, conventional scheme is followed, denoted by reporting mode : 0. When the list of raw data and the coordinate information grow past the threshold, the coefficient matrix is formed. Thereafter, all nodes contribute in the compressed mode.

```
function GET MODE OF REPORTING
    enum mode $\{0, 1\}$                                        ▷ 0 :raw data, 1 :compressed data
    message length $\leftarrow 0$
    for $i = 1, \ldots,$ number of children do
        Receive message $i$
        message length $+=$ length(message $i$)
    end for
    if message length $> 0.5 \times m^2$ then
        reporting mode $\leftarrow 1$
    else
        reporting mode $\leftarrow 0$
    end if
return reporting mode
end function

function CONSTRUCT MESSAGE
    if reporting mode $== 0$ then
        message $\leftarrow [\,]$
        for $i = 1, \ldots,$ number of children do
            receive message $i$
            message $\leftarrow$ [ message; message $i$]
        end for
        if measurement $!= 0$ then
            message $\leftarrow$ [ message; [coordinates, measurement]]
        end if
    end if
    if reporting mode $== 1$ then
        message $\leftarrow [\,]$
        for $i = 1, \ldots,$ number of children do
            receive message $i$
            message $+=$ message $i$
        end formessage $+=$ PSF$\times$reading
    end if
end function

function TRANSMITTING MESSAGE
    if reporting mode $== 1$ then TRANSMIT(message)
    end if
    if reporting mode $== 0$ then
        if length(message) $< 0.5 \times m^2$ then TRANSMIT(message)
        else
            msg $\leftarrow [\,]$
            for $i = 1, \ldots,$length(message) do
                coord $\leftarrow$ message $i$(coordinate)
                value $\leftarrow$ message $i$(measurement)
                msg $+=$ PSF(coord)$\times$value
            end forTRANSMIT(msg)
        end if
    end if
end function
```

FIGURE 4.17. Pseudo code for hybrid reporting scheme

4.5.3.4. *Re-distribution.* Future smart sensing schemes on large networks would benefit by being aware of the state of the entire network. Thus, a phase where the status of the network is re-distributed back to the network is discussed here. The methodology of re-distributing status of the network is intuitive from the hybrid scheme. The status of the $n \times n$ network is compressed to $m \times m$. But if the number of non-zero values in the network is below $\frac{1}{2}m \times m$, re-distribution is more effectively done with a conventional approach, where the location and reading information tuple is broadcast. Otherwise the compressed matrix delivers the status of the network more effectively.

Then at each node, the inverse transform is performed to recover the status of the entire network. By doing so each and every node becomes aware of the entire network.

4.5.3.5. *Potential Issues.* Here we discuss issues associated with the scheme that would be of interest to certain applications.

Reconfiguring nodes: since each node use a unique combination of Ψ_V and Ψ_H, reconfiguring can be tedious. However, using nodes programmable over the network would alleviate the effort.

Speed of the plume: a cycle of reporting and dissemination is expected to complete while the plume is effectively stationary. If the cycle is a slow process, the picture built using the reports would be inaccurate.

Ringing effects: this is an issue natural to lossy compression. Since the high-frequency components are discarded, a slight ringing artifact builds on the image.

Blurred image: approximation is analogous to a low-pass filter, which smoothens the image. Thus, reconstructed images would be less crispy and more blurred.

Effects of missing contributions: although the scheme interpolates the missing locations quite accurately according to a smoother description of the plume, it draws energy from the available contributions. Thus, missing contributions causes noise on the available.

Poor alignment: the simulation results presented later assumes a worst case of a purely random deployment. A pure random deployment would have a few cluttered nodes and a few blank areas. Thus, the approximation would be more biased to the cluttered locality and less towards the empty regions. However, actual deployments are not purely random and will suffer less from such effects.

4.5.4. Analytical Results. To identify uniquely and for communication sensor nodes need $\gtrsim \log_2 n^2$ bit long address. If we assume the reading produces some b bit floating point, the cost of reporting under conventional scheme is $\approx (\log_2 n^2 + b)\, \frac{n}{6}$ per node. If only k nodes read non-zero values and report, then the total reporting cost is $\approx k\,(\log_2 n^2 + b)\, \frac{n}{6}$. Pure compressed reporting scheme requires transmitting an $m \times m$ matrix. Thus, the reporting cost would be $km^2b'n/6$, where b' is the length of a coefficient. Reporting cost can be saved by reporting patches instead of the entire matrix, where applicable. Moreover, the hybrid scheme would provide much savings.

Reporting is economical for certain choices of wavelets and levels, which also determine the required precision of the coefficients. Nonetheless, reporting in compressed form is essential to implement compressed re-distribution in a distributed form. As well to interpolate for the missing location, compression scheme has to be employed at reporting, irrespective of the communication cost. Re-distributing is effectively achieved for large values of k and n, i.e. for large network with a large fraction of nodes reading non-zero measurements.

The key advantage of the compressed reporting and re-distributing is the information of a vast network is represented using only a few coefficients. Thus, less information needed to be transmitted in order to deliver the status of the network.

4.5.5. Sensor Deployment. When sensor nodes are placed on a regular grid, they can be matched to pixels of an image (Fig. 4.14(a) and Fig. 4.14(b)). To calculate the contribution made by each pixel (sensor) for the approximation, each sensor is fed with a corresponding PSF. Thus, at reporting, each node will scale its PSF by the reading and report the resultant matrix. Further, nodes fuse readings simply by adding the contribution matrices.

When all the contribution matrices of all the sensors are added, the approximation for the entire sensor field is formed. This approximation can be then transmitted back to the sensor field, so that each of the sensors learns the status of the entire network.

When constructing the approximation, if the contribution of some of the pixels were not available, an interpolated value will be automatically assigned to those pixels. This relaxes the need of a complete grid which is attractive for many applications, and is discussed next.

4.5.5.1. *Random points on a Grid.* A random deployment of sensors can be treated as a sparse deployment on a grid (Fig. 4.14(c)). As pointed out above, DWT based approximation scheme fills out the missing grid points with interpolated values based on the available grid points automatically. This allows using the same scheme even for a random deployment of sensor nodes.

As before, each node is assigned with its PSF matrix based on the grid point the sensor is located. The rest of procedure is the same. When reconstructing, an image of the size of the grid is formed, where missing grid points are assigned with an interpolated value based on the wavelet used.

4.5.5.2. *Representing a More Realistic Scenario.* Deploying sensors on an exact grid is difficult and not economical for many environmental sensing applications. A deployment exhibiting characteristics of a random deployment can be considered more realistic. The nodes can be considered to be randomly placed at points on a finer grid for computational convenience. Another issue with wireless sensor networks is the availability of the nodes. At a given time it is quite likely that a significant fraction of the nodes may either be sleeping, or even dead. Once the random deployment is treated as a sparse deployment over a fine grid, unavailability can be accounted as a much sparser deployment. Thus, when resolved, measurements will be interpolated for on each point on the fine grid.

4.5.6. Results. In this section we evaluate the compressed data reporting and dissemination scheme using a dataset corresponding to a subsurface plume. The dataset and numerical results are presented next.

4.5.6.1. *Synthetic Plume Data Set.* The data that is needed in field problems will come from a set of sensors that are installed in water quality monitoring wells. As such data set was not available, a synthetic data set using a groundwater flow (MODFLOW) transport model (MT3DMS) was generated [103–106]. Synthetic data emulating a propagating plume over a period of 3 years, collecting daily samples are used as experimental data for this work. The synthesizer software allows placing sensors and making measurements at any desired location. By placing sensors at a complete fine grid, the actual plume is recognized. Then sensors are placed at random location for the experiments. Sensor field is represented as a 64×64 pixel image. The readings are compressed using a two-level Daubeschie-4 wavelet. The compressed image is 16×16.

4.5.6.2. *Numerical Results.* At each time interval, selected based on plume tracking application, a snapshot of the sensor field is built using the compressed reporting method described in Section 4.5.3. For our experiment, the time interval was selected to be a day.

TABLE 4.2. Error of Approximation

	Mean over entire sensing period		Max over entire sensing period	
	Mean over a snapshot	Max over a snapshot	Mean over a snapshot	Max over a snapshot
Error (%)	2.5	55.4	9.5	82.4

The error is defined as the deference between the calculated value and the actual value normalized to the largest reading (which is the range of the measurements), and expressed as a percentage.

Four versions of errors are defined. Given a snapshot, the mean of the errors and the maximum of the errors can be taken. Then over the entire sensing duration (3 years in our case) the mean and the maximum of above two can be taken.

Transmission cost is evaluated in terms of the number of transmissions. The experiment used double precision floating point values for both measurements and coefficient matrices. Thus, the actual transmission cost is a factor of the number of transmissions made.

4.5.6.3. *Accuracy.* The proposed scheme exploits the effectiveness of lossy compression. Inevitably, some of the information is destroyed during the reporting phase. Table 4.2 assesses the error introduced by the approximation.

More realistic networks are represented as a sparse deployment of nodes over a grid. Their performance is comparable when a large fraction of nodes are unavailable on a grid. Table 4.3 summarizes error performance when 25%, 50% and 75% of the nodes are unavailable. Table 4.4 shows the mean and the standard deviation of the mean error over 100 random network settings. It can be noted that mean error is small and it varies very little.

4.5.6.4. *Communications Cost Savings.* Compression based data gathering and re-distributing scheme saves floating point transmissions by a factor of 5 in average. When hybrid scheme

Table 4.3. Effect of Partial Availability on the Error

Dead nodes 25%				
	Mean over entire sensing period		Max over entire sensing period	
Error compared against	Mean over a snapshot	Max over a snapshot	Mean over a snapshot	Max over a snapshot
Actual	2.5	55.4	9.5	82.4
Approximation	2.7	24.9	4.9	47.7
Dead nodes 50%				
Actual	4.9	76.7	13.6	98.8
Approximation	5.3	41.1	9.0	82.6
Dead nodes 75%				
Actual	7.0	87.9	18.4	103.5
Approximation	7.9	56.1	11.8	84.9

Table 4.4. Mean and Standard Deviation of the Accuracy

Dead node %	Error compared against	Mean	Standard deviation
25	Actual	3.2	0.09
	Approximation	2.7	0.16
50	Actual	4.9	0.14
	Approximation	5.3	0.19
75	Actual	7.0	0.15
	Approximation	7.9	0.16

is employed instead, the saving reaches a factor of 10. Figure 4.18 shows the total cost of using the three schemes over the sensing period.

Largest factor of the savings is accounted to the re-distributing phase as shown in Fig. 4.19. However, for the re-distribution to be implemented in a distributed fashion, the reporting scheme has to be implemented in either the compressed form or the hybrid form. The hybrid scheme improves the compressed scheme further by a factor of 2 in average. The performance of the hybrid scheme over the compressed scheme is presented in Fig. 4.20.

4.5.6.5. *Approximating Missing Data.* DWT coefficients automatically approximate values for the missing locations during reconstruction. Figure 4.21 displays the approximation capacity of the scheme. Only 25% of the sensors were activated in the sensor field. This

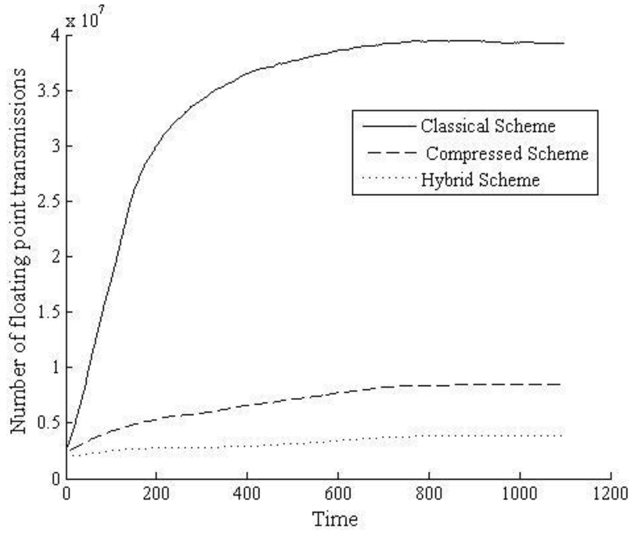


FIGURE 4.18. Cost of re-distributing sensor information over the entire network over time.

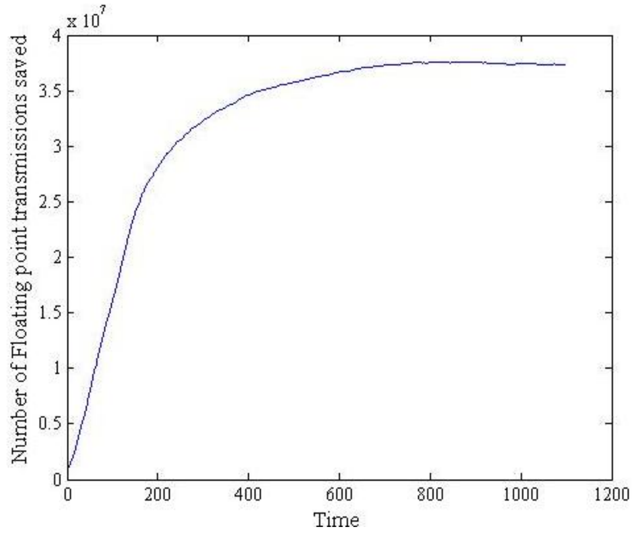


FIGURE 4.19. Amount of floating point transmissions saving by compression over time.

could also be interpreted as, only 25% of the grid points actually contained sensors. Figure 4.21(a) shows a snapshot of the plume to be detected. But only some random 25% of the grid points indicated in Fig. 4.21(b) are available for measurements. The non-zero measurements provided by the available sensors are indicated in Fig. 4.21(c). With coefficients for these non-zero measurements the plume is approximated as in Fig. 4.21(d). It is to be

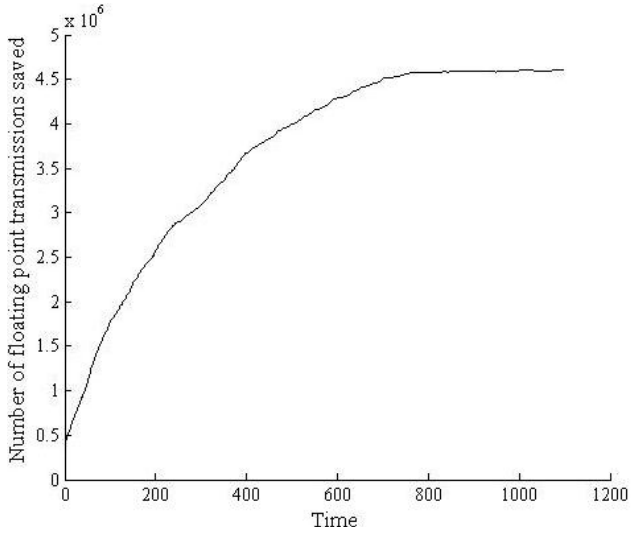


FIGURE 4.20. Amount of floating point transmissions saved using hybrid reporting instead of compressed reporting over time.

noted that the mean error between the approximated reconstruction using only 25% of the measurements is only 7% as shown in Table 4.3.

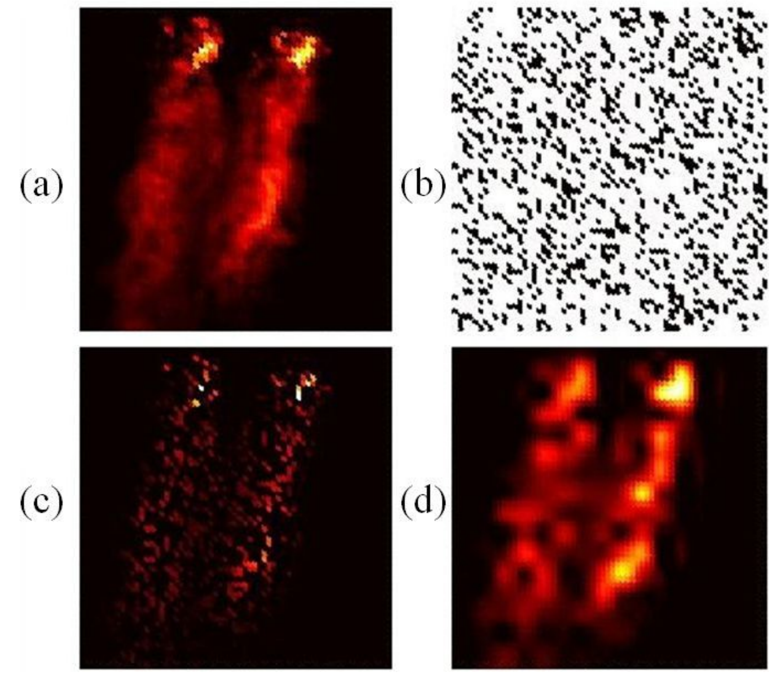


FIGURE 4.21. (a) The actual plume (b) a sample deployment of sensors (c) non-zero reading provided by 25% of sensors (d) approximate plume reconstructed.

4.5.7. Discussion. The goal of the presented scheme is to gather and re-distributed sensor data from each of the sensors to entire network cost effectively. The communication structure is a tree rooted at the center of the network. All the nodes observing the interested phenomenon generates a report and pass it up the tree. Thus, a description of the entire network is generated at the root. Then the root sends down this information back to the network, making all the nodes aware of the entire network.

Under conventional scheme each node reports its reading and the location information, and all the nodes take part in passing this information to the root. The root collects all the information and build giant picture of the network which is then passed down to the network. The conventional scheme does not take into account the compressibility of data. Although it preserves perfect accuracy, most applications tolerate errors to a certain degree to account for noise which is inevitable in measurements. Compressed re-distributing scheme proposed exploit the tolerance to mild loss of information. The coefficients also enable data fusion. Thus, when multiple messages are to be transmitted on the same link, they can be fused to a single message saving overhead. Moreover, the fusion does not change the data length, whereas under the conventional scheme the length of the message is increased when multiple messages are packed.

Compressed scheme reduces the operations at the root. Under the conventional scheme, the root has to gather and form the giant message containing information of the entire network. In the compressed scheme the root has no more operations than a regular node in the network. It sums the coefficients and pass on to the children nodes.

Although compressed reporting alone may not be communication effective, it is essential to facilitate interpolation of missing points, improve resolution and for a distributed implementation of the dissemination scheme. So that the burden on the root is alleviated, and producing a distributed deployment of the scheme. The hybrid scheme utilizes the effective components from both the conventional and compressed schemes. It prevents forming a large coefficient matrix where data is effectively transmitted conventionally, but also applies compression later on, to utilize the advantages in both the schemes.

Computation requirement at sensors nodes are commendable as well. Compression and decompression require matrix multiplication, which is an $\mathcal{O}\left(n^2\right)$ floating point operation. Fusion requires matrix addition which is $\mathcal{O}(n)$ floating point operation. The PSF needed for each node is proposed to be preloaded to each node. The hybrid scheme requires a list of potential PSFs of its children which can also be pre-loaded.

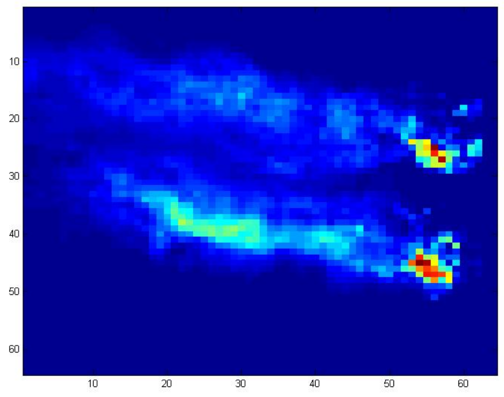
4.5.8. Conclusions.

The scheme estimated the state of the entire network within a 7% error bound using only 25% of the measurements, and demonstrated a communication savings by factor of 10 when applied for the plume data. Thus, the scheme is capable of improving resolution of the measurements made, and also to reduce the number of sensors to be used to achieve a given error bound.

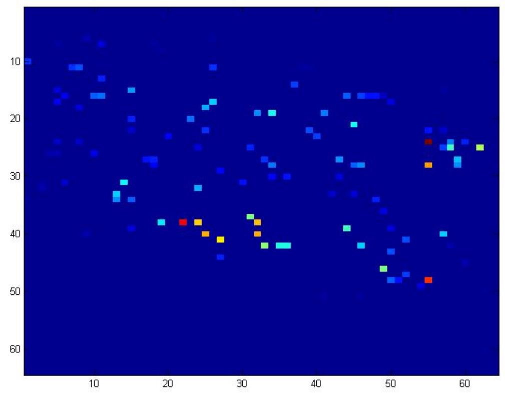
Hybrid scheme exploits the effective components from conventional and the compressed reporting schemes and cuts down the communication cost by a magnitude. Computation and memory requirements needed for all the operation in the feasible range for most common place sensor motes.

4.6. Combined Matrix Completion and Compressive Sensing based Image Reconstruction

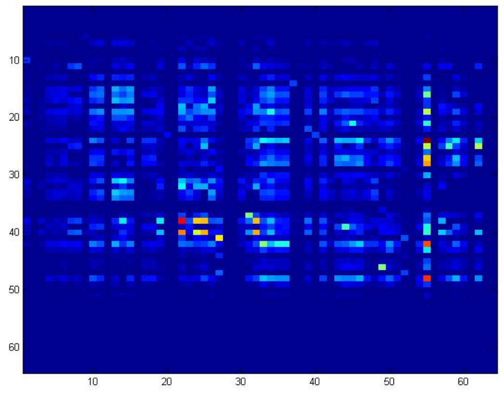
In this section we attempt to employ both matrix completion and compressive sensing to recover a highly under-sampled image. We use the same chemical plume data used in prior sections. In the example presented in Fig. 4.22 only 5% of the points are sampled.



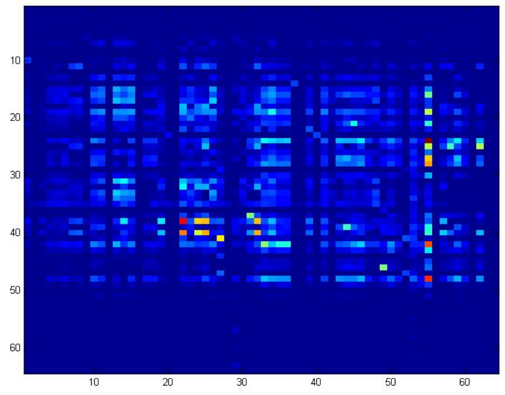
(A) Chemical Plume



(B) 5% Samples



(C) Matrix Completion



(D) Matrix Completion and Compressive Sensing

FIGURE 4.22. Reconstruction using matrix completion and compressive sensing.

The chemical plume is shown in Fig. 4.22a and the samples are shown in Fig. 4.22b. Then matrix completion is applied on the samples and its reconstruction is shown in Fig. 4.22c. A key limitation of matrix completion is that it does not take smoothness into account. Thus, we add another phase to enforce smoothness. Due to the similarity of chemical plumes to natural images, we select a DCT basis as the sparsity basis and perform compressive sensing over the recovered image. This step identifies significant DCT coefficient that describe the underlying plume. Then we recover the plume information as shown in Fig. 4.22d which has only a 4% reconstruction error.

4.7. Conclusions

We employed methods such as compressive sensing, wavelet transform and matrix completion for data recovery in this chapter. The developed techniques are applied on a few WSN applications to recover data at a lower cost. The developed methodologies enabled reconstructing sensor field information from a sparse set of samples as well as enable phenomena awareness. Further, recovery bounds of compressive sensing are derived for any sampling measure.

CHAPTER 5

Data Feature Extractions

5.1. Introduction

Extracting features from the gathered data is the focus of this chapter. We begin by taking a closer look at Robust Principal Component Analysis. As the sufficient conditions established in [6] are too conservative and the conditions in [5] do not provide all the information, there is no guidance to predict whether a combination of a low-rank matrix and a sparse matrix would recover. Thus, we investigate the empirical recovery region of RPCA. We establish the recovery regions for a variety of matrices and also look into a cross validation principle to determine whether a decomposition is a recovery. Validity of the established boundaries are tested on real-world matrices as well. Then we focus our attention to features of network data. We propose a scheme to extract network data features and compute derived features. Then we apply this scheme to detect a few network attacks. We also develop a methodology to extract subtle patterns in data. These findings are tested on real PCB capacitance measurements. The objective of this breadth of work is to find candidate methods to extract features of various nature.

5.2. Experimental Recovery Regions for Robust PCA

The principle of *Robust Principal Component Analysis* (*RPCA*) is to additively resolve a matrix into a *low-rank* and a *sparse* component. The question that arises in the application of this principle to experimental data is, "when is this resolution an identification of the actual *low-rank* and *sparse* components of the data?" That is, when is recovery successful? And, given a resolution, how can we know it is a recovery of the underlying matrices? In this paper we report several experimental findings: (1) the subset of matrices that satisfy published sufficient conditions is quite small compared to the set of matrices that successfully recover; (2) successful recoveries can only be expected at low fractional *ranks* and *sparsities*; (3) where recovery is unsuccessful, the returned matrices tend to be near half-*rank* and half-*sparsity*; (4) the demarkation between the region of consistent recovery and consistent failure is narrow, indicating a phase change in recoverability. We demonstrate these findings with a variety of synthetic matrices that are faithful to matrices appearing in practice. Furthermore, we apply and verify these results on real-world matrices.

5.2.1. Introduction. *Robust Principal Component Analysis* (*RPCA*) decomposes an input data matrix into a sum of a *low-rank* matrix and a *sparse* matrix. If the input matrix is in fact a sum of an unknown "true" *low-rank* matrix (which is low rank but not sparse) and an unknown "true" *sparse* matrix (which is sparse but not low-rank), *RPCA* "recovers" the original unknown matrices. In this paper we report experimental evidence for "recoverability" of matrices consisting of *low-rank* plus *sparse* components, using *RPCA*.

A *low-rank*—plus—*sparse* decomposition is of interest in many applications. A few classes of applications are reviewed in [5], namely : video surveillance, face recognition, Latent Signal Indexing (LSI) and ranking and collaborative filtering. The authors of [6] review *rank-sparsity* decomposition for statistical model selection, computational complexity and system identification. *RPCA* for cyber-security is discussed in [107–109]. *RPCA* is also used in a number of image processing applications such as texture extraction [110], image alignment [111], image tag refinement [112] and image signature analysis [113].

Among a few interpretations, *RPCA* is viewed as making classical *Principal Component Analysis* (*PCA*) robust against "gross" perturbations [5]. That is, it separates principal components of a matrix in the presence of additive *sparse* perturbations – hence the name *Robust PCA*. This can also be viewed as "recovering" a *low-rank* matrix from *sparse* and “gross” corruptions. In another point of view, *RPCA* can be viewed as a tool to extract *sparse* signals concealed by *low-rank* background interferences. The work presented here focuses on the “recovery” of both *low-rank* and *sparse* components.

Following [5] and [6], we model a data matrix $Y \in \mathbb{R}^{n\times n}$ as composed of a *low-rank* “baseline” matrix B and a *sparse* “anomalies” matrix A. The idea is to “recover” the two components from Y with *RPCA*:

$$Y = B + A \xrightarrow{RPCA} L + S = Y \tag{5.1}$$

The output *low-rank* matrix L and *sparse* matrix S solve

$$\arg\min_{L,S} ||L||_* + \lambda||S||_1 \qquad s.t. \quad L + S = Y \tag{5.2}$$

where $||\cdot||_*$ is the nuclear-norm, $||\cdot||_1$ is the one-norm, and λ is a weighting parameter. Our goal is to recover B via L and A via S.

The results of this paper suggest that the existing sufficient conditions for recovery via *RPCA* derived in [5] and [6], while certainly fundamental to the theory of Robust PCA, are so seldom satisfied for experimental data that they do not reliably predict when a resolution will actually be a recovery. These sufficient conditions are met only at extremely low *fractional-ranks* and *fractional-sparsities*, where recovery is also successful. At relatively small *ranks* and *sparsities* sufficient conditions are not met, but recovery is satisfactory. At moderate to high *ranks* and *sparsities* sufficient conditions are not met and recovery is unsuccessful. When recovery is unsuccessful, the resultant low-rank matrix is near half-rank and the sparse matrix is near half-sparse. This leads to a cross validation principle to determine failed recoveries. Further, we report a narrow phase change between the regions of successful recovery and unsuccessful recovery. We empirically establish ranges of *rank* and *sparsity* where recovery is successful. These ranges are estimated using a variety of synthetic matrices that represent a range of matrices appearing in practical problems. Then these results are validated on real-world matrices obtained from actual data.

5.2.2. Background. Conditions presented in [6] state that if B is "true" *low-rank* (meaning, it is not also *sparse*) and A is "true" *sparse* (meaning, it is not also *low-rank*) then *RPCA* recovers exactly. The two metrics ξ and μ presented in [6] measure these respective properties. There, a small $\xi(B)$ indicates that *low-rank* B is *not sparse* and a small $\mu(A)$ indicates that *sparse* A is *not low-rank*. The sufficient conditions for exact recovery of B and A via *RPCA* are stated as

$$\xi(B)\mu(A) \leq \frac{1}{6}\ . \tag{5.3}$$

The metrics ξ and μ are defined as follows. Let us denote the *Singular Value Decomposition* (*SVD*) of B as $U\Sigma V^T$ where $B \in \mathbb{R}^{n\times n}$ and $rank(B) = r$ with orthonormal $U, V \in \mathbb{R}^{n\times r}$. Then a measure of "true" *low-rank*-ness in [6] is

$$\xi(B) = \max_{N\in T(B), ||N||_2\leq 1} ||N||_\infty \tag{5.4}$$

where $T(B) = \{UX^T + YV^T \mid X, Y \in \mathbb{R}^{n\times r}\}$ is the tangent space at B w.r.t. the variety of $n \times n$ matrices whose *rank* is less than or equal to $rank(B)$. Further, *incoherence* bounds ξ as

$$inc(B) \leq \xi(B) \leq 2\, inc(B)\ . \tag{5.5}$$

Here, *incoherence* is

$$inc(B) = \max\left\{\max_i ||P_U e_i||_2, \max_i ||P_V e_i||_2\right\} \tag{5.6}$$

where P_U is the *rank-r* orthogonal projection onto the dimension-r subspace $\langle U \rangle$, and $\{e_i\}_1^n$ is the standard basis for $\mathbb{R}^n$. $||P_U e_i||_2^2$ may be interpreted as the cosine squared of the principal angle between e_i and the subspace $\langle U \rangle$.

A measure of "true" *sparsity* [6] is

$$\mu(A) = \max_{N \in \Omega(A), ||N||_\infty \leq 1} ||N||_2 \tag{5.7}$$

where $\Omega(A) = \{N \in \mathbb{R}^{n \times n} \,|\, supp(N) \subseteq supp(A)\}$ is the tangent plane at A w.r.t. the variety of *sparse* $n \times n$ matrices whose *support* size is less than or equal to $supp(A)$. The maximum and minimum matrix degrees bound μ above and below as

$$deg_{min}(A) \leq \mu(A) \leq deg_{max}(A) \tag{5.8}$$

where deg_{min} is the minimum number of *non-zero* elements along a row or a column, and deg_{max} is the maximum number of *non-zero* elements along a row or a column.

Using *incoherence* as a surrogate for ξ and *maximum degree* as a surrogate for μ, the recovery condition (5.3) is re-stated in [6] as

$$inc(B) deg_{max}(A) \leq \frac{1}{12} \; . \tag{5.9}$$

This condition provides a convenient test of sufficiency for recovery. This work is further extended in [114] where the interest is to recover the *column space* of the *low-rank* matrix.

The authors of [5] provide two different sufficient conditions for the *low-rank* component and the *sparse* component, but these conditions do not appear to be verifiable from data. *Stable Principal Component Pursuit* [115] provides a viable scheme to recover a *low-rank* component buried in a combination of small *entry-wise* and gross *sparse* noise. The *matrix* *completion* problem is viewed and solved in [55] as recovering a *low-rank* matrix by observing only a fraction of the entries. Work presented in [108, 109, 116] addresses the recovery of a *low-rank* matrix corrupted with both gross *sparse* and *entry-wise* noise, by observing only a fraction of the entries.

5.2.3. Experimental Setup. Here we discuss the construction and decomposition of test matrices. Though sufficient conditions depend only on the *eigenvectors* of the *low-rank* component and *support* of the *sparse* component, there remains the question of what role the *eigenvalues* play in recovery. Therefore we consider a range of different *eigenvalue structures* for *low-rank* matrices and a few different *magnitude distributions* for *sparse* matrices. Though these are by no means exhaustive, they cover a set of matrices of theoretical and practical interest.

5.2.3.1. *Test Matrices.* We experiment with a few classes of *low-rank* and *sparse* matrices with controllable *ranks* and *sparsities.* These matrix types occur commonly in first and second order multivariate analysis, likelihood-ratio testing, radar array processing, etc. Further, as shown later in Section 5.2.4.4, the matrices considered here faithfully indicate the behavior of real-world matrices.

Low Rank Matrices. Inspired by the *random orthogonal model* in [6] and [24], we build five types of *low-rank* matrices, namely, first and second order *Gaussian*; *Wishart*; and first and second order *Vandermonde* matrices.

Begin with the positive-definite matrix $C \in \mathbb{R}^{n\times n}$. Give it the eigenvalue decomposition $C = U\Sigma^2U^T$, $U^TU = UU^T = \mathbb{I}_{n\times n}$, and $\Sigma^2 = diag\,(\sigma_1^2, \ldots, \sigma_n^2)$. Let $h \in \mathbb{R}^{n\times 1}$ be the normal random vector $\sim \mathcal{N}_n\,[\mathbf{0}, \mathbb{I}_{n\times n}]$. Then for any $V \in \mathbb{R}^{n\times n}$, $V^TV = VV^T = \mathbb{I}_{n\times n}$, the random vector $g = U\Sigma V^T h$ is the normal random vector $\sim \mathcal{N}_n[\mathbf{0}, C]$. For i.i.d. $h_1, \ldots, h_r$, this experiment generates the first-order Gaussian matrix $G = [g_1, \ldots, g_r] \in \mathbb{R}^{n \times r}$, distributed as $\mathcal{N}_n [\mathbf{0}, \mathbb{I} \otimes C]$. The covariance, rank, and spectrum of G are controlled with U, Σ^2.

A second order random matrix is constructed using first order Gaussian matrices as $G_1 G_2^T$; with G_1 independent of G_2. *Low-rank* matrices of rank r used for simulations in [5] and [6] can be obtained by setting $U_1 = U_2 = \Sigma_1 = \Sigma_2 = \mathbb{I}_n$, $C = \mathbb{I}$ and drawing $g_i^{(1)}$, $g_i^{(2)}$ independantly.

A realization of a *Wishart* matrix W is obtained by taking the outer product GG^T.

To build random *Vandermonde* matrices, a standard *Vandermonde* matrix $Z = [z_{jk}]_{n \times n}; z_{jk} = \alpha_k^{j-1}; j = 1, \ldots, n; k = 1, \ldots, n$ and $\alpha_k = e^{\imath \frac{2\pi}{n}(k-1)}$ is constructed first. Then a first order random *Vandermonde* matrix V is built as

$$V = Z \Sigma G \quad . \tag{5.10}$$

The rank and the spectrum of V is controlled with Z and Σ. A realization of a second order random matrix is built as VV^T using a realization of a first order Vandermonde matrix.

Sparse Matrices. We employ Algorithm 1 to build a *sparse* matrix A with the desired *sparsity* and to uniformly scatter its support. It samples s points uniformly without replacement from n^2 locations. The values of active elements are set using a desired distribution for δ_A.

Algorithm 1 Build *sparse* A

```
$A \leftarrow 0_{n \times n}$
while $||A||_0 < s$ do
    $x \leftarrow$ discrete uniform random($n$)
    $y \leftarrow$ discrete uniform random($n$)
    if $A(x, y) == 0$ then
        $A(x, y) \leftarrow \delta_A$
    end if
end while
```

5.2.3.2. *Error Metrics.* Here we establish several error metrics for measuring the fidelity of a *low-rank*—plus—*sparse* matrix recovery.

Normalized Error Norm. We define the *normalized error norm* for recovery of the *low-rank* component as $\frac{||B-L||}{||B||}$. In particular, we use the *two* (*spectral*), *Frobenius*, *nuclear*, and *infinity* norms for the *low-rank* matrices. The *normalized error norm* for the *sparse* component is $\frac{||A-S||}{||S||}$. The norms used for the *sparse* matrices are *zero*, *one*, *two*, and *infinity.*

Alignment of Low-Rank Components. The *principal angles* and *geodesic distances* between the *column-spaces* and the *row-spaces* of B and L are used to assess the alignment of the decomposed *low-rank* component with the original. The *cosine* of the first principal angle between the column or row space of B and L is [117] $\cos\theta_1 = \max_k \sigma_k(Q_B^T Q_L)$, where Q_B and Q_L are orthogonal bases for the column or row spaces of B and L and σ_k is the k^{th} singular value. The geodesic distance [118] is $||\,\{\theta_k | k = 1, \cdots, r\}\,||_2$, where r is the minimum of $rank(B)$ and $rank(L)$, and θ_k is the k^{th} principal angle between the subspaces.

Support of *Sparse* Components. The accuracy of recovery of the *sparse* component is also assessed by comparing the *supports* of S and A. We define the detection rate D and false-alarm rate F as

$$\begin{aligned} D &= \frac{|supp(A) \cap supp(S)|}{|supp(A)|} \quad \text{and} \\ F &= \frac{|\overline{supp(A)} \cap supp(S)|}{|supp(S)|} \quad . \end{aligned}$$

5.2.3.3. *Reconstruction.* We use the implementation of *RPCA* in [108, 109] which is also employed in [107], to decompose Y of (5.1) into a *low-rank* matrix L and a *sparse* matrix S that approximate B and A. While the standard *RPCA* is noiseless, the ϵ*RPCA* algorithm [108, 109] solves the *RPCA* problem (5.2) with a point-wise error bound ϵ using *Augmented*

Lagrange Multipliers [119]. We set $\epsilon = \mathbf{0}$, $\lambda = \frac{1}{\sqrt{n}}$ and iterate to termination when $||Y - L - S||_F/||Y||_F < 10^{-8}$ or when more than 500 iterations have elapsed. Using the same threshold as in [5], recovery was considered successful when $||B - L||_F/||B||_F \leq 10^{-5}$. Further, matrix elements of the *sparse* components whose magnitudes are greater than 10^{-5} were considered active when support is calculated.

5.2.4. Experimental Results and Discussions. In Section 5.2.4.1 we find the ranges of *matrix sizes*, *ranks* and *sparsities* at which the published sufficient conditions are satisfied. Then in Section 5.2.4.2 we establish ranges for fractional-*rank* and fractional-*sparsity* where recovery is successful. These results indicate a very narrow range of *ranks* and *sparsities* where sufficient conditions are satisfied and a relatively larger region where recovery is successful. We use several error metrics in Section 5.2.4.3 to assess recovery. In Section 5.2.4.4 recoverability of real-world matrices are employed to demonstrate the accuracy of recovery regions established in Section 5.2.4.2.

When not otherwise noted, the *low-rank* and the *sparse* matrices are built as follows. The *low-rank* matrices are *rank-r Wishart* matrices with left singular vectors U drawn once from an $n \times n$, white *Gaussian* matrix and singular values

$$\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_n)\ ;\ \sigma_i = \begin{cases} \mathit{Unif}[0,1], & i \leq r \\ 0, & i > r \end{cases} . \tag{5.11}$$

The *sparse* matrices are built using Algorithm 1, with $\delta_A \sim \mathit{Unif}[-1,1]$. When appropriate, rank r is expressed in *fractional-rank* $\frac{r}{n}$ and sparsity s in *fractional-sparsity* $\frac{s}{n^2}$. Experiments are repeated 100 times for each configuration of low-rank and sparse matrices.

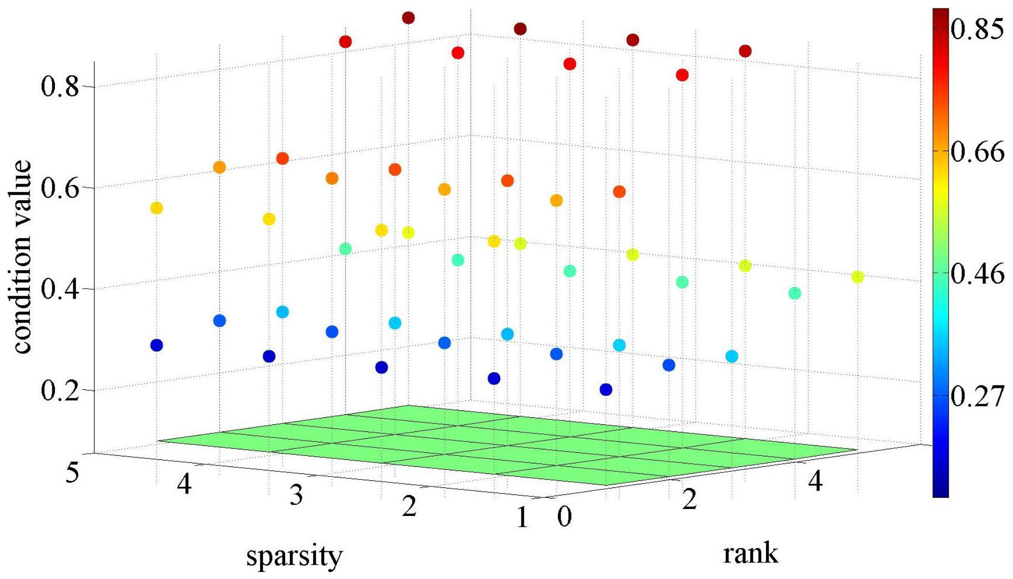
5.2.4.1. *Region Satisfying the Sufficient Conditions.* Here we seek answers to the following questions regarding the published sufficient conditions.

(1) At which *matrix sizes*, *ranks* and *sparsities* are published sufficient conditions satisfied?
(2) What impact does the matrix type have on recovery conditions?

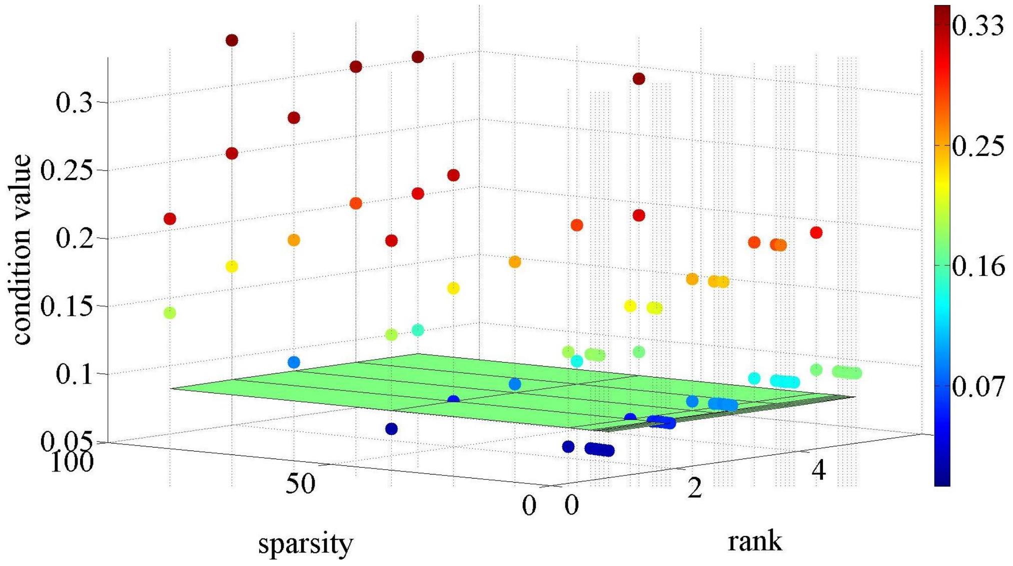
We begin by testing matrices for the probability of satisfying the sufficient condition (5.9). The matrices used here are inspired by the random orthogonal model in [6] and therefore produce similar *incoherence* and deg_{max} values. Fig. 5.1 displays the condition value of (5.9) for 100 realizations at each *rank*-*sparsity* pair for two matrix sizes, $n = 100$ and $n = 2000$. The surface in Fig. 5.1 indicates the condition (5.9) threshold $\frac{1}{12}$. Fig. 5.1 indicates that none of the realizations of 100×100 matrices satisfied the sufficient conditions, and a few of extremely low *ranks* and *sparsities* of 2000×2000 matrices satisfied the sufficient conditions.

Fig. 5.2 provides a summary of *ranks*, *sparsities* and *matrix sizes* that satisfy the sufficient conditions in over 90 out of 100 realizations. The results of Fig. 5.1 and Fig. 5.2 indicate that the sufficient conditions are more likely to be satisfied when the matrices are large, and the *ranks* and *sparsities* are very small.

These experiments indicate that sufficient conditions are satisfied for a very narrow region of *ranks* and *sparsities.* Fig. 5.3 shows the fraction of realizations of *Wishart* matrices that satisfied the sufficient conditions. Results corresponding to three matrix sizes $n = 2000, 2500, 5000$ are shown for *rank*-1, *rank*-2 , and *rank*-3 in Fig. 5.3. It can be noted that increasing the *rank* dramatically reduces the probability of satisfying the sufficient conditions. Further experiments show similar effects for other types of *low-rank* matrices.



(A) matrices of size 100×100



(B) matrices of size 2000×2000

FIGURE 5.1. Condition values of (5.9) for 100 realizations of each *rank-sparsity* pair. *Wishart* matrices are used for the *low-rank* component. The *support* of the *sparse* matrices is scattered *uniformly at random* and the magnitudes are distributed uniformly over $[-1, 1]$. The surface indicates the sufficient condition threshold $\frac{1}{12}$.

5.2.4.2. *Region of Successful Recovery.* We also note that all the realizations in Fig. 5.1 recovered successfully, though only a few satisfied the sufficient conditions. That is, recovery is successful over much wider ranges of *ranks* r, *sparsities* s and *matrix sizes* n than those

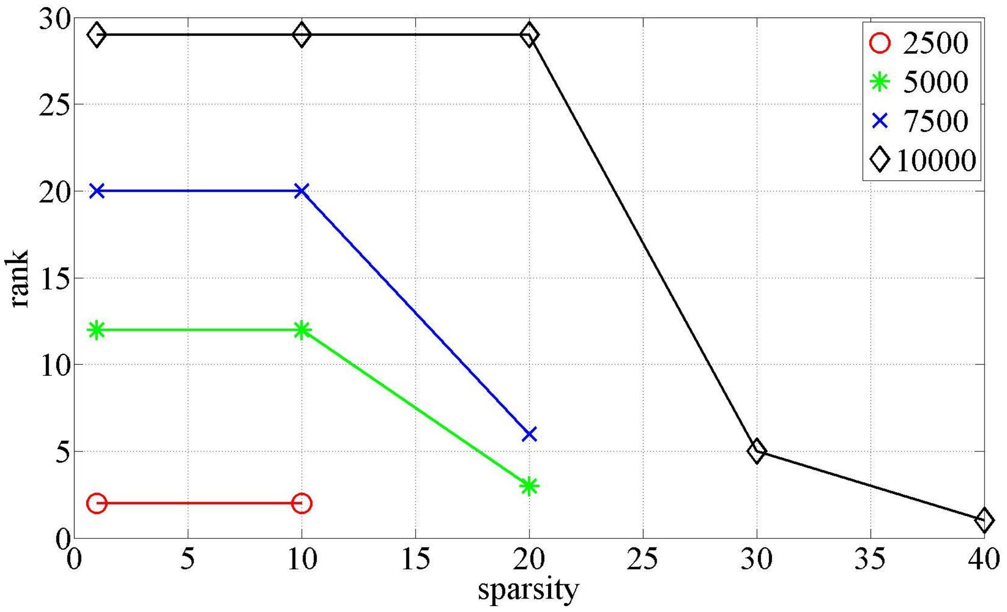


Figure 5.2. Highest *rank-sparsity* combinations at which at least 90 of 100 realizations satisfied the sufficient conditions, for different matrix sizes. *Wishart* matrices are used for the *low-rank* component. The *support* of the *sparse* matrices is scattered *uniformly at random*.

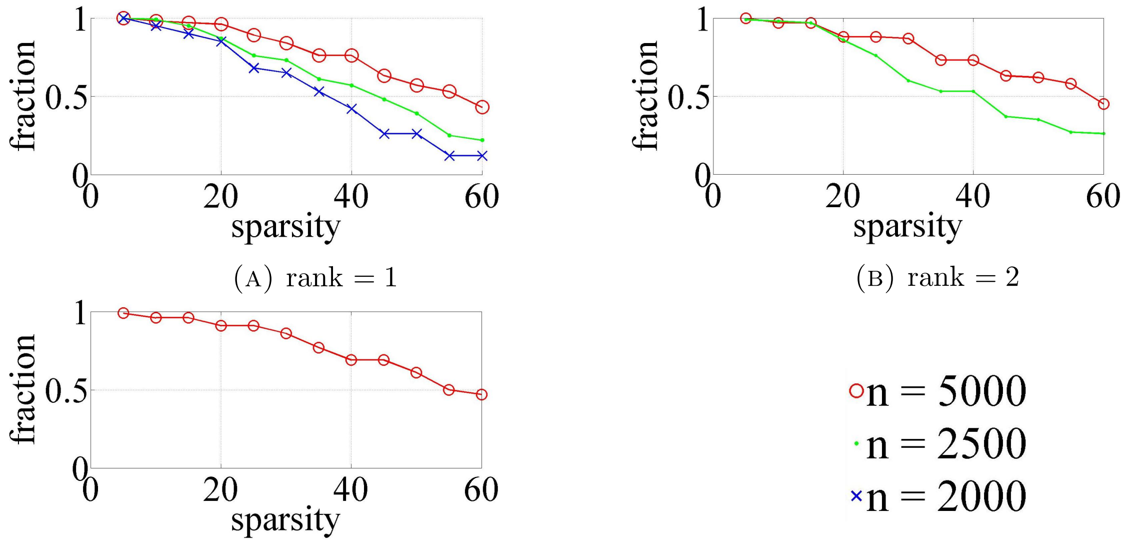


Figure 5.3. Fraction of the realizations that satisfied the sufficient condition for *Wishart* matrices with varying *size* $n \times n$ and *rank* r against *sparsity*. Curves are not shown when none of the realizations satisfied the sufficient conditions. The *support* of the *sparse* matrices is scattered *uniformly at random*.

might be suggested by the sufficient conditions. Therefore we investigate the empirical recovery regions of $RPCA$, as a function of rank and sparsity.

The first experiment on *recoverability* finds the boundaries of the recoverable *fractional-rank* $\frac{r}{n}$ and *fractional-sparsity* $\frac{s}{n^2}$ regions for different matrix sizes. These boundaries are shown in Fig. 5.4. As indicated in Fig. 5.2, *Wishart* matrices of size $n = 2500$ satisfy sufficient conditions with a probability of 90% at *rank*$= 2$ (*fractional-rank*$= 0.0008$) and *sparsity*$= 10$ (*fractional-sparsity*$= 1.6 \times 10^{-6}$). Any higher *ranks* or *sparsities*, or lower matrix sizes do not satisfy the sufficient conditions. But Fig. 5.4 indicates that *Wishart* matrices of size $n = 2000$ have 100% recoverability at (rank, sparsity) pairs far exceeding these values.

Practically recoverable boundaries are established with the pairs of highest *fractional-ranks* and *fractional-sparsities* for which 100 out of 100 realizations recovered successfully. Here we observe that boundaries of recoverable regions closely follow the relationship

$$\left(\frac{s}{n^2} + \alpha(n)\right)\left(\frac{r}{n} + \beta(n)\right) = \kappa(n) \tag{5.12}$$

for some α, β and κ. This model between *fractional-rank* and *fractional-sparsity* is qualitatively consistent with the inverse relationship between *maximum matrix degree* and *incoherence* in the sufficient conditions of [6].

Next we investigate the probability of successful recovery as *rank* and *sparsity* are varied in an experiment on 100×100 matrices. As shown in Fig. 5.5, at very low *ranks* and *sparsities* recovery is assured, but the probability decreases as the *rank* and the *sparsity* increase. Further, the narrow gap between the region of 100% recoverability and 0%, indicates a *phase change* from recoverable region to unrecoverable region in *fractional-rank-fractional-sparsity* plane.

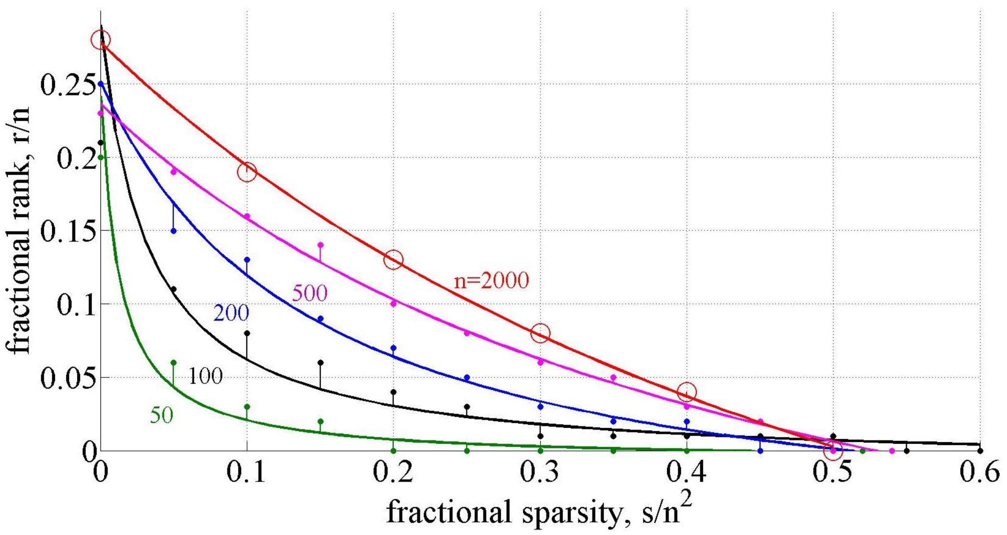


FIGURE 5.4. Boundaries of the 100% recoverable regions of different matrix sizes $n \times n$. Points indicates the largest rank that allows recovery at a given sparsity. Curves are fitted using (5.12). *Wishart* matrices are used for the *low-rank* component. The *support* of the *sparse* matrices is scattered *uniformly at random* and the magnitudes are distributed uniformly over $[-1, 1]$.

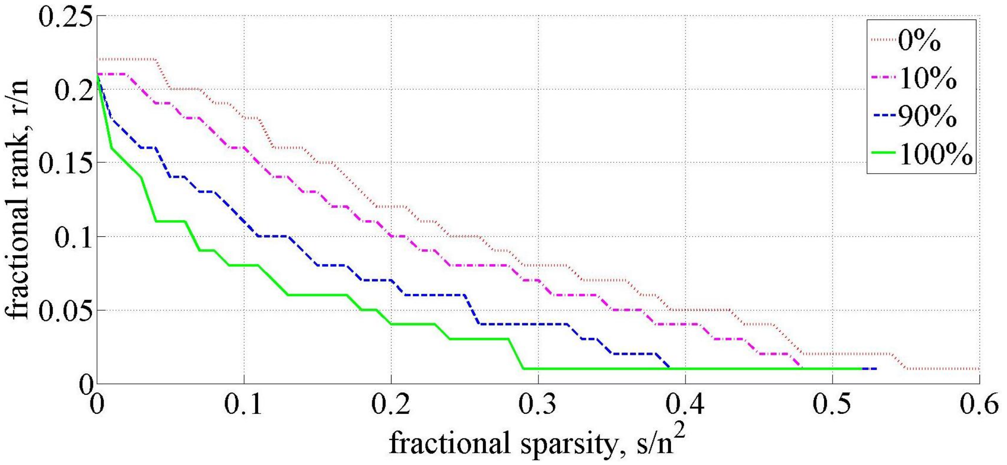


FIGURE 5.5. Recovery percentile contours on *fractional-rank—fractional-sparsity* plane. Matrices of size 100×100. *Wishart* matrices are used for the *low-rank* component. The *support* of the *sparse* matrices is scattered *uniformly at random* and the magnitudes are distributed uniformly over $[-1, 1]$.

The impact of the *type* of the *low-rank* matrix on recovery is studied next. We fit the model (5.12) onto the boundaries of the recoverable regions of different *low-rank* matrix types, as shown in Fig. 5.6. This indicates that the recovery region is relatively invariant to matrix type.

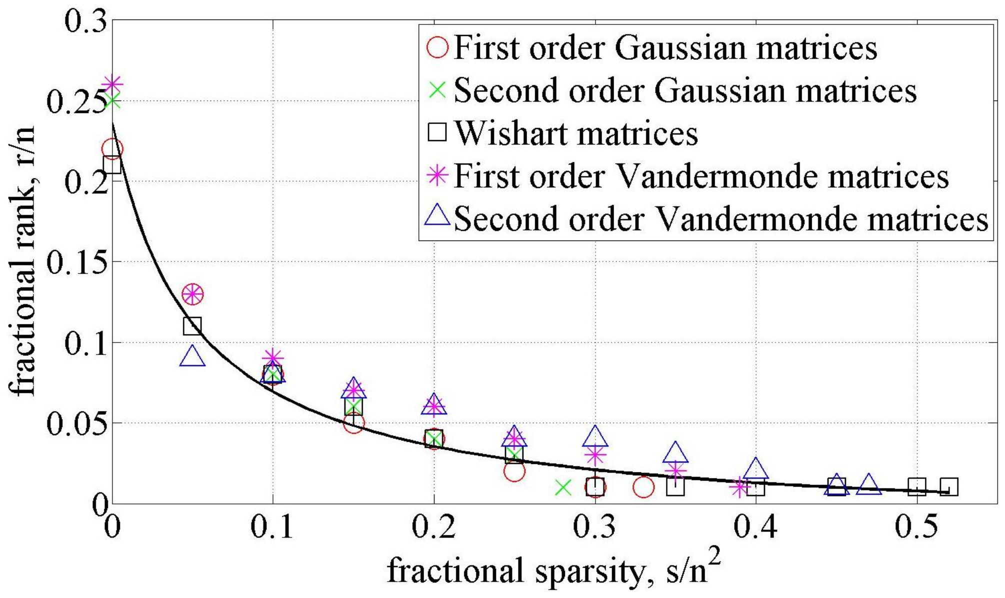


FIGURE 5.6. Boundaries of the 100% recoverable regions of different matrix types of sizes 100×100. Markers indicate the empirical boundary points and the solid line indicates the fitted curve. The fitted curve is computed for Wishart matrices. The *support* of the *sparse* matrices is scattered *uniformly at random* and the magnitudes are distributed uniformly over $[-1, 1]$.

Fig. 5.7 characterizes recovery as a function of fractional-rank and fractional-sparsity of the matrices that are passed to the RPCA algorithm (input matrices) and the matrices that are returned as recovered matrices. The figure shows a map of decompositions across the entire range of *fractional-ranks* and *fractional-sparsities.* The map is constructed using 100 realizations at each input matrix and showing the concentration ellipses for the corresponding output matrices. We observe that output matrix pairs corresponding to input matrix pairs that are of relatively *low-rank* and *low-sparsity*, map back to the inputs. Output pairs of input matrices of moderate to high rank and sparsities map to a near-*half-rank* and a near-*half-sparse* region. This indicates that at low *rank* and *sparsity*, where recovery is successful, *output* matrices are of low *rank* and *sparsity* (matching those of the *input*). At higher *input ranks* and *sparsities* where recovery fails, the *output* matrices are near half-*rank* and near half-*sparse*. Thus, we can characterize decompositions by the *rank* and *sparsity* of the output

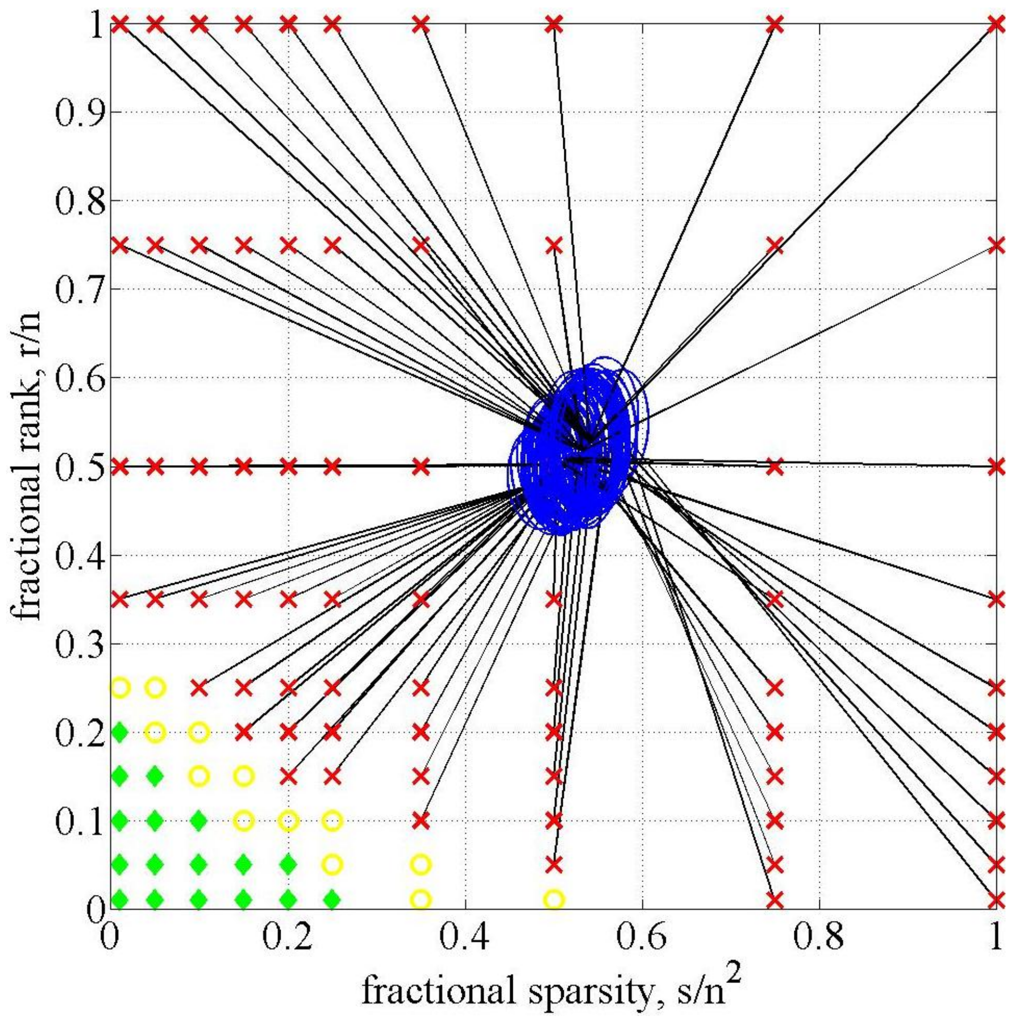


FIGURE 5.7. *Input* and *output fractional-rank—fractional-sparsity* combinations. Ellipses in blue indicate the *concentration ellipses* of the *output fractional-rank* and *fractional-sparsity* distributions corresponding to each *input fractional-rank* and *fractional-sparsity* combination. 100 realizations were generated at each *input* combination. Combinations corresponding to green diamond markers consistently recovered successfully. Combinations corresponding to red crosses consistently failed recovery. Recovery of the combinations corresponding to yellow circle markers were ambiguous. Each *input* combination that consistently failed recovery is linked to the corresponding average *output fractional-rank* and *fractional-sparsity*. Matrices are of size 100×100. *Wishart* matrices are used for the *low-rank* component. The *support* of the *sparse* matrices is scattered *uniformly at random* and the magnitudes are distributed uniformly over $[-1, 1]$.

matrices. This result suggests a *cross validation principle*. If the rank and sparsity of output matrices are near half-rank and near half-sparsity, the recovery has likely failed.

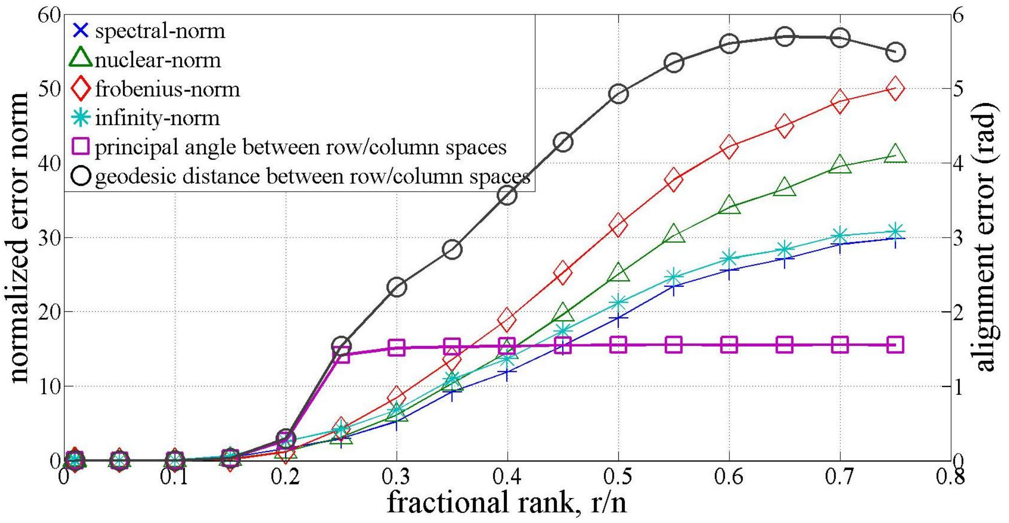


FIGURE 5.8. Recovery error of the *low-rank* component against *fractional-rank*. Normalized error norms are shown in thinner lines and read on the left axis. Alignment errors are in thicker lines and read on the right axis. Matrices are of size 100×100. *Fractional-sparsity* of the *sparse* component is fixed at 0.1. *Wishart* matrices are used for the *low-rank* component. The *support* of the *sparse* matrices is scattered *uniformly at random* and the magnitudes are distributed uniformly over $[-1, 1]$.

5.2.4.3. *Error in Recovery.* The definition of recovery depends on the application. Thus, we consider the common notions of errors discussed in Section 5.2.3.2. Figs. 5.8 and 5.9 indicate that recovery fails according to all error matrics at the same *fractional-rank* of the *low-rank* component and at the same *fractional-sparsity* of the *sparse* component. Thus, the results presented in this paper on successful recovery regions are consistent across error norms. An interesting phenomenon can be noted in Fig. 5.9. With increasing *fractional-sparsity*, the *false positive rate* shows a sudden rise and a decrease. This is due to the elements from the *low-rank* component leaking into the *sparse* component in malformed recoveries. As the *fractional-sparsity* of the original A increases, the support of the leakage overlaps with $supp(A)$ causing a decrease of the false negatives.

Sufficient conditions suggest that recovery only depends on the *support* of the *sparse* component, not on the distribution of the magnitudes of the active elements. The next

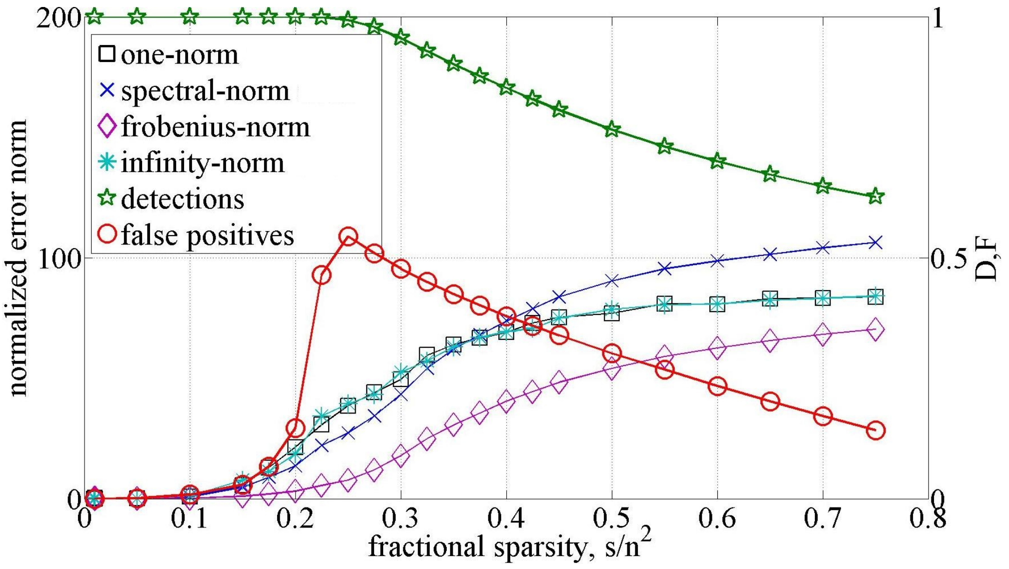


FIGURE 5.9. Recovery error of the *sparse* component against *fractional-sparsity.* Normalized error norms are shown in thinner lines and read on the left axis. Detections of non-zero elements are in thicker lines and read on the right axis. Matrices are of size 100×100. *Wishart* matrices are used for the *low-rank* component and their *fractional-rank* is fixed at 0.1. The *support* of the *sparse* matrices is scattered *uniformly at random* and the magnitudes are distributed uniformly over $[-1, 1]$.

experiment supports this claim. Here, we observe the effects of fixing and varying active element magnitudes δ_A of the *sparse* component. Fig. 5.10 shows the error in recovery of the *sparse* component for three distributions: fixed at 1, *unif*$[-1, 1]$, and $\mathcal{N}(0, 1)$. Results indicate that the value distribution of δ_A does not affect the recoverable region.

5.2.4.4. *Recovery of Real-World Low-Rank Matrices.* Finally, we seek to demonstrate the effectiveness of RPCA when applied to real-world matrices. For that, we investigate the maximum sparse contaminations from which a series of real-world low-rank matrices can recover. We further seek to predict the recoverability of these matrices using the recovery boundaries established earlier in Section 5.2.4.2 with synthetic matrices.

The real-world low-rank matrices used here are from the San Jose State University Singular Matrix Database [120]. We select a series of low-rank matrices that are of relatively

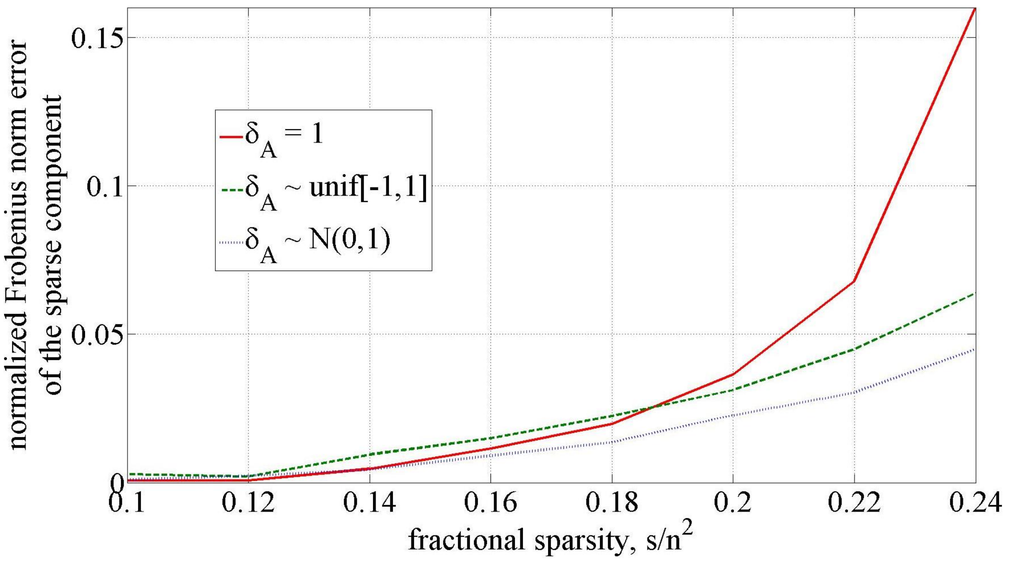


FIGURE 5.10. Recovery error of *sparse* matrices with varying *sparsity* for 100×100 matrices. The *support* of the *sparse* matrices is scattered *uniformly at random*. *Wishart* matrices are used for the *low-rank* component and they are fixed at rank=10.

low rank but not sparse. The selected matrices belong to the built-in test problems in Regularization Tools [121] that are used to analyze and solve discrete ill-posed problems. The matrix categories used here are:

- **baart** Used as noisy matrices to test solving Fredholm integral equation of the first kind [122].
- **shaw** Test matrices for one-dimensional image restoration model of image reconstruction problems of [123].
- **wing** Used for testing discretization of Fredholm integral problems with discontinuous solutions [124].

Then these matrices are contaminated with sparse matrices built using Algorithm 1, with $\delta_A \sim Unif[-1, 1]$. Table 5.1 lists the maximum fractional-sparsity of the contaminations that can be added and still successfully recover the low-rank matrices. Using the recovery boundaries built for *Wishart* matrices, the maximum fractional-sparsity of the tolerable

Table 5.1. Maximum Fractional Sparsity of Contamination from which Real *Low-Rank* Matrices Can Be Recovered

Group/Name/SJId	size	fractional-rank	actual maximum fractional sparsity	estimated maximum fractional sparsity	actual $\leq$ maximum
Regtools/baart_100/233	100×100	0.12	0.01	0.04	Yes
Regtools/baart_200/234	200×200	0.06	0.12	0.21	Yes
Regtools/baart_500/235	500×500	0.022	0.36	0.43	Yes
Regtools/baart_1000/236	1000×1000	0.013	0.48	0.48	Yes
Regtools/shaw_200/254	200×200	0.1	0.04	0.13	Yes
Regtools/shaw_500/255	500×500	0.04	0.3	0.37	Yes
Regtools/shaw_1000/256	1000×1000	0.02	0.44	0.46	Yes
Regtools/wing_200/262	200×200	0.04	0.07	0.27	Yes
Regtools/wing_500/263	500×500	0.016	0.28	0.46	Yes
Regtools/wing_1000/264	1000×1000	0.008	0.47	0.50	Yes

contamination is estimated and listed in Table 5.1. Notably, the actual maximum sparsities for recovery are close and below the estimated boundaries in all cases.

Next we consider the positioning of the estimated recovery boundary with respect to the actual recoverable region of a few real matrices. For that, we record the recoverability of three 200×200 matrices with varying contamination sparsities. Then we overlay the estimated recoverable boundary of 200×200 Wishart matrices. These results are shown in Fig. 5.11. As can be noted, the recovery boundary of Wishart matrices is a good estimate for the recovery boundary of the three real-world matrix types considered.

5.2.5. Conclusions.

The performance of *RPCA* in recovering a *low-rank* matrix and a *sparse* matrix from their sum is evaluated empirically. Our findings may be summarized as follows:

(1) Published sufficient conditions for recoverability are rarely satisfied even at low *ranks* and *sparsities*.

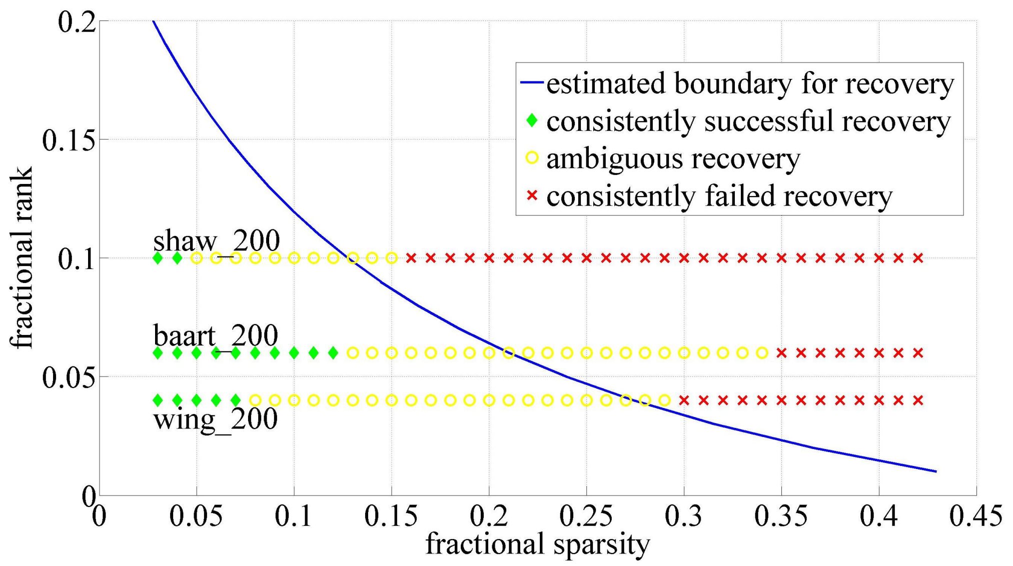


FIGURE 5.11. Comparison of recoverability of real-world matrices and the recovery boundary estimated using synthetic matrices. The matrices considered here are of 200×200. The real-world matrices are from SJSU Singular Matrix Database and the estimated boundary is based on Wishart matrices.

(2) Common random matrices do not satisfy the sufficient conditions, but recover at low *ranks* and/or low *sparsities*.

(3) At high *fractional -rank* and *-sparsity*, sufficient conditions are not met and recovery is unsuccessful.

(4) If the output matrices of a resolution are near-*half-rank* and near-*half-sparse*, the recovery has likely failed. This presents a cross validation principle for recovery.

(5) The gap between the region of *ranks* and *sparsities* where recovery consistently succeeds and the region where recovery consistently fails is narrow. This indicates a phase change of recoverability on the *fractional-rank—fractional-sparsity* plane.

(6) Experimentally established recovery regions are relatively invariant to the matrix type and the choice of the error metric.

(7) Empirical recovery regions established using synthetic matrices are reasonable estimators of recoverable regions of real-world matrices.

(8) As the sufficient conditions suggest, recovery only depends on the *support* of the *sparse* matrices, not on the distribution of the magnitudes.

The ranges of *rank* and *sparsity* for successful recovery suggest practical limits on what can be recovered with a practical *RPCA* algorithm.

5.3. Feature Filters for Network Data

Network data has a wide range of properties. Studying them and their correlations reveals interesting behaviors of communications. Further, extracting most of these features can be done cost effectively, as many network hardware support data filtering. In this section we present an organized approach to extract network data features and obtain derived features which we employ for network attack detection.

We assume an array of N network sensors each equipped with an array of (not necessarily similar) filters. These filters extract different features of network data. Then these features are fed into a computation unit, which calculate various derived features, such as cross correlations between different features. This structure is shown in Fig. 5.12. Our goal is to setup the feature filters and wire the computation unit in a way to reveal interesting information about the network data.

Here, we apply the proposed feature filter for network attack detection. We target five network attacks.

(1) **Neptune** attack opens a range of TCP connections and leave them hanging, preventing further legitimate requests establishing connections.
(2) **Smurf** attack overwhelm the victim by sending a large amount of echo-requests, thus, engaging the victim to a point it cannot execute its intended duties.
(3) **IP sweep** attack searches a sub-network for vulnerable hosts.

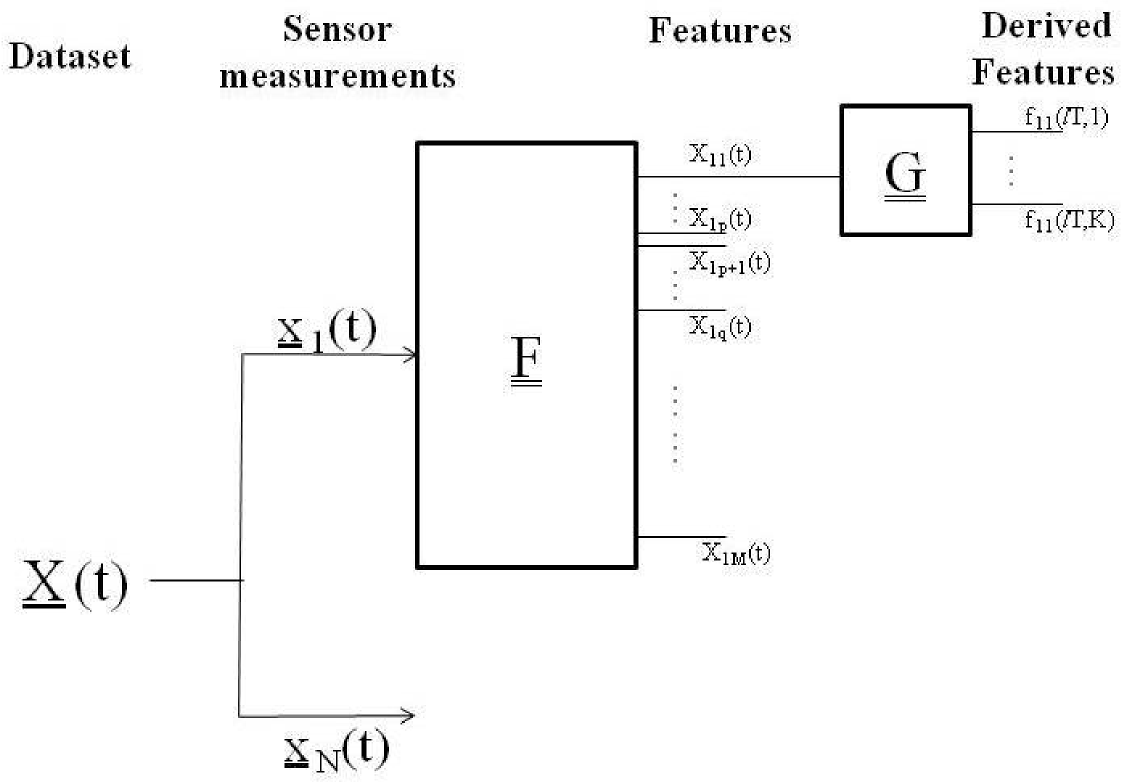


FIGURE 5.12. Feature Filter

(4) **Ping Of Death (pod)** attack sends a large echo-request packet which cause many systems to crash.

(5) **Port sweep** searches a victim for a vulnerable port.

The feature filter shown in Fig. 5.13 is designed to capture these five attacks. As can be noted only a few features and a limited calculations are needed to implement this detector. More specifically, the feature to be filtered are: protocol, packet type, destination IP, packet length and port. The computation unit require only a counter and time averaging units.

5.4. Detection of Subtle Variations in Analog Measurements - A Method for Counterfeit and Hardware Trojan Identification

A method to detect and localize subtle outliers in analog measurements is proposed, with applications to detection of hardware Trojans, counterfeits and other problems with low

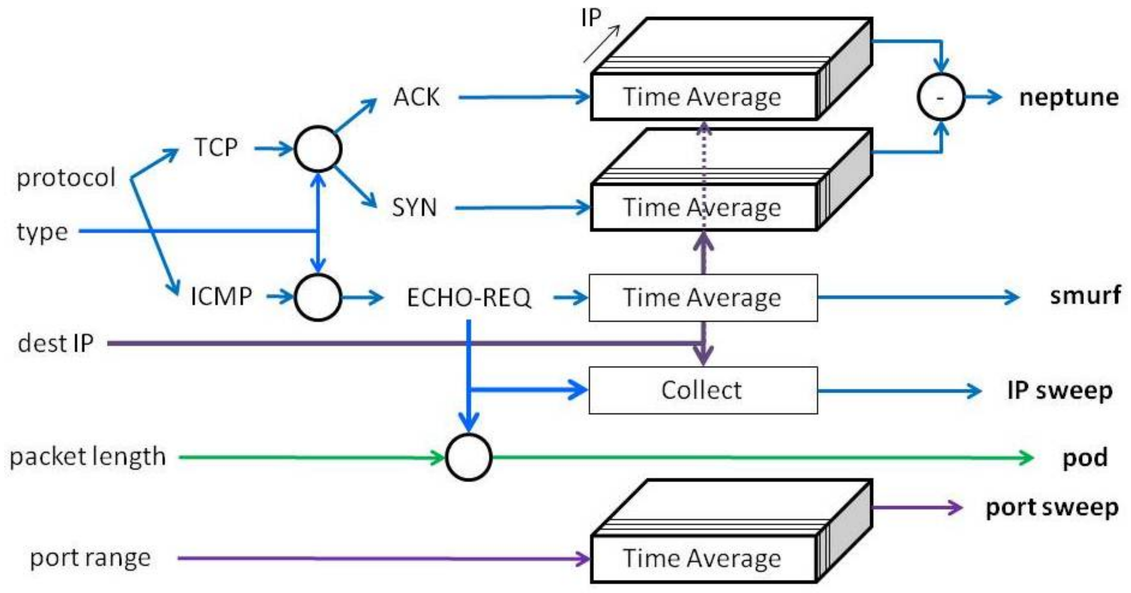


FIGURE 5.13. Detecting Various Network Attacks

SNR (signal-to-noise ratio) outcomes. Approach seeks disturbances from a trend established with reference samples. The method is tunable, takes global measurement correlations into account, and is sensitive and performs better than traditional statistical methods.

5.4.1. INTRODUCTION. Hardware trojans [125] and counterfeit components are becoming increasingly common and pose a real threat to physical and financial infrastructures. Since counterfeits functionally identical to legitimate devices and Trojans are inactive until the trigger sequence occurs, functional and digital testing scheme are not effective. Thus, techniques for detection often resort to analog measurements. However, “not obvious” counterfeits and well concealed Trojans cause only slight deviations even in analog measurements. Costly alternatives such as medical scanning, embedding and querying serial numbers have to be used in place of analog/digital testing in the fight against counterfeits [126]. Low SNR measurements are also becoming increasingly common in PCB fault detection as devices and features scale and speeds increase; the signals associated with faults are now buried within the manufacturing and test tolerances. Subtle outliers lying within such acceptable tolerances escape many statistical detections schemes as well.

Hardware Trojans for ICs may be inserted by modifying the design specification or by modifying fabrication. Countermeasures such as circuit obfuscation [127] lead to many challenges including testing and detection difficult. Advanced techniques such as side-channel signal testing [128] are emerging, but they are only effective in detecting large Trojans. Side-Channel Analysis (SCA) [129] operates by applying a various test sequences and measuring side-effects such as magnetic and electric fields. While SCA is used to detect hardware abnormalities such as Trojans, Trojans may use SCA to couple themselves to the circuit without a physical connection [130] hiding from test signals. Hardware Trojans are not restricted to modified ICs, but they may sit at port interfaces, connectors or externally but in close proximity to modules seeking side-channel signals [131]. Except for large and less sophisticated ones, Trojans produce only subtle deviations within the acceptable range of measurements, making detecting challenging with traditional means.

In a typical detection setup, a set of reference products/devices are used to establish the acceptable ranges for different measurements. This range depends on the variability of the Device Under Test (DUT) due to parametric and manufacturing variations and the variability of the measurement systems. In PCB testing, reference set would be readings at each pin of a set of functional boards, obtained perhaps even from different test fixtures or testers. In statistical detection, this set provides a characterization of acceptability in terms of statistics such as the mean and standard deviation. Many statistical techniques treat values corresponding to different test pins independently. [Note: Without loss of generality we use the term value of a pin to refer to one point measurement out of a set of such measurements, a term associated with PCBs. However, the description is applicable even when each measurement corresponds to a different value, e.g., a delay measurement or a current measurement.] What lacks here is a notion of a trend that takes into account the correlations among measurements and direction of variations from one measurement from one vector (pin) to the other. Although all measurements lie within the acceptable range of values for each vector (pin), if it does not follow the appropriate trend its acceptability is questionable. This issue becomes severe for high variance datasets, such as in low SNR situations. Even for perfectly operational reference set, measurements may vary over a significant range due to parameter variations with process margins, inherent thermal noise and variations in fixtures and testers. Such high variances bury subtle but critical deviations in measurement trends. A trend in general is associated with a particular ordering of pins, but our aim is not to depend on a particular ordering, rather to have the detection method consider multiple (or all) possible orderings in detecting devices going against the trend. Such an approach is essential for dealing with sophisticated circuits, e.g., pin-grid arrays and circuit boards in which many physical coordinate relationships exist among different pins, and circuits connected to seemingly unrelated pins. Results from approaches such those based on principle component analysis [132] are independent of the pin order.

We propose a scheme for detecting subtle outliers that go against the trends exhibited by reference devices even when each of the measurements individually lie within the acceptable range. A formulation is developed that highlights disturbances to the trends in measurements, thereby not only allowing for detection of unacceptable devices, but also localizes the suspect pins. Below we outline the formulation and underlying mathematics for the method. Results from PCB testing are provided to validate the method.

5.4.2. Methodology. The methodology utilizes measurements from a "good" set of devices - the reference set. We desire to test other devices for anomalous patterns or defects. The good set defines the baseline acceptable behavior, on top of which deviations are permitted within a tolerable margin.

First we consider the measurements obtained for a good set of devices. Denote the measurement trace for such a device by a length n row vector z_g ($\in \mathbb{R}^{1\times n}$). This measurement trace for a device is the resultant of many components such as the nominal design values, the parametric and manufacturing variations and the variations introduced by test fixtures. We assume the effect of these different components to be additive. Let p_i's be the row vectors of $\mathbb{R}^{1\times n}$ representing the individual contribution of these properties. Therefore $z_g = a_1p_1 + a_2p_2 + \ldots$ where a_i's are weights on vectors associated with different properties for the particular device under test. z_g is thus, given by $a_g = aP$, where P is the matrix containing p_i as the i^{th} row. However, P is unknown to us. As measurements are dependent on P, we can use measurements from a set of reference boards, denoted by B, to characterize P. i^{th} row of B is the vector of measurements of the i^{th} reference board, which is of the form $z_g = aP$. Therefore $B = AP$, where A contains a's of each reference board and thus, $P = A^+B$, where A^+ is the pseudo-inverse of A. Since $z_g = aP$ we may write:

$$z_g = aA^+B = \nu B \tag{5.13}$$

Let the number of reference devices be k. As each device provides n measurements, the reference set provides the baseline matrix B ($\in \mathbb{R}^{k\times n}$).

The outliers are detected by comparing the measurement vector z of device under test against the reference model described above. As per our linear model we express the relation between the B and z as:

$$z = \nu B + \epsilon \tag{5.14}$$

where νB correspond to the linear reference model and ϵ ($\in \mathbb{R}^{1\times n}$) is the row vector of deviations from the linear model. Information in z that does not fit the reference model such as noise, nonlinear components and abnormal measurement components of outliers are all represented by ϵ. Especially the measurement components of outliers that deviate from the reference model appear in ϵ. We take the value of ϵ to be the minimum vector value that satisfies (5.14), i.e., we minimize ϵ, over ν, by posing the following optimization problem:

$$\min . \|\epsilon\| = \|z - \nu B\| \tag{5.15}$$

L-1 norm is used for $\|\cdot\|$ owing to its sparsity promoting properties, as discussed at the end of this section. Apart from the norm $\|\cdot\|$, the baseline matrix B and constraints for the optimization problem are design parameters. These are chosen based on the nature of dataset and also due convenience. In the presented results, measurements from a carefully selected set of boards are used as the baseline B, as mentioned above L-1 norm and ν is bounded above zero and below one: $0 \leq \nu \leq 1$.

An important case is when ϵ consists only of a few significant values. Such an ϵ implies that the test trace deviates from the references significantly at a few locations, and these locations and the deviations are indicated in ϵ. In other words, faults get localized to a few measurement indices. Small Trojans are likely to be wired to a few critical data lines, and therefore appear on only a few measurement points. Similarly faults in PCBs that are difficult to be detected, may affect only a few pins. A highly sparse ϵ is yielded by forcing a minimum number of non-zero elements referred as the L-0 norm. But mathematical intractability of

L-0 norm prevents employing efficient solvers in this problem. It is known that tractable L-1 norm - the sum of absolute values, can be used in place of L-0 norm [133]. Therefore by solving (5.15) for a minimum ϵ in an L-1 sense, we can detect faulty devices and localize the fault to a minimum number of pins. A number of more mathematical implications construct the foundation of this work, such as low-rank-ness for de-noising, affine constraint for restricted fitting.

For the results presented here we selected the reference set by validating with multiple tests and finally manually selecting boards with similar measurement structure. However this tedious process is not a necessity. For situations where the reference set is possibly contaminated, we demonstrate two de-noising methods based on (1) Singular Value Decomposition (SVD) [134] and (2) Robust Principal Component Analysis (RPCA) [5].

5.4.3. Results. The section will present two sets of results. First is a set of PCB measurement data using Capacitive Leadframe Testing (aka TestJet® in industry). The Trojan sitting on the interposer layer (Fig. 5.14) samples the communication between the upper and lower dies looking for the trigger sequence. This will not affect regular communication between the two layers. Also the presence of a Hardware Trojan changes test measurements only mildly.

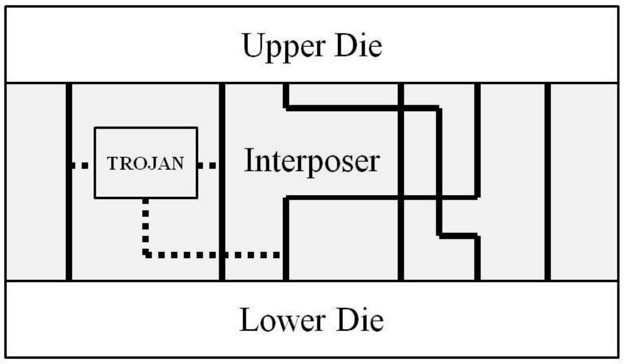
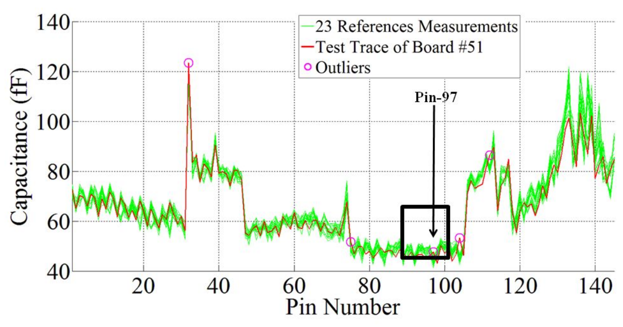


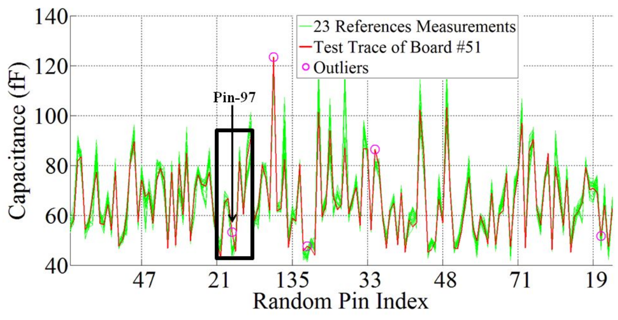
Figure 5.14. An example of a Hardware Trojan in a connector

Next we present a subset of results from the PCB testing experiment for a j24 connector mounted on a PCB tested with an Agilent 3070 tester running VTEP. Each board has 145 test pins that are not grounded or V_{DD}, for which the tester measured the capacitance between a sense plate above the boards and pin underneath the board [135, 132]. A good set of 23 boards was manually selected as the baseline test set B. 61 other board runs, for a total of 83 board runs, are made and each run is tested for abnormalities individually using the formulation in (3). Here, results are shown only for board #51. Figure 5.15 shows the subtle anomalies of board #51 localized to five pins. Figure 5.15a plots the measured values vs. the pin number, and in Fig. 5.15b we have shuffled the pin numbers to establish the invariance of the method to the order of the pins, i.e., such subtle violation of the trend occurs with respect to many other pins of the connector. Next we focus on the pin 97 (indicated by the box) in Figs. 5.15a and 5.15b. The same fault is marked in both pin orderings and the zoomed versions are shown in Fig. 5.16. The subtle fault is marked with a circle in the two sub-figures 5.16a and 5.16b. As can be seen, this subtle fault lies well within the acceptable measurement range. This subtle fault would easily escape statistical testing. However it does not follow the trend. The proposed method was capable of observing its disturbance to the nominal trend and detecting it. This establishes the key claim of this work.

Additionally we demonstrate the use of RPCA for de-noising a contaminated reference set, i.e., obtaining a good reference matrix B as discussed in Section 5.4.2 under selecting B. This is useful, for example when a reference set cannot be obtained via manual inspection. The measurements corresponding to the initial set of reference boards are shown in Fig. 5.17a and its de-noised version is shown in Fig. 5.17b. The final version will include a detailed description of this procedure and also results for validating the approach.

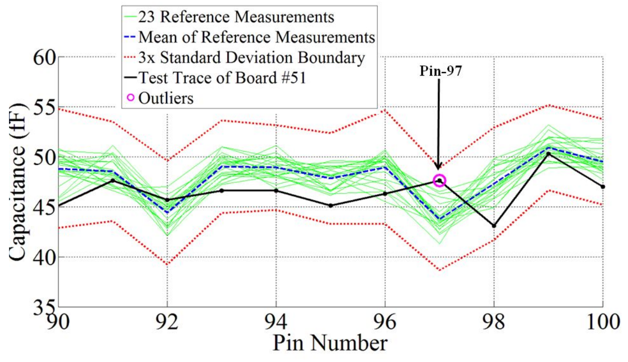


(A) pin number

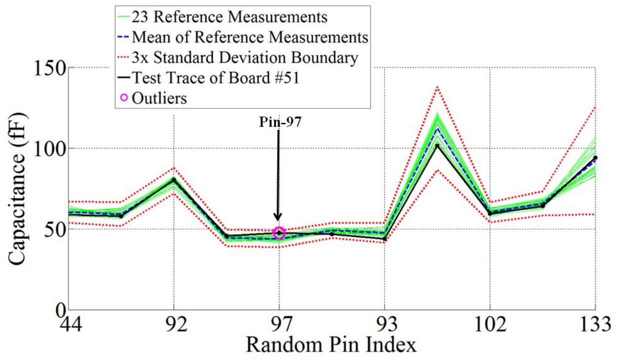


(B) randomly assigned pin

Figure 5.15. Capacitance values of different pins for 23 reference boards and test board #51 (a) vs. pin number, and (b) vs. a randomly assigned pin index.



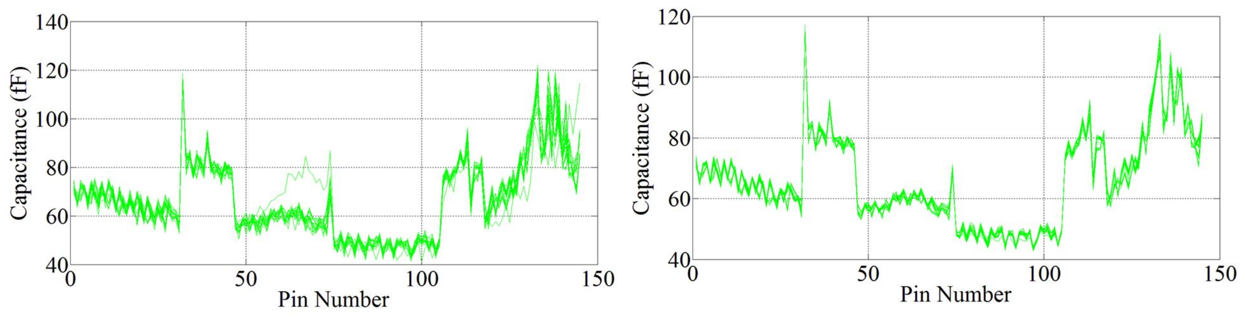
(A) pin number



(B) randomly assigned pin

Figure 5.16. Neighborhood of pin-97 of board #51 (a) a subtle outlier with respect to its neighbors in the original pin ordering (b) the same outlier with respect to a random set of pins.

5.4.4. Conclusions. A scheme to detect outliers causing subtle changes to in the measurement, but goes against the trend of a reference set of devices is presented. The method captures global trends over the entire set of measurements, and is insensitive to the order in which measurements are listed or even the physical order of the pins of a module under test. The method reveals subtle outliers such as those occur in presence of hardware Trojans and counterfeits even when they lie within the range of regular data variations. The insensitivity



(A) Original set of boards (B) RPCA selected boards

FIGURE 5.17. Obtaining a reference set of measurements for B (a) original set of measurement vectors, and (b) set of reference vectors obtained using RPCA approach.

to the pin ordering, relaxes the need of having measurements sorted in a particular order, which reduces the complexity. Additionally the final version will also present a sensitivity and robustness analysis of the proposed method.

5.5. Conclusions

A few feature extraction methods are studied in this section. Empirical recovery regions of RPCA are established along with cross validation principle to determine a successful recovery. Further, the findings are tested on a variety of matrices that include real-world matrices. A scheme to extract and derive meaningful features from network data is devised. This scheme is applied to detect a few network attacks. A methodology to detect subtle patterns is developed and tested on real PCB data.

CHAPTER 6

Network Traffic Anomalies

6.1. Introduction

We study network traffic anomalies in this chapter. Anomalies are sporadic and thus, are difficult to be modelled by classical approaches. Thus, we investigate a modularized approach to separately model different anomaly behaviors and combine them to describe the overall anomaly behavior. The resultant model concisely capture the anomaly behaviors of the network. The parameters of the models can be calculated with local measurements. However, they capture the global behavior of the anomalies. Using the concise description, two real-time applications are proposed. The presented work also includes anomaly extraction methods. The work is applied on to Internet2 throughput data to characterize its anomaly behavior. These findings help provision future networks and also develop more realistic traffic generators.

6.2. Modeling Spatial and Temporal Behavior of Internet Traffic Anomalies

A new approach based on graph wavelets for analyzing the spatial and temporal behavior of Internet traffic anomalies is presented. This approach is applied to Internet2 traffic measurements to evaluate the time duration and spatial spread (number of links affected) of anomalies. Based on the empirical results, a node model is proposed that captures the behavior of anomalies at individual network nodes. The model considers various aspects of anomalies, such as its origin, termination, propagation, duration and volume changes. The derivation of the model parameters requires only local node information, but the model is capable of producing network-wide anomalies whose behavior mimics network wide anomalies. Model is verified by using Internet2 traffic data. Since the proposed model can be specified using only a few parameters, it can be used in place of large anomaly traces with a great data reduction. As extensions, the model is applied over a path and an aggregated model that applies to a neighborhood in the network is also presented. A method to use the graph wavelet components found during the analysis to implement a real-time anomaly monitoring system is also discussed.

6.2.1. Introduction. Identifying Internet traffic anomalies, such as flash crowds and denial of service attacks, along with their spatial and temporal characteristics (e.g., life time and spatial spread) is vital for robust network design and operation. These characterizations provide critical information for designing and updating link, buffer and router capacities that are necessary for stable operation. They can also be used to identify the vulnerable regions of the network and to plan for adversarial attempts. Understanding the behavior of traffic anomalies also helps improve QoS provisioning and performance modeling. Modeling anomaly properties directly contributes for studies and also enables higher level representations. The model presented in this section captures the behavior of anomalies at a node. As shown later in the section, the model is also extended to aggregate a region of the network. Such aggregations capture traffic behavior between ISP level regions.

Recognizing a deviation from the usual traffic pattern is the main goal in anomaly detection and analysis. A thresholding technique in which traffic/flow rate is compared with a threshold would be an obvious choice, but due to the bursty and self-similar nature of Internet traffic [3], and daily and weekly variations, such approaches are not adequate. Frequency domain solutions in which anomalies are observed and characterized at certain frequency bands have shown some promise [136]. Wavelets decomposition and time-series analysis techniques have been used in [137] to study the statistical characteristics of network traffic anomalies.

Fourier analysis is used in [138] to find the source of an anomaly (origin-destination), as well as to characterize regular trends. A simple but effective method to detect and diagnose "black-holes," where packets are dropped in large quantities due to faults, is presented in [139]. Other anomaly detection and tracking techniques using principal component analysis [140, 138], machine learning [141], data mining [142], statistical analysis of payloads (for intrusion detection) [143], and risk modeling and Bayesian analysis (for IP fault tracking) [144, 145] have also been proposed. The reader is referred to [146] for a more comprehensive literature survey. In addition, a number of active probing techniques have been reported [29] for detecting and localizing anomalies and their spatial spread over the network, by comparing probe measurements against service level agreements. Probing techniques have also been proposed for fault diagnosis [147].

In [148], a relatively low complexity spatial analysis of network traffic is presented, which can be implemented at individual routers across the network to alleviate the need for monitoring network-wide data at a central location. Graph wavelets [149] provide a new way for spatial traffic analysis at different granularities. The idea behind graph wavelets is similar to that of standard discrete wavelet transform, where wavelet coefficients at different scales are obtained by aggregating or differencing adjacent data points in time, with appropriate weights depending on the type of wavelet. In graph wavelets, adjacent points correspond to neighborhoods defined on the network graph. For example a neighborhood can be a collection of nodes that are within a radius of a node of interest on the graph, or they can be a collection of links that are all connected to the same node. The size of the neighborhood in turn represents fine or coarse scales for multi-scale analysis. The former definition for neighborhoods is used in [149] for spatial traffic analysis, where the graph corresponds to the line graph of a network. The graph wavelet approach of [149] has shown great promise for discovering traffic patterns at different spatial granularities and for extracting low-dimensional representations of the traffic data.

Data adaptive approaches include diffusion wavelets [150], which have been used in [151, 12] for dimension reduction for network traffic analysis. Here, we propose a new approach for analyzing and modeling Internet traffic anomalies. We use graph wavelet analysis to evaluate the contribution of each link at a node, as opposed to [149] where graph wavelet analysis was done over an expanding neighborhood of the network. In [149], links at a certain depth of the neighborhood are aggregated, so the contribution of an individual link becomes less significant. Analyzing links at each node provides an analysis which does not loose granularity. Our aim is to analyze the temporal spread (lifetime) and the spatial spread (spread path and extent) of traffic anomalies across the network, and to develop simple models that capture such behavior at different spatio-temporal scales for network traffic modeling. Our main contributions are as follows:

(1) We first develop a multi-scale traffic analysis framework to analyze how traffic anomalies migrate through the network. Using graph-wavelet coefficients of the traffic data we determine how long an anomaly persist in the network, which route it takes, and how deep it spreads through the network. We show that this behavior can be adequately described using well-known statistical distributions.

(2) We then develop a model that captures the input-output relationship between anomaly traffic at a node. We extend this model to capture the overall (composite) input-output relation of the anomaly traffic for a set of nodes in the network. We also propose a real-time distributed detection system for traffic anomalies.

In Section 6.2.2, the traffic measurements and data preprocessing steps are discussed. Section 6.2.3 presents the proposed graph-wavelet based technique for tracing anomalies. Results explaining the spatial and temporal behavior of anomalies are presented in Section 6.2.4. A model for anomaly generation and propagation at nodes is presented in Section 6.2.5. Anomaly characteristics observed on Internet2 are presented in Section 6.2.6 in terms of model parameters. Application of the model is discussed in Section 6.2.7, and conclusions are drawn in Section 6.2.8.

6.2.2. Dataset and Preprocessing. We use traffic volume measurements from Internet2 network starting from Oct. 16th, 2005 [152]. The corresponding network topology is shown in Fig. 6.1. The data were collected at 11 nodes probing with 5 minute intervals. There were 28 inbound and outbound links spanning the network. However the techniques applied are very general and thus, can be used to analyze measurements from any network.

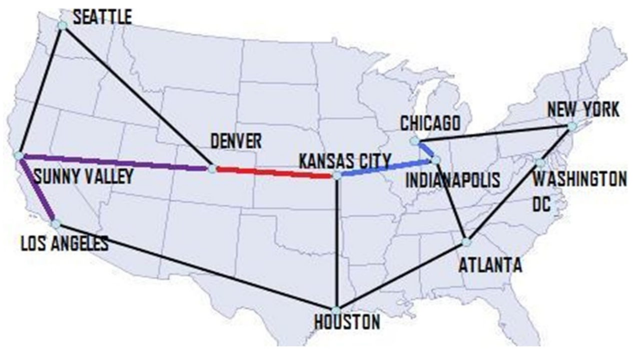


Figure 6.1. Network nodes and links from Internet2

Internet traffic is best characterized as bursty and self-similar [3]. Nonetheless, daily and weekly slow-varying patterns are present in traffic measures [136] as evident in Fig. 6.2. Before applying a threshold to detect anomalies, data has to be preprocessed to remove these trends as otherwise such trends could mask an anomaly. For example the measure of anomaly (A) in Fig. 6.2 is less than most of the peaks in the trace. Therefore the threshold is applied after de-trending the trace.

As probing is performed at 5 minute intervals, 2016 samples per link are collected over a complete week at every node. The common trend of traffic is identified by recognizing the prominent frequency components, by performing 2048 point FFT. In the Internet2 dataset, 20 frequency components sufficiently captured the common trends in a week. De-trending is done by zero-forcing these frequency components.

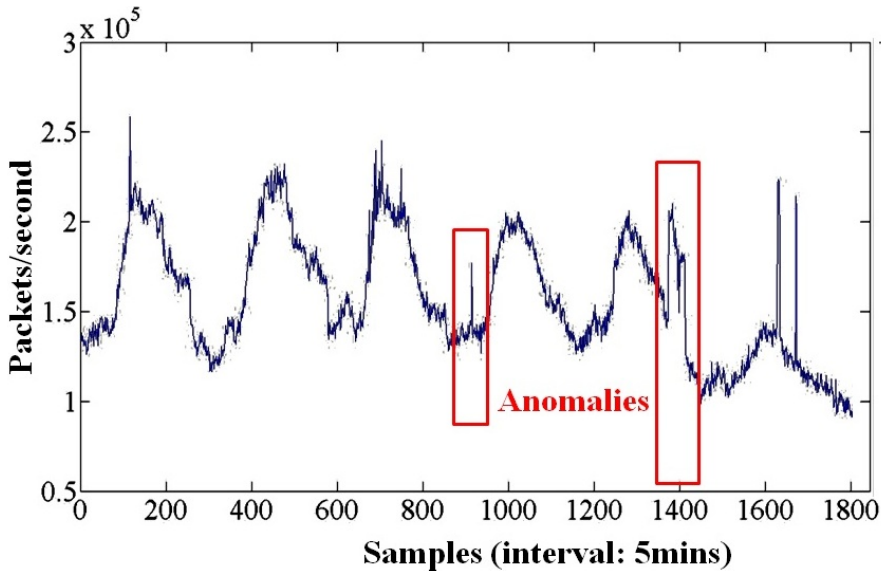


FIGURE 6.2. A sample weekly traffic trace

A simple thresholding is then performed to identify the large deviations (three times the standard deviation) back in the time domain. We consider these large deviations as traffic anomalies and we are interested to understand the spatial and temporal behavior of them.

Fig. 6.3 shows an example of the preprocessed data: Large deviations in the raw data (blue) are preserved in both the de-trended data (cyan) and the thresholded (red) data. But the traffic trends present in the observed data are significantly suppressed in the de-trended data. Thresholding has revealed the anomalies on the dataset.

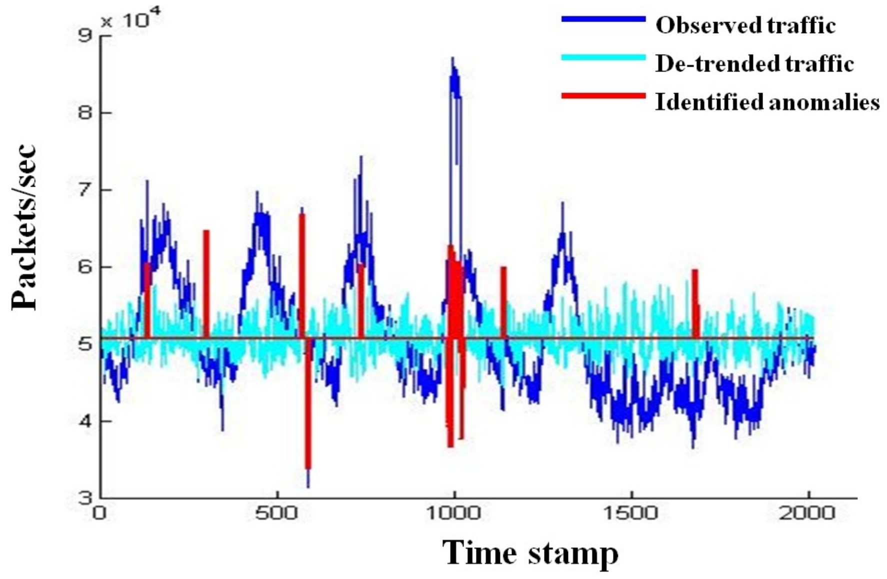


Figure 6.3. Detected anomalies (marked in red)

6.2.3. Graph Wavelets for Tracing Anomalies. Our aim is to analyze the spatial and temporal behavior of the anomalies across the network. Specifically, we wish to analyze the temporal spread (lifetime) and the spatial spread (spread path and extent) of the anomalies to understand how an anomaly affects the networks operation.

6.2.3.1. *Graph Wavelets.* To explain the basic idea behind graph-based wavelets, let us first recall the simplest form of *Discrete Wavelet Transform* (DWT), i.e., the *Haar* wavelet [153]. The Haar multi-scale representation of the data is obtained by forming differences between aggregated versions of the data at different scales. For instance, the Haar wavelet coefficients at the first scale are simply the differences between adjacent data points. The

TABLE 6.1. Properties Revealed by Graph Wavelet Coefficients

Node degree	Coefficient		
	Scale-1	Scale-2	Scale-3
2	Originating/ Propagating anomalies	N/A	N/A
3	Originating anomalies	Propagating anomalies	N/A
4	Originating anomalies	Propagating anomalies	Remaining contribution

coefficients at higher scales are obtained by computing differences between aggregated data points in neighboring dyadic intervals.

In this section, we apply Graph-based wavelets at each node over the links. This is different from the approach in [149], where analysis was done over the radius of neighborhood. Analysis on each node captures all the activities at each of them. Further, the wavelets coefficients reveal certain physical properties which depend of the degree of the node, as shown in Table 6.1. These properties are explained later in Section 6.2.5.

Referring to Fig. 6.4, suppose an anomaly is detected in L_0. We wish to analyze how incoming traffic from other links contribute to the anomaly effect in L_0, or how the anomaly in L_0 affects the outgoing traffic in other links connected to the node. We take a multi-scale approach that is, we study the anomaly propagation at various scales, with 1 corresponding to the finest scale.

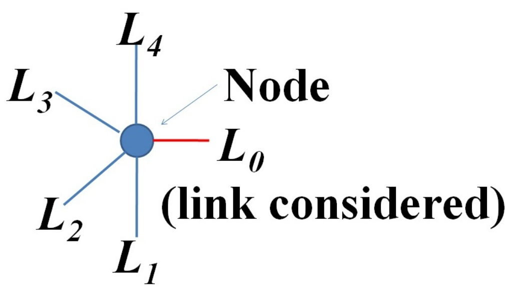


FIGURE 6.4. Multi-scale analysis of links around a node

At scale 1, we look at weighted differences between the traffic data at L_0 and the traffic data at each of the other links one by one. These differences are viewed as graph-wavelet coefficients at scale 1. In the example depicted in Fig. 6.4, there are four scale-1 coefficient time series, indexed from 1 to 4 depending on their location with respect to L_0 in a clockwise fashion. The scale-1 analysis compares the link under consideration with the other links individually and captures the difference (actually similarity) between the links. If the difference between L_0 and another link is small (magnitude-wise), we conclude that the anomaly must have propagated from there. On the other hand, if the difference is large then anomaly must have propagated from other links.

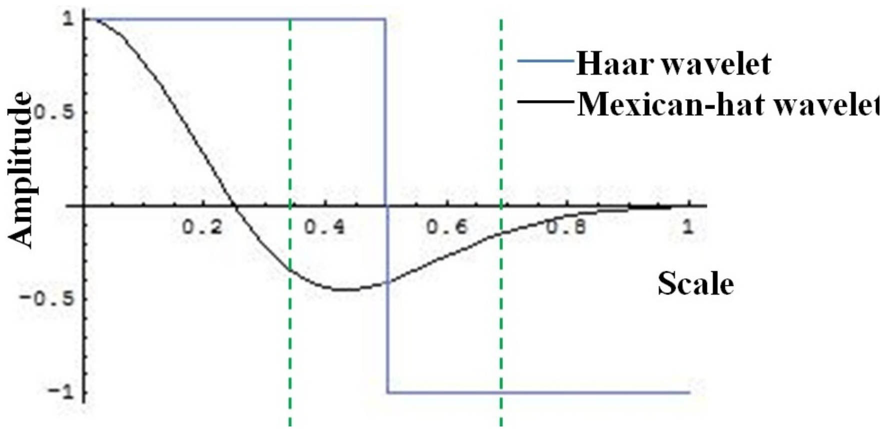


FIGURE 6.5. Partitioning of the wavelet function for weight calculation

The scale 2 analysis is coarser, and does not identify how every individual link is similar or dissimilar to L_0. Rather it explains how the aggregated traffic at a pair of nodes contributes to or is affected by the anomaly at L_0. Letting L_i denote the traffic in link i, at scale 2 we form differences (possibly with weights) of the form $L_0-(L_i+L_j)$, for all $i, j = 1, 2, \ldots, n-1$ and $i \neq j$. Here n is the total number of nodes.

At scale 3, the aggregated traffic at a collections of three links is compared with the traffic in L_0 that is we look at link differences of the form $L_0-(L_i+L_j+L_k)$, where (i, j, k)

TABLE 6.2. Weights for Scale 2 Analysis for the Haar and the Mexican-Hat Wavelets

	Haar wavelet	Mexican-hat wavelet
scale-0	+1	0.548
scale-1	0	-0.472
scale-2	-1	-0.075

enumerate all distinct link triples. Higher scale analyses, up to $n-1$, are defined in a similar fashion. Under the Haar wavelet an N-scale analysis will have the form:

$$\hat{L}_N = L_0 - \sum_{\forall \text{size} N \text{link sets}} L_i \tag{6.1}$$

6.2.3.2. *Coefficient Assignment.* The values of the weights are determined by the choice of the wavelet function. To determine the weights we first partition the wavelet function in time into n intervals of equal duration and then calculate the area under each interval. An example of this partitioning is shown in Fig. 6.5 for the *Haar* and *Mexican Hat* wavelets and the corresponding weight values for scales 1 and 2 are given in Table 6.2.

6.2.3.3. *Tracing Anomalies.* The next phase is to traverse the links that had a contribution to the anomaly. Fig. 6.6 presents the pseudocode for the traversal. Two types of traversals are involved depending on the direction of the traffic analyzed. Forward tracking (by invoking trace(forward)) is performed by iteratively comparing an inbound link with the outbound links connected to the same node.

Similarly backward tracking (by invoking trace(backward)) is performed by iteratively comparing an outbound link with inbound links connected to the node. Tracing terminates when no links show significant contribution to the anomaly. This approach selects links contributing to the anomaly explicitly, and generates a map of the spread of the considered anomaly over the entire network.

```
function TRACE($direction$)                    ▷ $direction = enum\{forward, backward\}$
    Initialize $LinkSet \leftarrow \{\}$
    if $direction == forward$ then $LinkSet \leftarrow outgoinglinks$
    else$LinkSet \leftarrow incominglinks$
    end if
    $ChosenLinks \leftarrow$ SELECT($LinkSet$)
    if $ChosenLinks == \{\}$ then
        exit
    else
        for all $link$ in $ChosenLinks$ do TRACE($direction$)
        end for
    end if
end function

function SELECT($LinkSet$)
    Initialize $window \leftarrow$ window of anomaly on current link                    ▷
$window = struct\{StartTime, EndTime\}$
    Initialize $ChosenLinks \leftarrow \{\}$
    for all $link$ in $LinkSet$ do
        Move $StartTime$ to start of anomaly on the $link$
        Move $EndTime$ to end of anomaly on the $link$
        if $(StartTime - EndTime) > 0$ then $ChosenLinks \leftarrow ChosenLinks \cup link$
        end if
    end forreturn $ChosenLinks$
end function
```

FIGURE 6.6. Pseudocodes for anomaly tracing

The only information passed from one iteration at a node to the next iteration at the other node is the duration information (time window) of the anomaly. The time window is moved and stretched/shrunk to find the best correlating time window to pick the anomaly. The same anomaly may be marked at different routers at different timestamps, due to propagation delays time synchronization tolerances, and duration variation due to traffic shaping. Matching the best correlating time window overcomes these issues.

6.2.4. RESULTS. We use the anomaly detection and traversal methods presented above to investigate properties of anomalies. The revealed properties are characterized by fitting to appropriate statistical distributions.

6.2.4.1. *Anomaly Detection and Tracing.* Fig. 6.7 shows a sample trace of a detected anomaly (marked in red) in the link from Denver to Kansas City. Haar wavelet-based tracing indicates that the anomaly originated in the Sunny Valley to Denver link (by backward tracking), and terminated after the Kansas City to Indianapolis link (by forward tracking). For the studies presented some 4313 anomalies observed in a period of 50 weeks starting from Oct. 16th, 2005 are used.

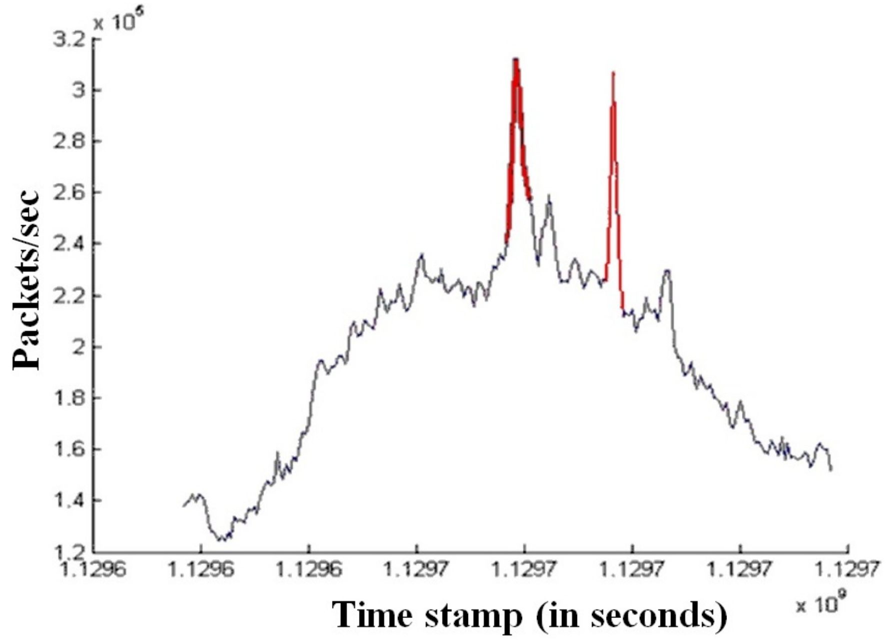


FIGURE 6.7. Traffic from Denver to Kansas City link (the spikes marked in red are anomalies detected)

6.2.4.2. *Anomaly Characterization.* Next we characterize the anomalies observed over the entire network, in terms of the distribution of their time duration, and the distribution of their spatial spread. Since anomalies are rare occurrences, data has to be collected over a long period of time to calculate reasonably accurate statistics. A statistically significant sample set large enough to evaluate properties of anomalies is generated by averaging traffic over 10 weeks.

Fig. 6.8 shows the average time duration characteristics of anomalies over the entire network. These results show the distribution of the time duration of anomalies in five 10-week periods. Time distribution shows that they all follow the same distribution, as well show a thorough compliance to a *Geometric distribution* with parameter 0.5 in timestamps. This means that about 50% of anomalies die within 5 minutes. About 25% of the anomalies last beyond 5 minutes but disappear within 10 minutes, etc.

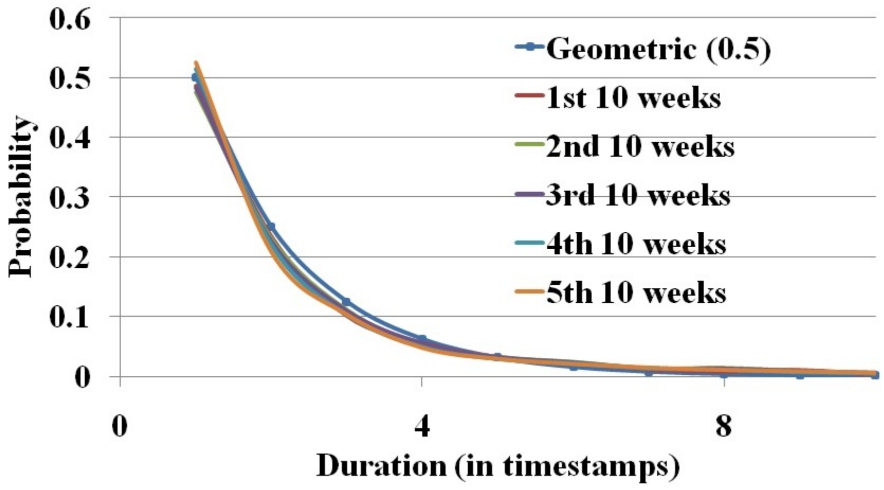


FIGURE 6.8. Probability distribution of average time duration of anomalies

The spatial spread is assessed in terms of number of links involved in the anomaly, and depth to which the anomaly descends. The spatial spread distribution explains the probability an anomaly would prevail in a particular number of links down the network. This distribution heavily depends on the structure of the network. Though the probability distribution for any 10 weeks period is found to be nearly the same, finding a standard distribution that will fit well (as with the case of duration analysis) is not possible.

The "depth" an anomaly propagates is less network-structure-dependent than the set of links affected by the anomaly. Therefore depth is a more reasonable measure for spatial analysis. Here, the number of levels an anomaly spread is counted, instead of the number of links. We found the distributions to be well converged, and to be close to a *Log-Pearson type*

III distribution. The *Kolmogorov-Smirnov* (KS) test [154] verified this claim. The resulting distribution is shown in Fig. 6.9.

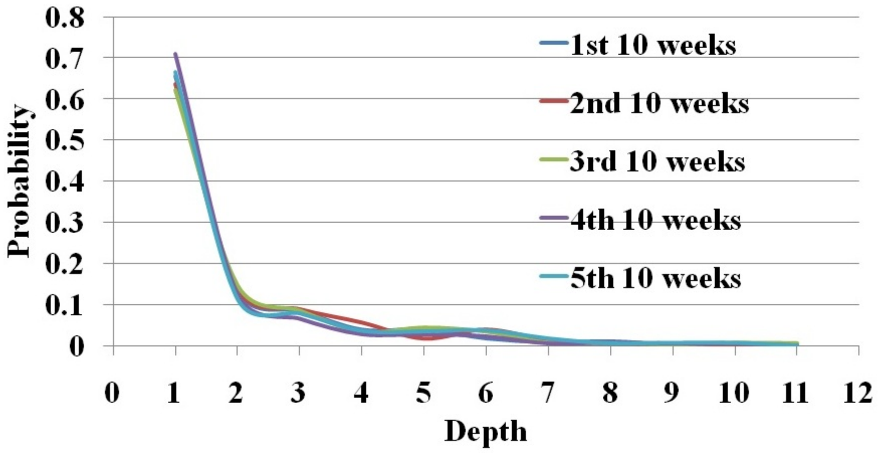


FIGURE 6.9. Probability distribution of spatial spread (in terms of depth of the anomaly)

The KS test [154] was performed on all the 10-week datasets to assure our claims. For most of the observation sets, KS-test confirmed similarity to the claimed distributions with a P-value close to 1.

6.2.5. ANOMALY MODEL. Being able to regenerate anomalies similar to those in practical networks is essential for accurate network modeling, evaluation and forecasting. Based on the properties of anomalies observed, a model to describe the origination, propagation and termination characteristics of anomalies is proposed next. The model is intended to capture the statistical properties of anomalies. Distinct from other Internet traffic models, it identifies and characterizes different properties of anomalies. Such a model is useful for applications such as Internet traffic simulators - to generate anomalies having realistic statistical properties. The model is validated by comparing the statistics of anomalies generated by the model and actual anomalies observed in the network. Statistics of the anomalies generated by the model showed a maximum *Kullback-Leibler (KL) divergence* of 8% from the statistical properties of detected anomalies. KL divergence provides a distance between two probability densities compared. A low distance value claims the probability densities are similar. Accuracy of the model further verified when applied over a path as shown later in Section 6.2.7.

6.2.5.1. *Proposed Model to Emulate Anomaly Behavior at a Node.* The proposed model captures the behavior of anomalies at a network node. Fig. 6.10 shows the basic structure of the proposed model for a node having three links. Traffic flows marked are only for the link-i. Each link connects the node at an interface block, represented with the detailed structure shown in Fig. 6.11. To characterize the core, a splitting ratio is used where a fractions $\alpha_{i,j}$ ($\alpha_{i,k}$), of anomalies received at interface-i propagate to interface-j (or k).

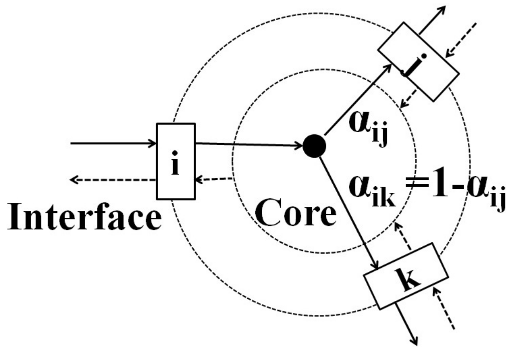


FIGURE 6.10. Basic anomaly model for a network node

Parameters of the model are derived using the characteristics observed in the network. The core is specified using a *splitting ratio*, which is the fraction of anomalies that would take each path. The interface block is specified using two probability values and three probability distributions as presented in Table 6.3, which also indicates the values for an example node based on measurements. P_{abs} is the probability a received anomaly would be absorbed, i.e., moved out of the network. For certain anomalies a responding anomaly can be observed on the same link in the reverse direction. These can be interpreted as acknowledgements for the anomalies received. The probability of having a responding anomaly is identified

Table 6.3. Parameters in the Interface Block

Parameter	Definition	Sample values (for Kansas City nodes Denver interface)
P_{abs}	Probability of absorbing a received anomaly completely	68%
pdf_{gen}	Distribution of inter anomaly gap (in hours)	Exponential(45.33)
P_{res}	Probability of responding to a received anomaly	3%
pdf_{vol}	Distribution of volume adjustment	Normal(1,0.188)
pdf_{res_vol}	Distribution of volume change of the responding anomaly	Normal(1,0.305)

with P_{res}. The *inter-anomaly gap* is found to have an exponential distribution (noted with Exponential(mean) and valued in hours) is marked as pdf_{gen}. When anomalies propagate and get responded their volume is adjusted. These factor by which the volume is adjusted is explained using a normal distribution (noted with Normal(mean, standard deviation)) and identified with pdf_{vol} and pdf_{res_vol} respectively.

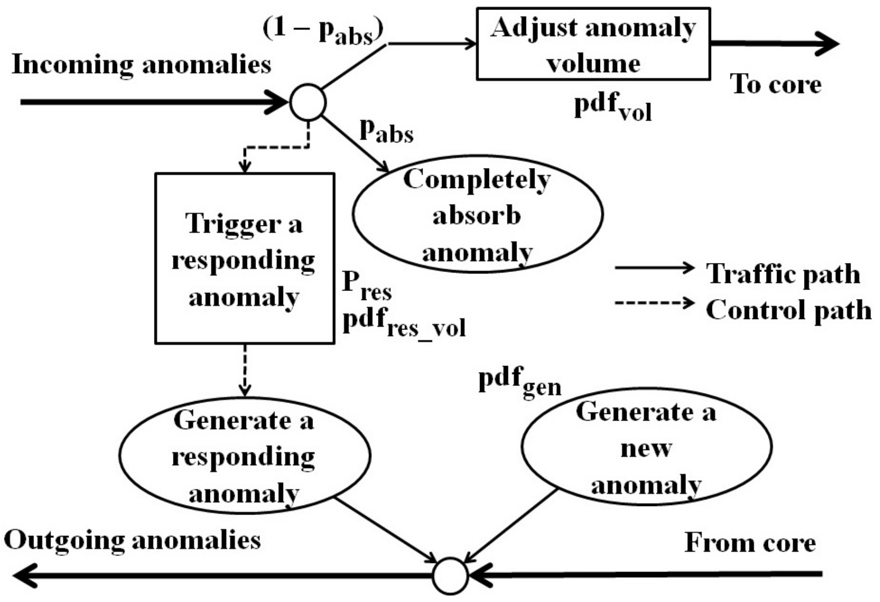


Figure 6.11. Structure of the interface model

6.2.5.2. *Utilizing the Model.* The proposed model is capable of regenerating anomalies having similar statistical properties as observed in the actual network. The model covers the three types of outgoing anomalies:

(1) An originating anomaly
(2) An anomaly as a response to a received anomaly
(3) As propagation of a received anomaly

The inter-arrival time of originating anomalies and the initial anomaly volume showed a wide range of behaviors on different links. Yet in general, it could be observed that an originating anomaly would have about 3 million to 18 million packets distributed logarithmically, and the inter-arrival time between originating anomalies have a Poisson distribution. When inter-arrival times of anomalies of each link were fitted with an exponential distribution they only had standard error of about 5%. An example case is shown in Fig. 6.12, for anomalies observed in the Chicago to New York link. The histogram of inter-arrival times of anomalies is shown in bars in Fig. 6.12. The histogram can be closely approximated to an exponential curve as shown.

It was also noted that the distributions used to characterize the volume adjustment when an anomaly propagate, and when a responding anomaly is generated could be approximated using a normal distribution. A standard error of about 10% was seen at worse case.

Once the model parameters (shown in Fig. 6.10 and 6.11) are identified for a network, anomalies having similar statistical properties can be generated. The complete process to regenerate anomalies having similar statistical behaviors to the observed is summarized using a pseudocode in Fig. 6.13. Originating anomalies are generated with an inter-anomaly gap of pdf_{gen}. A receiving node generates a responding anomaly in the reverse link with probability P_{res}, with the volume related to the incoming anomaly by a multiplicative factor randomly

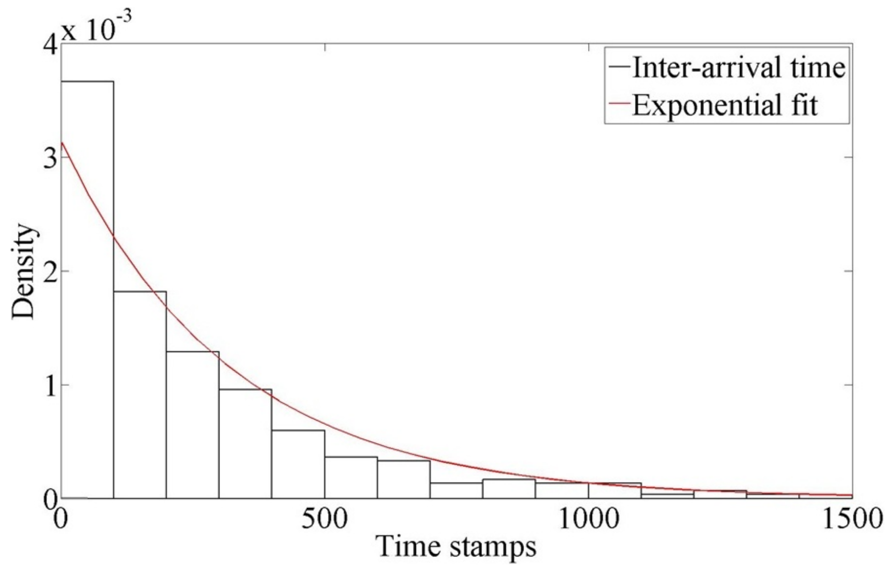


FIGURE 6.12. Inter-arrival time distribution on the Chicago to New York link Structure of the interface model

distributed according to pdf_{res_vol}. A receiving node absorbs the received anomaly with P_{abs}, or propagates it after changing the volume by a multiplicative factor given by pdf_{vol} and forwards to the core. If the node has more than one out-going link, the anomaly will choose a link with the links splitting ratio. The outgoing interface spread the anomaly volume over a period given by geometric(0.5) distribution.

6.2.6. ANOMALY PARAMETERS OF INTERNET2. The model captures the statistical properties of anomalies observed at a node. Thus, now it becomes possible to evaluate or classify the anomalies, and also regenerate anomalies having the same statistical behavior as observed at each node. Tables 6.4 and 6.5 presents model parameter values for the Internet2 network. Table 6.5 lists the splitting ratios. A propagating anomaly would pick an outgoing link based on the splitting ratio.

The link having a higher ratio is referred as the “dominant link” and the other as the “other link.” As many phenomena of interest such as buffer overflow, packet loss, etc. occur due to traffic anomalies, such characterization is useful for network design and resource

```
function ANOMALY RECEIVE PROCESS(anomaly)
    RESPONDING ANOMALY GENERATION PROCESS(anomaly)
    With $P_{abs}$ absorb anomaly
    if anomaly not absorbed then
        Adjust volume according to $pdf_{vol}$
        ANOMALY PASSING TO CORE PROCESS(anomaly)
    end if
end function

function RESPONDING ANOMALY GENERATION PROCESS(anomaly)
    With $(1 - P_{abs})$ discard anomaly
    if anomaly not discarded then
        Construct an anomaly according to $pdf_{res\_vol}$
        Force an outgoing anomaly
    end if
end function

function ANOMALY PASSING TO CORE PROCESS(anomaly)
    According to splitting ratio choose the outgoing link
    Pass the anomaly
end function

function ANOMALY GENERATION PROCESS
    According to $pdf_{gen}$ OR by a trigger from RESPONDING ANOMALY GENERATION PROCESS
        Spread the anomaly over time according to geometric(0.5)
        Transmit anomaly
end function
```

FIGURE 6.13. Pseudocodes to implement the model

provisioning. Notably, only a few parameters (measured locally) are required to capture the statistical properties of anomalies over the entire network. As shown Section 6.2.7, these parameters can be extended over a path or a region. An extension to the model over a path and region is discussed in Section 6.2.7.

6.2.7. APPLICATION OF THE MODEL.

6.2.7.1. *Anomaly Propagation Characteristics.* The model proposed above captures propagation of anomalies over the network. The probability that an anomaly would survive over

Table 6.4. Parameters in the Interface Block

Node	Link	P_{abs} (%)	Standard Deviation of Normal pdf_{vol}	P_{res} (%)	Standard Deviation of Normal pdf_{res_vol}	Mean of Exponential pdf_{gen} (hours)
KSCY	DNVR	39.3	2.17	23.5	0.926	45.33
	IPLS	31.8	0.617	14.4	1.27	42.28
	HSTN	77.1	57.8	29.4	2.60	33.86
HSTN	KSCY	73.0	31.9	31.6	2.50	37.96
	LOSA	37.8	16.4	33.2	48.3	42.95
	ATLA	48.8	1.48	28.2	2.10	28.57
DNVR	STTL	65.0	12.9	22.5	2.07	41.96
	SNVA	50.8	6.68	31.0	2.68	24.11
	KSCY	23.0	1.72	15.9	10.3	38.38
IPLS	KSCY	30.1	0.599	18.0	1.83	41.79
	CHIN	24.0	5.97	19.1	14.4	44.22
	ATLA	73.5	12.5	26.1	1.02	37.11
ATLA	WASH	48.5	2.84	31.8	2.07	42.61
	IPLS	85.6	5.38	20.7	1.25	45.59
	HSTN	47.2	5.69	30.5	1.86	35.52
SNVA	STTL	83.9	21.3	11.8	20.0	35.71
	DNVR	25.2	4.82	21.7	1.47	52.64
	LOSA	29.7	1.89	27.5	2.36	16.46
STTL	SNVA	93.1	201.0	13.1	114.0	30.13
	DNVR	92.5	3.19	19.3	2.31	42.52
LOSA	SNVA	86.9	5.93	20.1	2.52	21.80
	HSTN	81.9	2.76	30.8	5.24	25.88
WASH	ATLA	76.5	3.60	31.7	3.25	29.73
	NYCM	82.6	1.55	9.76	3.38	49.90
NYCM	WASH	75.0	20.2	13.1	0.976	39.94
	CHIN	60.7	0.838	17.1	1.74	21.56
CHIN	NYCM	59.1	5.09	12.4	3.01	33.02
	IPLS	74.4	2.67	17.5	2.11	29.35

a path depends on P_{abs}s and $\alpha_{i,j}$s of intermediate nodes and given by:

$$P_W = \prod_{i \in W} \left((1 - P_{abs,i})\, \alpha_{c,i} \right) \tag{6.2}$$

where, $W = \{\text{links of the path}\}$, $\alpha_{c,i}$ is splitting ratio for the chosen outgoing link from link-i. A comparison between the P_W's derived from the model and the actual is shown in Table 6.6. It compares the probability an anomaly would survive over path provided by the

TABLE 6.5. Splitting Ratios

Link	Preferred link	Splitting ratio (%)	Other link
ATLA-HSTN	HSTN-LOSA	95.0	HSTN-KSCY
ATLA-IPLS	IPLS-CHIN	70.5	IPLS-KSCY
CHIN-IPLS	IPLS-KSCY	96.1	IPLS-ATLA
DNVR-KSCY	KSCY-IPLS	88.5	KSCY-HSTN
DNVR-SNVA	SNVA-LOSA	97.0	SNVA-STTL
HSTN-ATLA	ATLA-WASH	98.6	ATLA-IPLS
HSTN-KSCY	KSCY-DNVR	55.8	KSCY-IPLS
IPLS-ATLA	ATLA-WASH	87.5	ATLA-HSTN
IPLS-KSCY	KSCY-DNVR	93.7	KSCY-HSTN
KSCY-DNVR	DNVR-SNVA	82.8	DNVR-STTL
KSCY-HSTN	HSTN-LOSA	63.3	HSTN-ATLA
KSCY-IPLS	IPLS-CHIN	94.5	IPLS-ATLA
LOSA-HSTN	HSTN-ATLA	90.4	HSTN-KSCY
LOSA-SNVA	SNVA-DNVR	89.9	SNVA-STTL
SNVA-DNVR	DNVR-KSCY	95.7	DNVR-STTL
STTL-DNVR	DNVR-KSCY	86.4	DNVR-SNVA
STTL-SNVA	SNVA-LOSA	61.5	SNVA-DNVR
WASH-ATLA	ATLA-HSTN	90.9	ATLA-IPLS

TABLE 6.6. Comparison Between P_Ws

Propagation	Actual P_W (%)	Model P_W (%)	Error
LOSA-SNVA-DNVR-KSCY	68.1	70.4	2.3
NYCM-WASH-ATLA-HSTN	95.0	88.9	6.1
STTL-SNVA-LOSA-HSTN	98.6	98.6	0.0
CHIN-NYCM-WASH-ATLA	92.5	93.0	0.5
IPLS-ATLA-HSTN-LOSA	98.7	99.1	0.4

model and actually observed. The error column states the difference between the probability values. The "Actual P_W" is the value observed in the network. The "Model P_W" is the value calculated using model parameters using (6.2). The error columns is the difference between the two probability values.

As an anomaly propagates, its volume changes according to pdf_{vol} of each node in the path. The effective pdf of volume adjustment over a path is the product of individual pdfs over each link. As these pdfs are independent, the effective pdf can be expressed as follows:

$$pdf_{vol,W} = \mathbb{M}^{-1}\left\{\prod_{i\in W}\mathbb{M}\left\{pdf_{vol,i}\right\}\right\} \tag{6.3}$$

where $W = \{\text{links of the path}\}$, $\mathbb{M}$ is the Mellin transform [155]. When the above statistics were derived for a number paths we observe a standard error of about 7% between the estimated and actual values.

6.2.7.2. *Model for Node Aggregates.* Further attempts were made to devise a higher level node model, which can capture a region, as in Fig. 6.14. Having each node characterized with a very few parameters, eases forming higher level nodes.

Aggregating nodes hides links, and a region has multiple internal paths. Entry points to the region will become interfaces to the aggregated node. The probability of absorption (P_{abs}) for an interface in the aggregated node is given by:

$$P_{abs} = 1 - \sum_{w\in T} P_w \tag{6.4}$$

where, T is the set of all internal paths starting from the interested interfaces of the aggregated node.

The splitting ratio of an anomaly arriving at interface-i, choosing interface-j ($\alpha_{i,j}$) depends on all the possible internal paths exist between the two interfaces, and the splittings encountered internally, and is given by:

$$\alpha_{i,j} = \sum_{t\in T_{i,j}}\prod_{k\in t}\alpha_k \tag{6.5}$$

where, $T_{i,j}$ is the set of all the paths between interface-i to interface-j, α_k splitting ratio of choosing link-k.

When (6.4) and (6.5) are applied for the portion of network indicated in Fig. 6.14(a), to reach Kansas City from Los Angeles, there are two internal paths in the aggregated node. The first path has a probability of 29.75% to reach the interface to Kansas City. The path-2 (through Seattle) has a P_W of 0.16%. Thus, the cumulative probability to reach at the interface to Kansas City from Los Angeles interface is 29.91%. Therefore according to the model P_{abs} for the interface towards Los Angeles is 70.1%. The actual value was found to be 62.4%.

6.2.7.3. *A Real-time Distributed Monitoring System.* Although the analysis presented here is aimed at deriving a statistical model, the analysis can easily be extended to a real-time distributed system for monitoring anomalies. The wavelet coefficients presented in Section 6.2.7 are computed using traffic around a node. Thus, each router is capable of constructing a data-structure with its wavelet coefficients.

Management Information Base (MIB) on each router will perform the Fourier based detection on each of its links, and the wavelet analysis to produce the data-structure with the coefficients shown in Table 6.1. By exchanging the data-structures with neighbors a description of anomalies in the local network can be constructed. The data-structure of a node contains the anomaly information on all its links. When a node is aware of it neighbors' data-structures, the node is aware of anomalies in a local network up to two links deep.

When an anomaly propagates towards a certain node, the observing node could forward all known coefficient data-structures to the same node. Thus, all nodes experiencing an anomaly have the capacity to develop the complete spatio-temporal map of the anomaly. Fig. 6.15 shows the pseudocode for this scheme. Such a scheme enables any node observing an anomaly to derive properties of the anomaly. If the anomaly behavior of the network has been modeled, then these properties will enable predicting and decision making.

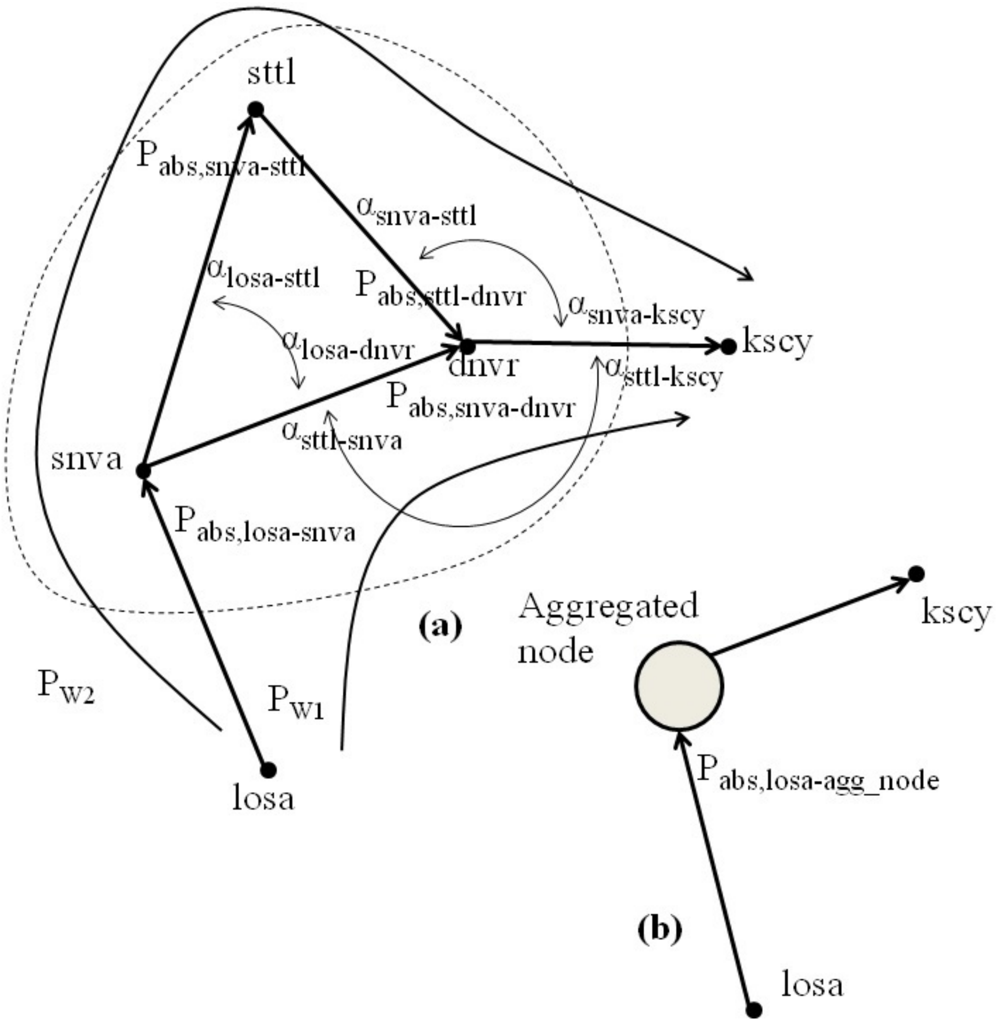

sttl
$P_{abs,snva\text{-}sttl}$
$\alpha_{snva\text{-}sttl}$
$\alpha_{losa\text{-}sttl}$
$P_{abs,sttl\text{-}dnvr}$
$\alpha_{snva\text{-}kscy}$
kscy
$\alpha_{losa\text{-}dnvr}$
dnvr
$\alpha_{sttl\text{-}kscy}$
$P_{abs,snva\text{-}dnvr}$
$\alpha_{sttl\text{-}snva}$
snva
$P_{abs,losa\text{-}snva}$
(a)
Aggregated node
kscy
P_{W2}
P_{W1}
losa
$P_{abs,losa\text{-}agg\_node}$
(b)
losa

FIGURE 6.14. Portion of the network made into a higher level node

6.2.8. CONCLUSIONS. A method to characterize and model Internet traffic anomalies was proposed. To detect anomalies a simple Fourier-based method was employed. A new approach based on graph wavelets was developed to analyze the spatial and temporal behavior of anomalies. Measurement data for Internet2 was used to evaluate model parameters and to validate the region model. In particular, we studied how an anomaly propagates from one link to other links connected to the same node in the network (or vice versa). We demonstrated that the time duration and spatial spread of anomalies, observed in the Internet2 data used in this section, can be adequately captured by standard statistical distributions. A node model capable of capturing the input-output relation for the anomaly traffic in a

```
function Throughput sampler
    Set Timer to sampling time
    At expiry
        Sample all links
    Anomaly detector
    Form wavelet coefficients
    Exchange coefficients

end function

function Anomaly detector
    Apply Fourier transform to sample record
    Threshold and mark anomalies
end function

function Form wavelet coefficients
    Perform link comparison to form wavelet coefficients
end function

function Exchange coefficients
    Send coefficients to all neighbors
    Receive coefficients from all neighbors
    Construct message with own coefficients and neighbor coefficients
    Forward the message in the anomaly direction
end function
```

Figure 6.15. Pseudocodes for real-time distributed monitoring system

node was developed. This model was then extended to a composite input-output model, capturing the anomaly propagation over a path or region.

6.3. Spatiotemporal Model for Internet Traffic Anomalies

Models for Internet traffic anomalies greatly benefit a range of applications including robust network design, network provisioning and performance studies. A novel approach to analyze and model network traffic anomalies is presented. The proposed approach individually characterizes different aspects of anomalies, such as origin, termination, propagation, and changes in duration and volume, with common random processes. These characteristics are then integrated into a single model that successfully captures the overall anomaly behaviours. Characterization of each anomaly property requires only a few parameters, leading to a concise set of parameters for the entire model. Though the model is calibrated with local measurements made at nodes, it successfully represents the global behaviours of anomalies over the network. The proposed model is applicable both at nodal level and at subnet level. This enables hierarchically analyzing large and sophisticated networks. Anomalies are analyzed using a multi-scale analysis framework based on which, a real-time monitoring system that efficiently communicate ongoing anomaly information across the network is developed. The system is also used for learning regional model parameters distributively. Internet2 traffic data is analyzed using the framework, and the corresponding model parameters are derived. These results provide insight on the nature of anomalies in networks.

6.3.1. Introduction. Characterizing the nature of traffic anomalies such as those due to flash crowds, Distributed Denial of Service (DDOS) attacks and link failures is crucial for robust network design, operation and forensics. Behaviours of these anomalies, in time and space, and the correlations among these behaviours are referred to as spatiotemporal characteristics. Spatial properties of interest include the links, subnets and regions affected, while temporal properties of interest include amplitude variations, durations, and rates of anomalies. These properties reveal vulnerable regions and periods in network operation. Modelling anomalies aims at capturing anomaly properties coherently and concisely. Representative models for the characterization of spatiotemporal anomaly properties help provisioning of network resources such as link capacities, trunks, router capacities and failovers. Efficient and scalable frameworks to extract anomaly features also facilitate calibration of models, which in turn are useful for design, evaluation and forensics. Anomaly models can significantly enhance applications such as network simulators. Anomaly modelling provides different levels of abstractions, such as activities at link/node level and aggregated activities at ISP (Internet Service Provider) level. Despite the significant importance of anomaly modelling, formal models that effectively capture spatial and temporal features of anomalies are yet to emerge. This is mainly due to the difficulty in modelling anomalies as pointed out in [156]. Traffic anomalies do not adhere to common random processes as they are sporadically scattered in time and space with arbitrary durations, gaps, volumes, and spreads. This poses a challenge for modelling traffic anomalies with traditional approaches. However, this dissertation achieves a formal model that successfully characterises network traffic anomalies.

Two main contributions are made herein. First, we present a multi-scale framework for network traffic anomaly analysis. A spatiotemporal filter that extracts the duration and volume information of anomalies at various scales is proposed for this purpose. The filter outputs summarise the variation of volumes of anomalies and the durations they are in effect. Second, using the above filter outputs and time gaps between anomalies, we characterize individual properties of anomalies and integrate them into a comprehensive model that captures the input-output anomaly relationships at a node. We also extend this model to describe the behaviours of anomalies at subnet level. As each component of the model accurately captures various aspects of anomalies, such as inter-anomaly gaps and anomaly volume/duration distributions, the proposed integrated model accurately represents the overall anomaly behaviours. The model parameters identified adhere to common random processes, successfully and concisely describing different anomaly features. We demonstrate that the model apprehends anomaly behaviours both at node level and at subnet level with the same limited set of parameters and relationships. Therefore the proposed model characterizes the network at a desired level of granularity. The attention of this work is on the traffic intensity and volume anomalies, i.e., unusual deviations in throughputs. We note however that the techniques developed here are valid for other types of measurements that can be represented as arrays of time-series.

Much of the literature on anomalies is focused on detection rather than modelling, though detection does assume an underlying anomaly model. Detection schemes are specialized and fine-tuned based on properties of anomalies, implicitly modelling anomalies. Initial attempts to model anomalies vary from approaches based on simple thresholding for anomaly detection to localizing anomalies in certain frequency bands [136] or wavelet coefficients [29], followed by statistical characterization. Many approaches model regular trends, for example with Fourier analysis [138], and declare significant deviations as anomalies. Other anomaly detection techniques have been developed based on principal component analysis [157], machine learning [141], data mining [142], statistical analysis of payloads [143], risk modelling, and Bayesian analysis [144, 145]. A relatively low complexity spatial analysis is presented in [148], which can be implemented at individual routers across the network to alleviate the load at a central location. Correlation Layers for Information Query and Exploration (CLIQUE) [158] models anomalies in streaming datasets by indicating atypical patterns. Generalized Anomaly and Fault Threshold (GAFT) [159] is a hierarchical, multi-tier, multi-window, soft-fault detection system based on statistical models. Other statistical approaches for anomaly modelling use Chi-Square [160] and Bayesian models [161]. A multi-tier extension of Bayesian models is discussed in [162]. Statistical models provide pragmatic approaches for anomaly detection as pointed out in [163]. Hidden Markov Models are another approach used to describe anomaly activities [164, 165]. An adaptive approach for anomaly detection using a model with a simple architecture is discussed in [166]. Among the other approaches are the usages of danger models [167] and cluster analysis [15]. In [168], knowledge gained by anomaly models is used to improve control mechanisms such as flow sampling. Model selection for anomaly detection in wireless ad-hoc networks is addressed in [169], where an extended anomaly model is employed in WLANs (Wireless Local Area Networks) with RSVP (Resource Reservation Protocol) mechanism to improve QoS (Quality of Service). Data adaptive approaches for traffic analysis include use of diffusion wavelets [150], e.g., for dimension reduction of network traffic [151, 12]. Graph wavelets [149, 7] enable spatial traffic analysis at different granularities. A graph wavelet based approach for modelling Internet traffic anomalies is presented in [149], while [7] uses graph wavelets without losing granularity.

Models for anomalies have a wide range of practical applications, only a few of which are briefly addressed here. Simulators for network traffic greatly benefit from anomaly models. Typically, the normal component of traffic is well behaved allowing it to be reproduced/regenerated accurately with common distributions. The anomalous components however are difficult to be reproduced due to their complicated spatial and temporal behaviours and dependencies. A model that captures statistics of anomaly features can facilitate synthesizing statistically accurate anomalies and ultimately realistic traffic traces. An alternative is to record and store traces of real traffic and then drive the simulators. However such trace-driven approaches are not scalable, and are often not feasible even for moderate sized networks. The model presented here captures the vital features of traffic traces. Storing only such characteristics of the traces is scalable and requires orders of magnitudes less storage. Further, realistic anomalies for traffic streams can be successfully synthesized with the stored statistics. These features are also useful in specifying anomalies. A formal and concise specification would be needed, for example, to stipulate descriptions of anomalies for network analysis, and to query for anomalies exhibiting a certain behaviour. The model is also a prediction tool. The model describes the probability at which an anomaly can be expected with certain features. Robust network design can greatly benefit from such anomaly models. Information captured by the model parameters also has a design value. They reveal interesting information about the network, such as distributions of the severity of anomalies, which may otherwise be difficult to obtain. Such information is useful for applications such as provisioning of resources. The proposed scheme models anomalies at various levels of abstractions. Therefore the model helps assess anomalies over the network at different levels of hierarchy, such as node and subnet levels.

The rest of the section is arranged as follows. Section 6.3.2 discusses properties of anomalies and the analysis framework. The proposed model is presented in Section 6.3.3 and accompanied by modelling at the node level. Modelling at subnet level is discussed in Section 6.3.4. Concluding remarks are in Section 6.3.5. An appendix in Section 6.3.6 provides the details of the anomaly extraction process.

6.3.2. Internet Traffic Anomalies and Their Properties. This section takes an extensive look at characteristics of traffic anomalies. Internet traffic anomalies are complex combinations of volumes, durations, and gaps. Therefore describing traffic anomalies with a single standard random process is challenging if not impossible. However individual temporal and spatial anomaly properties can be described more successfully with well-known random processes.

We discuss the analysis procedure with a running example based on the Internet2 network data. The process however is general and applicable to any network. The dataset from Internet2 network [152] shown in Fig. 6.16 [2], contains throughput measurements sampled every five minutes starting October 16th, 2005. The nodes in Internet2 network are abbreviated as: Atlanta (ATLA), Chicago (CHIN), Denver (DNVR), Houston (HSTN), Indianapolis

(IPLS), Kansas City (KSCY), Los Angeles (LOSA), New York (NYCM), Sunnyvale (SNVA), Seattle (STTL), Washington (WASH).

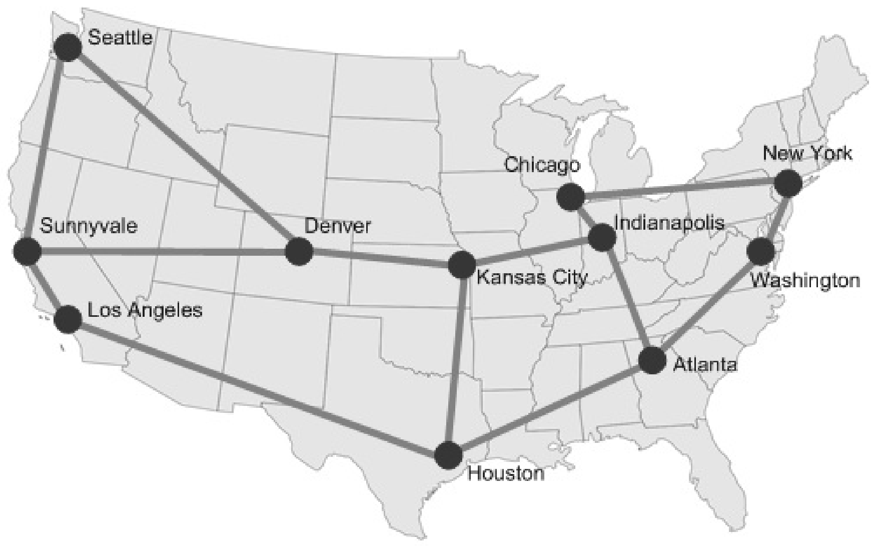


FIGURE 6.16. The Internet2 network [2], consisting of 11 nodes and 14 bi-directional links.

6.3.2.1. *Classes of Anomalies.* We classify anomalies into three classes: originating, propagating and responding. This categorization is based on the generation of the anomalies with respect to the node they are observed at. Each class has distinct properties and builds a separate component of the proposed model.

Originating anomalies: Anomalies that originate at the considered node are termed "originating anomalies.". The process of originating anomalies is described with three features: (1) the interval between anomalies, (2) the volume of each anomaly, and (3) the duration of each anomaly. In other words, anomalies of random volumes and random durations are generated at random intervals.

Propagating anomalies: Anomalies that originate at a different node and pass through the observing node, are called "propagating anomalies." However, not all anomalies arriving at a node propagate further. A fraction of anomalies gets absorbed. The outgoing anomalies may have different volumes and durations than the corresponding incoming anomalies. Therefore the ratios between the incoming and outgoing anomaly volumes and durations are used to characterize the reshaping of propagating anomalies.

Responding anomalies: An interface of a node has an inbound link and an outbound link. In certain cases, an anomaly is also observed on the outbound link over a time period that overlaps with the duration of an anomaly on the inbound link. Such anomalies on the outbound link are referred to as "responding anomalies." A probability value is used to describe the probability of observing a responding anomaly. Similar to propagating anomalies, ratios relating the volumes and durations of the responding anomaly and incoming anomaly are used for characterization.

6.3.2.2. *Spatial and Temporal Properties of Anomalies.* Network traffic anomalies are endowed with a number of spatial and temporal properties. Temporal properties are observed at a point in the network over time, such as at an interface of a node or a tapping point on a link. It produces a time-series of the anomalous activities. Properties describing the spread of anomalies over the network are spatial properties. They describe features such as nodes where anomalies were absorbed or propagated, number of links the anomaly affected, etc. We find the following anomaly properties to be of interest:

Inter-anomaly gap: The inter-anomaly gap is the time interval between two adjacent anomalies.

Duration / duration change: The length of the period the anomaly is in effect is the duration of the anomaly. For example, durations of the anomalies shown in Figs. 6.17(b), (c) and (d) are one, two and two sample periods, respectively. As anomalies propagate, the duration they are in effect also changes, which can be described using a multiplicative factor. In the example from Fig. 6.17, the duration of the incoming anomaly doubles as it propagates out, i.e., by a factor of two.

Volume / volume change: Volume of an anomaly is the area under the anomalous trace over the duration of the anomaly of interest, or in other words, anomalous throughput integrated over the duration of the anomaly. For the anomalies shown in Fig. 6.17(b), (c) and (d), the volumes are 8.82×10^6, 2.35×10^7 and 8.52×10^6 packets, respectively. Volume change can be characterized similar to duration change. As the incoming anomaly on KSCY to IPLS link propagates, its volume has grown by a multiplicative factor of 3.63 .

Absorption: When an anomaly does not propagate beyond a certain node, it is considered absorbed at that node.

Split / join: As an anomaly propagates across a node with multiple outgoing links, the anomaly may split between outgoing links. Also when anomalies from a few links arrive at a node at the same time, they may jointly propagate onto outgoing links.

The individual features discussed above are amenable to common random processes. Therefore a feature-wise anomaly model can successfully be built. Motivated by this ability, we propose a model for network traffic anomalies, in Section 6.3.3.

6.3.2.3. *Anomaly activity at a node.* Anomaly extraction schemes in general provide time-series of anomaly measurements on individual links. Following [149], we consider volume anomalies. The extraction process is described in detail in [7] and is outlined in Section 6.3.6. This scheme de-trends to remove normal traffic and applies a threshold to separate anomalies. It produces a train of impulses with amplitudes corresponding to anomalous throughputs at

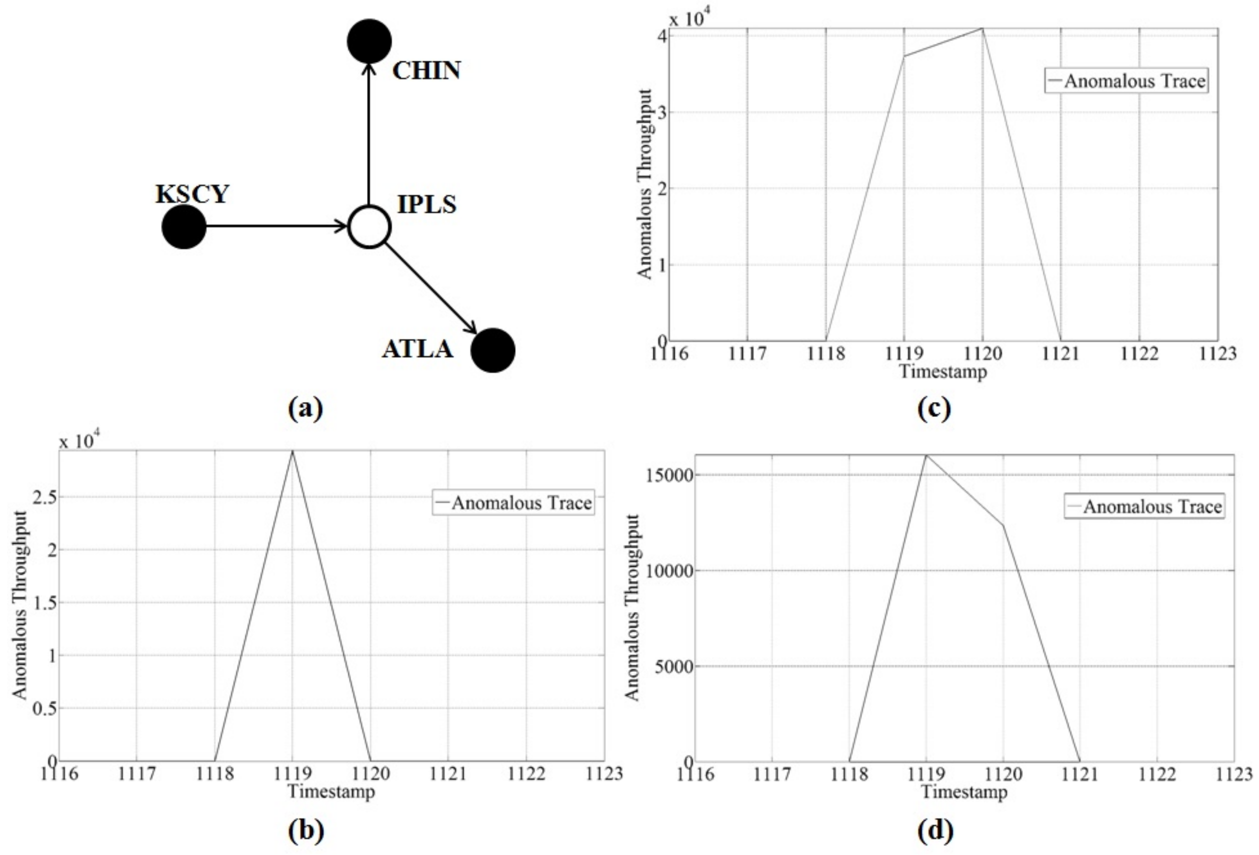


FIGURE 6.17. An example anomaly propagation: (a) The map of the propagation, (b) Trace showing anomaly in the incoming link - KSCY to IPLS, (c) Trace showing anomaly in outgoing link - IPLS to CHIN, and (d) Trace showing anomaly in outgoing link - IPLS to ATLA.

each sample time. During the anomaly free periods, the time-series is empty. When anomalies last multiple sample times, the time-series have consecutive impulses, which are treated as a single anomaly. With further treatment, three basic properties of anomalies, duration, inter-anomaly gap and volume are obtained. Each node indexes the anomaly activity on all its links and records these properties. The duration is the interval between the start time and end time. Inter anomaly gap is the interval between the start time of the current anomaly and the end time of the previous anomaly. Volume is obtained by integrating the throughput over the duration of the anomaly. Couple of example anomalies from the New York node is shown in Table 6.7. The anomaly indexed 203 entered from both the Chicago and Washington interfaces, and left from the Washington interface. This anomaly was observed after 12 samples (one hour) from a previous anomaly on the Chicago interface. The anomaly lasted for two samples (10 minutes) on the Chicago interface and for one sample time (five minutes) on Washington interface. We define $stt(k, l, in/out)$, $ent(k, l, in/out)$, $dur(k, l, in/out)$, and $vol(k, l, in/out)$ as the start time, end time, duration and volume respectively. Here, k is the anomaly index, l is the link index and in/out indicates the direction of anomaly.

6.3.2.4. *Spatial-Temporal Filtering.* The anomaly activities, such as listed in Table 6.7, needs to be analyzed in order to obtain anomaly properties discussed in Section 6.3.2. This section proposes a filtering scheme termed "Spatial-Temporal filtering" (ST-filtering) for anomaly activity analysis. The filters derive spatial and temporal properties of anomaly activities at a node by comparing volumes and durations on subsets of input and output links. Though this procedure has similarities to graph wavelets [149, 7], it is more flexible as there is no dyadic restriction on selecting subsets. Table 6.8 lists the notations used here onwards.

We propose two filters, one analyzing anomalous volumes and the other analyzing anomaly durations. The volume filter is defined as:

$$L^v_{I,J,s}(k) = \sum_{i \in I} vol(k, i, in) - \sum_{j \in J_s} vol(k, j, out). \tag{6.6}$$

The volume filter (L^v) compares the total input volume on the subset of input links I against s-size subsets of output links J, yielding the difference between the input anomalous volume on a specific subset of input links and the anomalous output volume on a specific subset of output links. Let $I = \{a, b\}$ and $J = \{c, d, e\}$ as shown in Fig. 6.18. The parameter s is called the scale, and it is the number of links in the subset to be compared at a time. Set J_s is

Table 6.7. A Sample of Anomaly Activities At New York Node

Anomaly index	Incoming link from Chicago				
	Start time	End time	Duration	Inter anomaly gap	Volume
$\vdots$	-	-	-	-	-
202	17032	17036	4	3	3.15×10^5
203	17048	17050	2	12	9.17×10^4
$\vdots$	-	-	-	-	-
Anomaly index	**Outgoing link to Chicago**				
	Start time	End time	Duration	Inter anomaly gap	Volume
$\vdots$	-	-	-	-	-
202	-	-	-	-	-
203	-	-	-	-	-
$\vdots$	-	-	-	-	-
Anomaly index	**Incoming link from Washington**				
	Start time	End time	Duration	Inter anomaly gap	Volume
$\vdots$	-	-	-	-	-
202	17032	17033	1	5	1.98×10^4
203	17048	17049	1	15	2.04×10^4
$\vdots$	-	-	-	-	-
Anomaly index	**Outgoing link to Washington**				
	Start time	End time	Duration	Inter anomaly gap	Volume
$\vdots$	-	-	-	-	-
202	-	-	-	-	-
203	17048	17049	1	206	2.02×10^4
$\vdots$	-	-	-	-	-

a set of all cardinality-s subsets of J, for example, if $s = 2$, then $J_s = \{\{c, d\}, \{d, e\}, \{e, c\}\}$. Each filter produces $\binom{|J|}{s}$ many outputs, one for each subset in J_s. The scale $s = 1$, will

Table 6.8. Notation

Notation	Description
I	Set of incoming links
a, b	Indices of the incoming links
J	Set of outgoing links
c, d, e	Indices of the outgoing links
s	Scale
k	Anomaly index
$L^v_{I,J,s}$	Volume filter (defined below)
$L^d_{I,J,s}$	Duration filter (defined below)
subscript $i - j$	Symbolises that the parameter is on anomalies going from node i to node j
N_i	The set of neighbours of node i
N_T	Statistically sufficient sample size
a_{i-j}	The set of anomalies in N_T
q_{i-j}	The subset of propagating anomalies
r_{i-j}	The subset of responding anomalies
n_{i-j}	The subset of originating anomalies
λ_{i-j}	The set of inter anomaly gaps
v_{i-j}	The set of volumes of anomalies
d_{i-j}	The set of durations of anomalies

As a_{i-j} is the set of anomalies on the link from node i to node j, the cardinality $|a_{i-j}|$ indicates the number of anomalies on the link.

compare the total incoming volume against the output volumes at individual links:

$$\begin{aligned} L^v_{\{a,b\},\{c,d,e\},1}(k) = \{&\{vol(k,a,in) + vol(k,b,in) - vol(k,c,out)\},\\ &\{vol(k,a,in) + vol(k,b,in) - vol(k,d,out)\},\\ &\{vol(k,a,in) + vol(k,b,in) - vol(k,e,out)\}\}. \end{aligned} \tag{6.7}$$

It produces three outputs, one corresponding to each output link. At scale $s = 2$, the output links are taken pair-wise:

$$\begin{aligned} L^v_{\{a,b\},\{c,d,e\},2}(k) = \{ & \{vol(k,a,in) + vol(k,b,in) - (vol(k,c,out) + vol(k,d,out))\}, \\ & \{vol(k,a,in) + vol(k,b,in) - (vol(k,d,out) + vol(k,e,out))\}, \\ & \{vol(k,a,in) + vol(k,b,in) - (vol(k,e,out) + vol(k,c,out))\}\}. \end{aligned} \tag{6.8}$$

Again, it produces three outputs, one per each pair. Finally, at the maximum possible scale $s = 3$, the total incoming volume is compared against the total outgoing volume:

$$L^v_{\{a,b\},\{c,d,e\},3}(k) = vol(k,a,in) + vol(k,b,in) - (vol(k,c,out) + vol(k,d,out) + vol(k,e,out)) . \tag{6.9}$$

At boundary setting $s = 0$ and $I = \{\}$, the input volume and output volumes are produced.

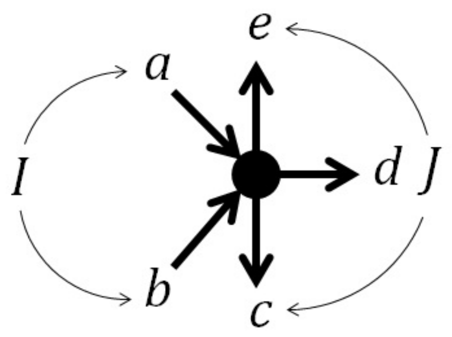


FIGURE 6.18. Analyzing anomalies at a node, whose links are grouped into an in-bound set I and out-bound set J.

Taking an example from Internet2, the filter $L^v_{\{CHIN\},\{KSCY,ATLA\},1}$ applied to Indianapolis node will return two outputs: the amount of anomalous throughput changed from Chicago interface to Kansas City interface and that from Chicago interface to Atlanta interface. The filter $L^v_{\{CHIN\},\{KSCY,ATLA\},2}$ will provide one output expressing the amount of volume changed from the Chicago interface to the sum of volumes on the Kansas City and the Atlanta interfaces.

We define the following duration filter to analyze duration change of anomalies.

$$L^d_{I,J,s}(k) = \left(\max_{i \in I}\{ent(k,i,in)\} - \min_{i \in I}\{stt(k,in)\}\right) - \left(\max_{j \in J_s}\{ent(k,j,out)\} - \min_{j \in J_s}\{stt(k,j,out)\}\right) \tag{6.10}$$

where k is the anomaly index, I and J are sets of link indices corresponding to input and output links, s is the scale.

These filter outputs are utilized in calculating anomaly properties in Section 6.3.3. Notably, the filters outputs are sufficient to explain all the volume and duration properties of anomalies.

6.3.3. Traffic Anomaly Model. Being able to regenerate actual or realistic anomalies is imperative for robust network design, studies and many other applications. The node model proposed below captures the statistical properties of the observed anomalies and is capable of regenerating anomalies having the same statistical behaviours. The model consists of two components: the core and the interface. We attribute anomaly splitting and joining behaviours to the core. Rest of the anomaly properties are described at the interface.

6.3.3.1. *The Core Model.* Figure 6.19 shows sample nodes of degrees two, three and four. In Fig. 6.19(b) anomalies entering from interface-i may propagate out from interface-j and interface-k. The "splitting ratio" α_{i-j} is for the fraction of the anomalies propagating from interface-i to interface-j. In the same fashion, a fraction α_{i-k} is defined for interface-k.

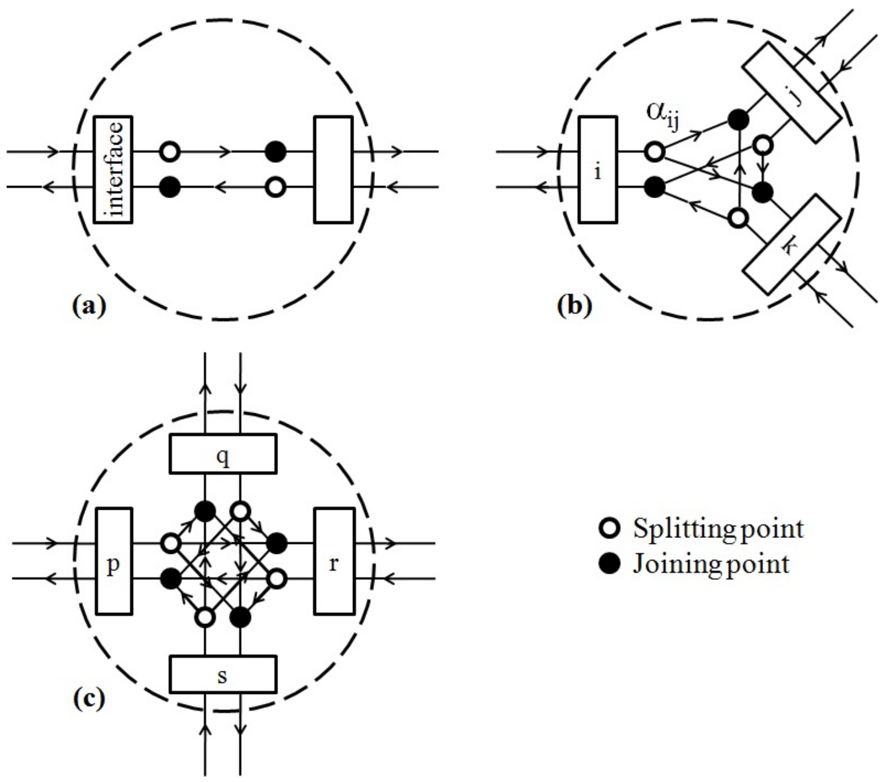


FIGURE 6.19. model: (a) Two link node, (b) Three link node, and (c) Four link node.

For a typical node b, splitting ratio of anomalies entering from node i towards node j is defined as follows:

$$\alpha_{i-j} = \frac{|a_{i-b} \cap a_{b-j}|}{\left|a_{i-b} \cap \left(\cup_{\forall l \in \{N_b \setminus i\}} a_{b-l}\right)\right|}. \tag{6.11}$$

The numerator expresses the number of propagating anomalies from node i to node j. The denominator expresses the total number of propagating anomalies entered node b from node i. The following relationship is used to derive the splitting ratio from the ST-filter outputs:

$$\alpha_{i-j} = \frac{\|L^v_{\{i\},\{j\},1}(k)\|_0}{\left|L^v_{\{i\},\{J\},|J|}(k)\right|} \text{ s.t. } L^v_{\emptyset,J,|J|-1}(k) \neq 0 \tag{6.12}$$

where k runs through the indices of the anomalies during N_T, $J = N_j \setminus i$, $\emptyset$ is the null set, $|\cdot|$ indicate the cardinality and $\|\cdot\|_0$ is the L_0-norm (the number of nonzero elements). The constraint $L^v_{\emptyset,J,|J|-1}(k) \neq 0$ asserts usage of only the propagating anomalies for splitting ratio calculation.

6.3.3.2. *The Interface Model.* Activities at the interface are illustrated in Fig. 6.20. Each link is bidirectional. Thus, there is an in and out channel on either side of the interface - towards the link and towards the core.

At a certain probability, nodes absorb anomalies. We characterize this phenomenon using a probability value p_a - the probability of absorption. The probability of absorption at node j on the interface towards node i is:

$$p_{a,i-j} = \frac{\left|a_{i-j} \setminus \left(\cup_{\forall m \in \{N_j \setminus i\}} a_{j-m}\right)\right|}{|a_{i-j}|} \tag{6.13}$$

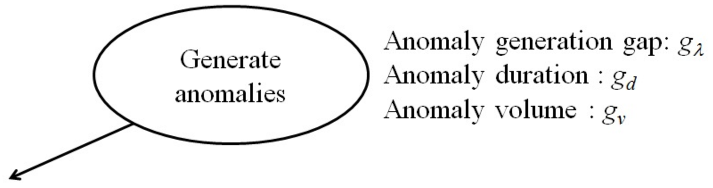
This is the ratio between the number of anomalies that did not propagate past node j and the number of total anomalies arrived from node i. The parameter p_a is calculated using ST-filter with the relationship:

$$p_{a,i-j} = 1 - \frac{\|L^v_{\{i\},J,|N_j|-1}(k)\|_0}{\left|L^v_{\{i\},J,0}(k)\right|}. \tag{6.14}$$

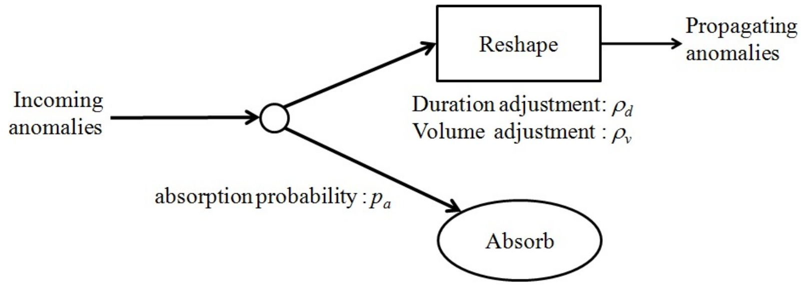
Anomaly Reshaping: If anomalies survive, they propagate onto the core. During the propagation, their volumes and durations are adjusted by factors drawn from distributions ρ_v and ρ_d respectively. To derive distributions ρ_v and ρ_d, the subset of propagating anomalies q_{i-j} defined as below for node j's interface towards node i.

$$q_{i-j} = a_{i-j} \cap \left(\cup_{\forall l \in \{N_j \setminus i\}} a_{j-l}\right) \tag{6.15}$$

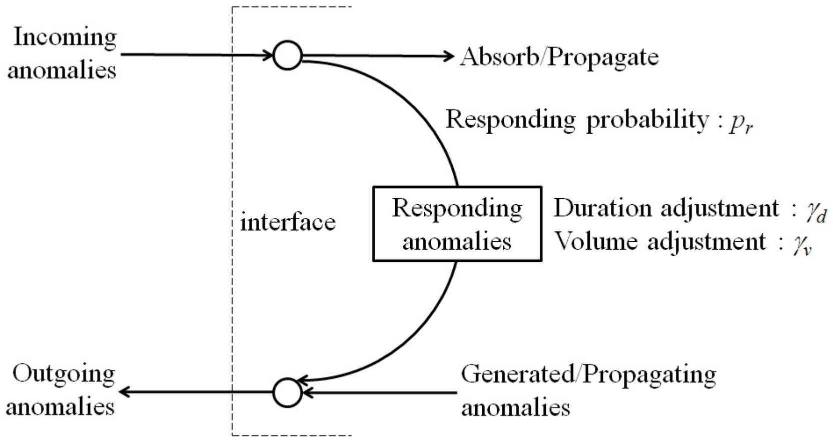
Then distribution parameters for ρ_v and ρ_d are tuned such that they best fit the volume adjustment factors and duration adjustment factors of the anomaly set q_{i-j}. To calculate



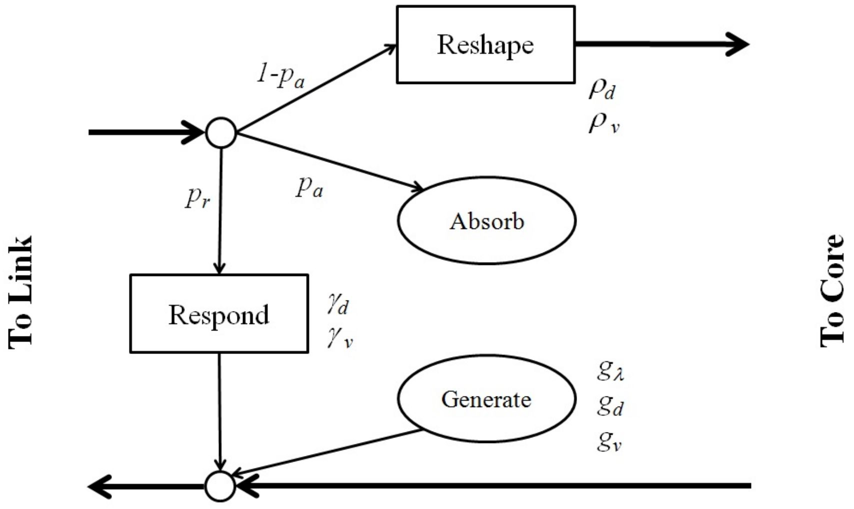
(A) Generation


Incoming anomalies
Reshape
Propagating anomalies
Duration adjustment: ρ_d
Volume adjustment : ρ_v
absorption probability : p_a
Absorb

(B) Absorption and propagation


Incoming anomalies
Absorb/Propagate
Responding probability : p_r
interface
Responding anomalies
Duration adjustment : γ_d
Volume adjustment : γ_v
Outgoing anomalies
Generated/Propagating anomalies

(C) Responding anomalies



(D) Complete interface model

FIGURE 6.20. Interface model.

the parameters of these two distributions using the ST-filter, we define two temporary sets L_ρ^v and L_ρ^d and use the relationships shown below:

$$L_\rho^v = \left\{ \frac{L_{i,J,|N_j|-1}^v(k)}{L_{i,J,0}^v(k)} \text{ s.t. } L_{i,J,|N_j|-1}^v(k) \neq \emptyset \right\} \tag{6.16}$$

$$L_\rho^d = \left\{ \frac{L_{i,J,|N_j|-1}^d(k)}{L_{i,J,0}^d(k)} \text{ s.t. } L_{i,J,|N_j|-1}^d(k) \neq \emptyset \right\} \tag{6.17}$$

The two sets L_ρ^v and L_ρ^d contain the fractional changes in volume and duration. Then two distributions ρ_v and ρ_d are selected to characterize the multiplicative factor of change in volume and duration. For example, if Gaussian distribution is found to be a good fit, it is characterized with the mean and standard deviation as follows:

$$\rho_{v,i-j} = \mathcal{N}\left(1 + \mathbb{E}L_\rho^v, stdev\left(L_\rho^v\right)\right) \rho_{d,i-j} = \mathcal{N}\left(1 + \mathbb{E}L_\rho^d, stdev\left(L_\rho^d\right)\right) \tag{6.18}$$

where $\mathbb{E}$ is the empirical expectation and $stdev(\cdot)$ is the standard deviation. The constraint $L_{i,J,|N_j|-1}^v(k) \neq \emptyset$ selects only the propagating anomalies. The two quantities $\mathbb{E}L_\rho^v$ and $\mathbb{E}L_\rho^d$ are mean fractional changes in volume and duration. By adding a one to them, as in $1 + \mathbb{E}L_\rho^v$, yields the mean multiplicative factor of change. To further clarify this using a simplistic example, let x be the input volume and y be the output volume. Then $(y-x)/x$ is the fractional change in volume and $1 + (y-x)/x = y/x$, is the multiplicative factor corresponding to the change.

Originating Anomalies: Nodes generate originating anomalies with volumes and durations drawn from distributions g_v and g_d with an inter-anomaly gap drawn from g_λ. To find these distributions, we observe the subset of originating anomalies n_{i-j} on node i's interface towards node j.

$$n_{i-j} = a_{i-j} \setminus \left(\cup_{\forall l \in \{N_j \setminus i\}} a_{l-j}\right) \tag{6.19}$$

Then find distributions g_v, g_d and g_λ that best fit volumes, durations and inter anomaly gaps of originating anomalies n_{i-j}. To estimate these distributions from ST-filter outputs, first we define temporary sets L_g^v, L_g^d, M and L_g^λ that contains volumes, durations, an index set, and inter-anomaly gaps of the originating anomalies. Here we define $iag(M)$ as the set of inter anomaly gaps of the index set M. Then the following relationships emerge.

$$L_g^v = \left\{ -L_{J,\{i\},1}^v(k) \text{ s.t. } L_{J,\{i\},0}^v(k) = 0 \right\} \tag{6.20}$$

$$L_g^d = \left\{ -L_{J,\{i\},1}^d(k) \text{ s.t. } L_{J,\{i\},0}^d(k) = 0 \right\} \tag{6.21}$$

$$M = \left\{ k \mid k = 1 \dots N_k \text{ s.t. } L_{J,\{i\},0}^v(k) = 0 \right\} \tag{6.22}$$

$$L_g^\lambda = \{iag(M)\} \tag{6.23}$$

The distributions that best fit to L_g^v, L_g^d and L_g^λ are selected for g_v, g_d and g_λ. For example, if they were Gaussian, exponential and exponential, respectively, the distributions will be specified with mean, standard deviation and rate as follows.

$$g_{v,i-j} = \mathcal{N}\left(\mathbb{E} L_g^v, stdev\left(L_g^v\right) \right) \tag{6.24}$$

$$g_{d,i-j} = Exp\left(\mathbb{E} L_g^d \right) \tag{6.25}$$

$$g_{\lambda,i-j} = Exp\left(\mathbb{E} L_g^\lambda \right) \tag{6.26}$$

Responding Anomalies: The probability at which responding anomalies appear is described with p_r. The responding volumes and durations are factors of the original anomaly drawn from distributions γ_v and γ_d. As above, we define the subset of responding anomalies

r_{i-j} on the node i's interface towards node j, to derive the parameters.

$$r_{i-j} = a_{i-j} \cap a_{j-i} \tag{6.27}$$

The probability of responding at node i's interface towards node j is:

$$p_{r,i-j} = \frac{|r_{i-j}|}{|a_{i-j}|} \tag{6.28}$$

The distributions γ_v and γ_d are selected to fit the volume and duration adjustment factors of the responding anomaly set r_{i-j}. As before, we define two temporary variables L_γ^v and L_γ^d that contain volumes and durations of responding anomalies, and use ST-filter outputs to find above parameters.

$$L_\gamma^v = \left\{ \frac{L_{i,i,1}^v(k)}{L_{i,i,0}^v(k)} \right\} \tag{6.29}$$

$$L_\gamma^d = \left\{ \frac{L_{i,i,1}^d(k)}{L_{i,i,0}^d(k)} \right\} \tag{6.30}$$

Distributions γ_v and γ_d are selected in a similar fashion as with propagating anomalies. If the selected distributions were Gaussian, they would be specified with mean and standard deviation as follows:

$$\gamma_{v,i-j} = \mathcal{N}\left(1 + \mathbb{E}L_\gamma^v, stdev\left(L_\gamma^v\right)\right) \tag{6.31}$$

$$\gamma_{d,i-j} = \mathcal{N}\left(1 + \mathbb{E}L_\gamma^d, stdev\left(L_\gamma^d\right)\right) \tag{6.32}$$

The probability values p_a, p_r and splitting ratios were estimated with sufficiently large sample sets N_T. Standard distributions are fitted for other parameters and verified using Kolmogorov-Smirnov (KS) test [154]. All the distribution fits satisfied the KS test with a 5%

significance level. This indicates that the selected anomaly properties can be well modelled with the chosen random processes. However, such a successful fit cannot be achieved for modelling the anomaly trace as a whole. This implies that the proposed anomaly model more accurately captures anomaly behaviours into common random processes.

6.3.3.3. *Modelling at Node Level.* Here, the model is applied to nodes in the Internet2 network (Fig. 6.16). To find the most suitable time frame, parameter values are estimated over increasing number of weeks. Parameter values converged when averaged over 10 weeks of data. In other words N_T needs to be at least 10 weeks for the parameter values to be stable. The results presented below are for $N_T = 50$ weeks. It was found that inter anomaly gap, duration and volume of originating anomalies are best described using exponential, exponential and Gaussian distributions respectively. Anomaly duration and volume change factors for both propagating and responding anomalies were well described using Gaussian distributions. The resulting parameters listed in Table 6.9, capture the statistical properties of anomalies at each node. With these results, now it is possible to evaluate and classify anomalies, and also to regenerate anomalies having statistical behaviours similar to nodes in Internet2 network, even for different topologies. Notably, only a limited number of parameters, measured locally, are required to successfully capture the statistical properties of anomalies over the entire network.

Table 6.9. Node Parameters for Internet2 Network

Node	Interface	p_a	p_r	Splitting ratio		μ for $g_\lambda \sim Exp(\mu)*$	μ for $g_d \sim Exp(\mu)$	(μ, σ^2) for $g_v \sim \mathcal{N}(\mu, \sigma^2)$	(μ, σ^2) for $\rho_d \sim \mathcal{N}(\mu, \sigma^2)$	(μ, σ^2) for $\rho_v \sim \mathcal{N}(\mu, \sigma^2)$	(μ, σ^2) for $\gamma_d \sim \mathcal{N}(\mu, \sigma^2)$	(μ, σ^2) for $\gamma_v \sim \mathcal{N}(\mu, \sigma^2)$
KSCY	DNVR	0.40	0.24	$\alpha_{IPLS} = 0.95$	$\alpha_{HSTN} = 0.15$	45	1.8	(29, 69)	(1.1, 1.2)	(1.0, 2.2)	(1.6, 1.5)	(1.2, 0.9)
	IPLS	0.32	0.14	$\alpha_{DNVR} = 0.91$	$\alpha_{HSTN} = 0.22$	42	1.7	(43, 95)	(1.3, 0.9)	(1.0, 0.6)	(1.2, 1.2)	(1.3, 1.3)
	HSTN	0.77	0.29	$\alpha_{IPLS} = 0.71$	$\alpha_{DNVR} = 0.65$	34	1.8	(4.6, 25)	(1.7, 2.5)	(5.4, 58.0)	(1.1, 0.8)	(1.4, 2.6)
HSTN	KSCY	0.73	0.32	$\alpha_{LOSA} = 0.90$	$\alpha_{ATLA} = 0.86$	38	1.5	(7.7, 26)	(2.0, 3.1)	(4.0, 32.0)	(1.2, 1.0)	(0.6, 2.5)
	LOSA	0.38	0.33	$\alpha_{KSCY} = 0.43$	$\alpha_{ATLA} = 0.93$	43	1.9	(2.7, 28)	(1.2, 0.6)	(0.8, 16.0)	(1.3, 1.5)	(1.1, 48.0)
	ATLA	0.49	0.28	$\alpha_{KSCY} = 0.31$	$\alpha_{LOSA} = 0.98$	29	1.7	(20, 41)	(1.2, 0.8)	(1.3, 1.5)	(1.2, 1.1)	(1.1, 2.1)
DNVR	STTL	0.65	0.22	$\alpha_{SNVA} = 0.24$	$\alpha_{KSCY} = 0.93$	42	1.9	(5.2, 66)	(1.0, 1.0)	(1.6, 13.0)	(1.5, 1.5)	(1.8, 2.1)
	SNVA	0.51	0.31	$\alpha_{STTL} = 0.09$	$\alpha_{KSCY} = 0.98$	24	1.6	(17, 46)	(1.1, 0.9)	(1.5, 6.7)	(1.4, 1.4)	(1.2, 2.7)
	KSCY	0.23	0.16	$\alpha_{STTL} = 0.33$	$\alpha_{SNVA} = 0.74$	38	1.6	(42, 220)	(1.6, 1.5)	(1.1, 1.7)	(1.0, 0.5)	(0.6, 10.0)
IPLS	KSCY	0.30	0.18	$\alpha_{CHIN} = 0.98$	$\alpha_{ATLA} = 0.98$	42	1.6	(37, 99)	(1.1, 0.6)	(1.2, 0.6)	(1.2, 0.8)	(1.3, 1.8)
	CHIN	0.24	0.19	$\alpha_{KSCY} = 0.89$	$\alpha_{ATLA} = 0.18$	44	1.8	(38, 130)	(1.2, 1.1)	(1.1, 6.0)	(1.4, 1.8)	(1.5, 14.0)
	ATLA	0.74	0.26	$\alpha_{KSCY} = 0.57$	$\alpha_{CHIN} = 0.71$	37	1.9	(11, 44)	(1.5, 1.8)	(4.8, 12.0)	(1.1, 0.6)	(1.1, 1.0)
ATLA	WASH	0.48	0.32	$\alpha_{IPLS} = 0.20$	$\alpha_{WASH} = 0.94$	43	1.9	(23, 66)	(1.3, 1.2)	(0.9, 2.8)	(1.1, 0.6)	(1.0, 2.1)
	IPLS	0.86	0.21	$\alpha_{WASH} = 0.91$	$\alpha_{HSTN} = 0.66$	46	1.7	(9.6, 31)	(1.6, 1.2)	(0.9, 5.4)	(1.6, 3.2)	(1.3, 1.2)
	HSTN	0.47	0.30	$\alpha_{WASH} = 0.99$	$\alpha_{IPLS} = 0.17$	36	1.8	(15, 39)	(1.1, 0.7)	(2.1, 5.7)	(1.2, 1.1)	(1.3, 1.9)
SNVA	STTL	0.84	0.12	$\alpha_{DNVR} = 0.51$	$\alpha_{LOSA} = 0.79$	36	1.9	(5.2, 17)	(1.8, 2.4)	(0.9, 21.0)	(1.0, 0.5)	(3.0, 20.0)
	DNVR	0.25	0.22	$\alpha_{STTL} = 0.10$	$\alpha_{LOSA} = 0.97$	53	1.9	(25, 49)	(1.2, 0.6)	(1.0, 4.8)	(1.4, 1.5)	(1.4, 1.5)
	LOSA	0.30	0.28	$\alpha_{STTL} = 0.09$	$\alpha_{DNVR} = 0.95$	16	1.2	(11, 24)	(1.1, 0.5)	(1.1, 1.9)	(1.4, 1.8)	(1.1, 2.4)
STTL	SNVA	0.93	0.13	$\alpha_{DNVR} = 1.0$		30	2.0	(7.5, 16)	(1.3, 0.9)	(5.3, 200.0)	(1.2, 0.6)	(4.6, 110.0)
	DNVR	0.92	0.19	$\alpha_{SNVA} = 1.0$		43	2.4	(44, 95)	(2.8, 5.3)	(0.5, 3.2)	(1.4, 1.8)	(1.1, 2.3)
LOSA	SNVA	0.87	0.20	$\alpha_{HSTN} = 1.0$		22	2.1	(45, 80)	(2.0, 3.0)	(1.3, 5.9)	(1.3, 1.1)	(0.4, 2.5)
	HSTN	0.82	0.31	$\alpha_{SNVA} = 1.0$		26	2.2	(22, 55)	(1.1, 1.3)	(5.5, 2.8)	(1.2, 1.1)	(0.6, 5.2)
WASH	ATLA	0.76	0.32	$\alpha_{NYCM} = 1.0$		30	2.3	(29, 72)	(0.9, 0.4)	(0.3, 3.6)	(1.2, 0.7)	(1.6, 3.2)
	NYCM	0.83	0.10	$\alpha_{ATLA} = 1.0$		50	2.2	(56, 82)	(1.5, 1.0)	(0.3, 1.6)	(1.0, 0.5)	(1.1, 3.4)
NYCM	WASH	0.75	0.13	$\alpha_{CHIN} = 1.0$		40	2.4	(69, 130)	(1.2, 1.3)	(1.2, 20.0)	(1.4, 1.3)	(1.0, 1.0)
	CHIN	0.61	0.17	$\alpha_{WASH} = 1.0$		22	2.0	(52, 120)	(1.3, 1.1)	(0.8, 0.8)	(1.5, 1.3)	(1.3, 1.7)
CHIN	NYCM	0.59	0.12	$\alpha_{IPLS} = 1.0$		33	2.1	(69, 130)	(1.1, 1.0)	(1.5, 5.1)	(1.3, 1.6)	(1.2, 3.0)
	IPLS	0.74	0.18	$\alpha_{NYCM} = 1.0$		29	1.8	(67, 230)	(1.3, 0.9)	(1.5, 2.7)	(1.3, 1.1)	(2.1, 2.1)

* g_λ is expressed in hours, as oppose to sample periods. Mean μ and standard deviation σ^2 are expressed in 1000's.

These results are useful in many aspects. The model parameters reveal illuminating information about the network. For instance, model parameters indicate that anomalies originating at the Washington interface of New York node have significantly large volumes, on average 68870 pkts/sec. Another example is most of anomalies, in fact 98.2%, entering the Indianapolis node from the Kansas City interface propagate to Chicago node. A more important aspect however is that the model parameters are useful for design of other similar networks. Based on the information as above, large and sophisticated networks can be calibrated. When a new network is designed, the model parameters of representative nodes from an analyzed network can be used to provision capacities needed to withstand possible network anomalies.

6.3.4. Modelling at Subnet Level. Similar to anomaly activities at and between nodes, anomaly activities at subnet level are also important for ISPs and backbone service providers. Subnet level anomalies correspond to anomalous behaviours at higher abstractions of the network. We observe that the above proposed anomaly model captures the anomaly behaviour even at subnet levels. Approaches to extend the nodal model over paths and regions were discussed in [7]. Though the same model captures the nodal level and subnet level anomaly behaviours, we have not yet been able to derive a rigorous derivation of the subnet level model parameters from the nodal level model parameters due to a number of limitations.

6.3.4.1. *Limitations on Extending Nodal Model to Regional Model.* Given the model parameter values of a set of nodes does not permit deriving the model parameter values for the region made out of those nodes, due to a number of reasons. Despite node level characterization provides a satisfactory estimate on global behaviours, information is insufficient to aggregate nodes. We identify key limitations preventing aggregating nodal models into regional models, below:

Nodal model use only local information: All the anomalies arriving and leaving a node are used to calibrate nodal parameters without a distinction. Thus, distinctive properties of anomalies, such as: the specific origin, path, changes in volumes and durations as they travel, are not captured. Not having a classification of anomalies may raise issues in aggregating nodes, as parameters for different groups of anomalies may not be the same. Short range anomalies may not be of interest when regions are formulated, as they become internal behaviours. Anomaly properties of long range anomalies would govern propagation properties across a region. But such differences are not taken into account as all anomalies are treated equally when deriving nodal parameters. The properties based on the origin of the anomalies play a key role. For example, most anomalies a certain node receives may be from another specific node. If such a pair of nodes aggregated, most anomalies become internal and regional properties will change significantly. Since origin information is not recorded, the model will not carry such pieces of information, despite the key role they play during aggregation. However, representing such behaviours will require more details stored.

Rules of aggregations not understood: Another critical issue in network node aggregation is that there are no understood rules, unlike in, say, circuit theory.

(1) No conservation theories: The anomalous throughput entering and leaving an entity, and the amount generated and absorbed do not form a conservation of traffic. However, the count of anomalies is treated conserved during development of the anomaly model, i.e., anomalies that did not propagate treated absorbed.
(2) No simplification theories: A key concern is whether two nodes required to be directly linked to be aggregated. With the assumption of no such restriction, a number

of primitive simplification modes can be thought of, as shown below. However, aggregating any of these modes is not understood.

- Two nodes connected with a direct link (in series)
- Two nodes with no direct link between them, but connects to a common node (in parallel)
- A loop of nodes (ring)
- A random scatter of nodes with no condition of direct links or common nodes

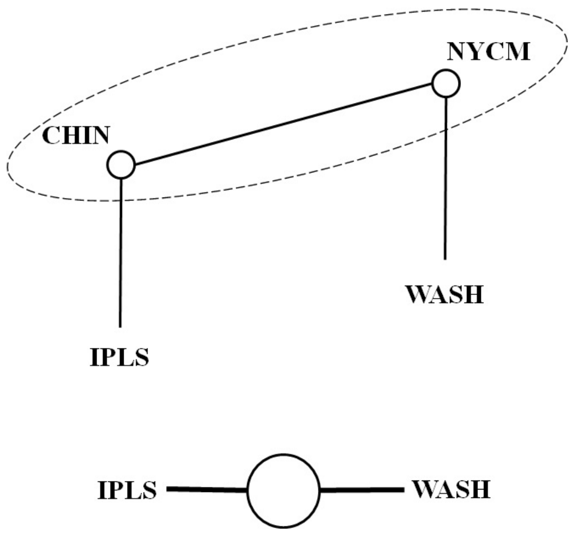
(3) No rules on super-imposing links: Links may collapse onto each other when aggregating nodes. An example scenario is when two nodes connected to a third are merged. The aggregated node will have two links running to the same third node that need to be merged. Conditions and constraints on super-imposing these two links are not understood.

6.3.4.2. *Region parameters.* Here we present the anomaly statistics of a set of subnets (Fig. 6.21) modelled with the proposed anomaly model. The key characteristics of the chosen subnets are summarised in Table 6.10. The model parameters for these subnets are listed in Table 6.11. Similar to the node level analysis, distributions are chosen such that they best capture the observed properties. Similar to the node level statistics, interesting features are revealed by the model parameters. For example, over 96% of anomalies entered the subnet (d) from the Denver interface propagated to the Los Angeles interface. Here we note that the parameters that are described using distributions, fitted to the same type of distributions as the nodal model did. For example the inter anomaly gap was still well described with an exponential distribution; volume and duration changes of propagating and responding anomalies are well described with Gaussian distributions. Being able to characterize anomaly behaviours of different subnets with the same model used for nodal

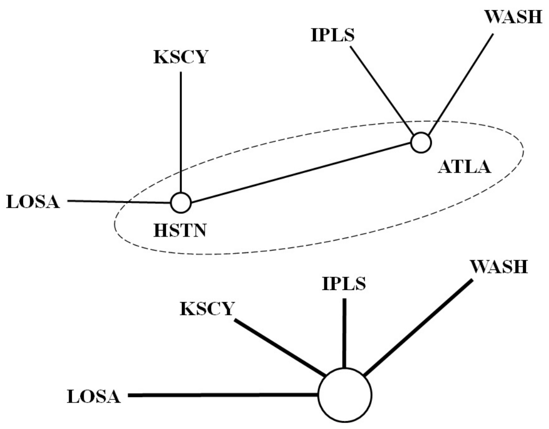
Table 6.10. Description of the Subnets

Subnet	Description
a	Two nodes of degree-2 in series; subnet is degree-2
b	Two nodes of degree-3 in series; subnet is degree-4
c	Two connected nodes in parallel
d	Two unconnected nodes in parallel
e	A ring of three nodes
f	A ring of four nodes

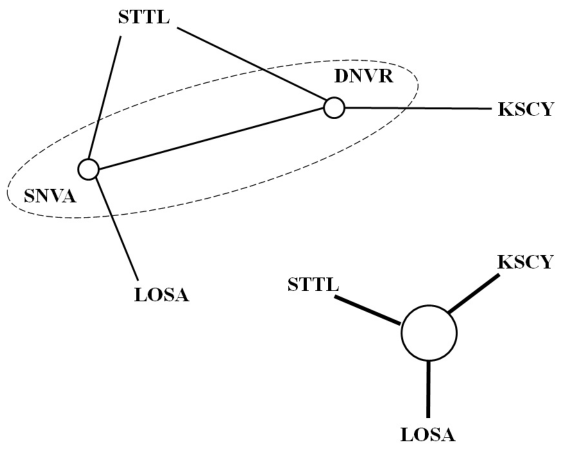
level characterization implies that the proposed model can capture the network anomaly behaviour at different levels of granularity.



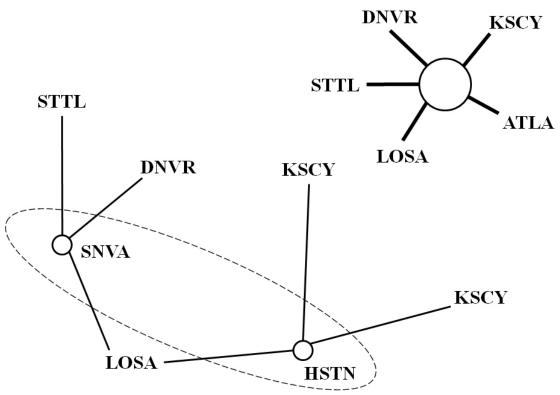
(A) Chicago - New York



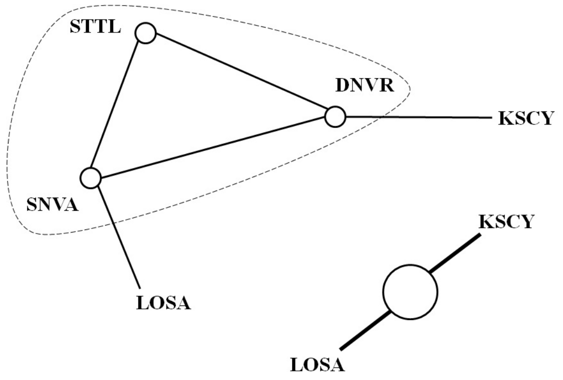
(B) Houston - Atlanta



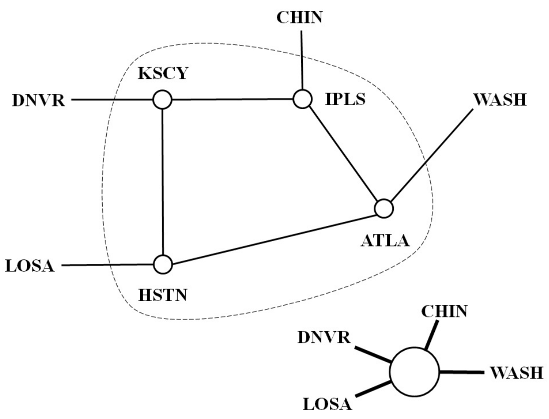
(C) Sunny Valley - Denver



(D) Sunny Valley - Houston



(E) Seattle - Denver - Sunny Valley



(F) Kansas City - Indianapolis - Atlanta - Houston

Figure 6.21. Sample subnets aggregated.

TABLE 6.11. Model Parameters for Selected Set of Subnets

Subnet	Interface	p_a	p_r	Splitting ratio				μ for $g_\lambda \sim Exp(\mu)*$	μ for $g_d \sim Exp(\mu)$	(μ, σ^2) for $g_v \sim \mathcal{N}(\mu, \sigma^2)$	(μ, σ^2) for $\rho_d \sim \mathcal{N}(\mu, \sigma^2)$	(μ, σ^2) for $\rho_v \sim \mathcal{N}(\mu, \sigma^2)$	(μ, σ^2) for $\gamma_d \sim \mathcal{N}(\mu, \sigma^2)$	(μ, σ^2) for $\gamma_v \sim \mathcal{N}(\mu, \sigma^2)$
a	IPLS	0.90	0.18	$\alpha_{WASH} = 1.0$				24	2.0	(64, 220)	(1.4, 1.2)	(0.7, 1.0)	(1.3, 1.1)	(2.1, 19.0)
	WASH	0.82	0.11	$\alpha_{IPLS} = 1.0$				44	2.7	(72, 130)	(1.4, 1.9)	(3.1, 7.6)	(1.5, 1.3)	(1.0, 1.3)
b	LOSA	0.43	0.53	$\alpha_{KSCY} = 0.56$	$\alpha_{IPLS} = 0.22$	$\alpha_{WASH} = 0.86$		80	2.0	(14, 48)	(1.2, 0.8)	(0.7, 6.7)	(1.3, 1.5)	(1.1, 9.9)
	KSCY	0.70	0.28	$\alpha_{LOSA} = 0.79$	$\alpha_{IPLS} = 0.32$	$\alpha_{WASH} = 0.77$		42	1.5	(7.1, 25)	(2.1, 3.1)	(2.8, 23.0)	(1.2, 1.0)	(0.7, 3.6)
	IPLS	0.78	0.25	$\alpha_{LOSA} = 0.60$	$\alpha_{KSCY} = 0.54$	$\alpha_{WASH} = 0.80$		41	1.8	(9.4, 32)	(1.9, 2.5)	(1.3, 12.0)	(1.6, 3.2)	(1.3, 0.9)
	WASH	0.48	0.35	$\alpha_{LOSA} = 0.89$	$\alpha_{KSCY} = 0.37$	$\alpha_{IPLS} = 0.24$		47	2.1	(34, 72)	(1.4, 1.4)	(1.0, 2.2)	(1.1, 0.6)	(1.0, 2.4)
c	STTL	0.61	0.22	$\alpha_{KSCY} = 0.76$	$\alpha_{LOSA} = 0.42$			37	1.9	(4.7, 55)	(1.2, 1.6)	(2.3, 9.5)	(1.4, 1.5)	(2.7, 10.0)
	KSCY	0.28	0.25	$\alpha_{STTL} = 0.39$	$\alpha_{LOSA} = 0.70$			68	1.7	(42, 89)	(1.6, 1.7)	0.8, 2.4)	(1.0, 0.6)	(0.6, 3.1)
	LOSA	0.44	0.23	$\alpha_{STTL} = 0.15$	$\alpha_{KSCY} = 0.93$			45	1.5	(15, 33)	(1.2, 1.2)	(1.4, 7.7)	(1.4, 1.8)	(1.1, 2.0)
d	LOSA	0.20	0.27	$\alpha_{STTL} = 0.10$	$\alpha_{DNVR} = 0.77$	$\alpha_{KSCY} = 0.15$	$\alpha_{ATLA} = 0.31$	51	1.4	(9.3, 26)	(1.1, 0.6)	(2.0, 28.0)	(1.4, 1.8)	(0.2, 15.0)
	STTL	0.40	0.35	$\alpha_{LOSA} = 0.85$	$\alpha_{DNVR} = 0.46$	$\alpha_{KSCY} = 0.25$	$\alpha_{ATLA} = 0.32$	120	1.6	(6.9, 78)	(2.3, 2.7)	(9.6, 50.0)	(1.0, 0.5)	(3.0, 33.0)
	DNVR	0.14	0.39	$\alpha_{STTL} = 0.10$	$\alpha_{LOSA} = 0.97$	$\alpha_{KSCY} = 0.21$	$\alpha_{ATLA} = 0.30$	150	1.9	(30, 48)	(1.4, 0.9)	(0.5, 2.1)	(1.4, 1.5)	(1.4, 2.8)
	KSCY	0.68	0.28	$\alpha_{STTL} = 0.22$	$\alpha_{DNVR} = 0.58$	$\alpha_{LOSA} = 0.80$	$\alpha_{ATLA} = 0.70$	45	1.5	(7.8, 27)	(2.3, 3.6)	(3.9, 20.0)	(1.2, 1.0)	(0.6, 3.6)
	ATLA	0.37	0.32	$\alpha_{STTL} = 0.12$	$\alpha_{DNVR} = 0.29$	$\alpha_{KSCY} = 0.30$	$\alpha_{LOSA} = 0.97$	46	1.6	(20, 41)	(1.4, 1.2)	(0.6, 3.1)	(1.2, 1.1)	(1.1, 2.2)
e	LOSA	0.47	0.22	$\alpha_{KSCY} = 1.0$				38	1.7	(18, 46)	(1.0, 0.9)	(1.4, 7.8)	(1.4, 1.8)	(1.1, 2.0)
	KSCY	0.50	0.25	$\alpha_{LOSA} = 1.0$				45	2.0	(67, 130)	(1.4, 1.5)	(0.6, 2.8)	(1.0, 0.6)	(0.6, 3.1)
f	LOSA	0.45	0.53	$\alpha_{DNVR} = 0.48$	$\alpha_{CHIN} = 0.36$	$\alpha_{WASH} = 0.89$		80	2.0	(15, 48)	(1.3, 1.2)	(3.0, 31.0)	(1.3, 1.5)	(1.1, 9.9)
	DNVR	0.31	0.17	$\alpha_{LOSA} = 0.18$	$\alpha_{CHIN} = 0.90$	$\alpha_{WASH} = 0.18$		66	1.9	(39, 79)	(1.3, 1.3)	(0.7, 1.2)	(1.6, 1.5)	(1.2, 1.1)
	CHIN	0.66	0.20	$\alpha_{DNVR} = 0.92$	$\alpha_{LOSA} = 0.29$	$\alpha_{WASH} = 0.27$		51	1.9	(47, 120)	(1.7, 2.5)	(0.8, 1.6)	(1.4, 1.8)	(1.5, 2.1)
	WASH	0.48	0.35	$\alpha_{DNVR} = 0.41$	$\alpha_{CHIN} = 0.25$	$\alpha_{LOSA} = 0.90$		46	1.9	(29, 53)	(1.3, 1.0)	(1.1, 8.3)	(1.1, 0.6)	(1.0, 2.4)

* g_λ is expressed in hours, as oppose to sample periods. Mean μ and standard deviation σ^2 are expressed in 1000's.

6.3.4.3. *Real-Time Monitoring and Regional Parameter Learning.* A scheme using Graph Wavelet Transform (GWT) [149] coefficients to share anomaly information over the network is proposed in [7]. Spatial scales of graph wavelets are defined along the connections in a graph. Here, the scheme from [7] is implemented with ST-filter shown in Fig. 6.22. MIB (Management Information Base) on each router records traffic samples and summarises anomaly activities. Then the ST-filter will produce the set of outputs of interest. By combining these filter outputs from the different neighbours, a description of anomalies in the local network can be obtained. The choice of exchanged information and the direction of information propagation would depend on the application. We consider two applications: (1) monitoring anomaly activity in the network, and (2) learning regional subnet model parameters.

For monitoring purposes, two sets of outputs $L^v_{i,J,1}(k)$ and $L^d_{i,J,1}(k)$, where $J = Nj \setminus i$, are passed down the direction of anomalies, similar to [7]. Here a node j formulates outputs for all its interface i's that experience anomalies.

Based on the ST-filter outputs used in Section 6.3.3 to estimate model parameters, a specific set of outputs are shared for distributed learning of regional model parameters. Since anomaly activity on nodes internal to the region play no role in the model, only the boundary nodes share a selected set of ST-filter outputs. If each boundary node share $L^v_{I,J,|J|}(k)$, $L^d_{I,J,|J|}(k)$, $L^v_{J,I,|I|}(k)$ and $L^d_{J,I,|I|}(k)$, with all the other boundary nodes, regional parameters can be learned distributively. Here I is the set of interfaces internal to the region and J is the set of interfaces external to the region. Calculation of splitting ratio, absorption probability, and volume adjustments for propagating anomalies will require $L^v_{I,J,|J|}(k)$ from the other boundary nodes. Similarly, duration adjustment for propagating anomalies requires $L^d_{I,J,|J|}(k)$ from all boundary nodes. To find properties of originating anomalies, $L^v_{J,I,|I|}(k)$

```
function MAIN
    Set Timer to sampling time
    At expiry
        Sample all links
        Form $traffictimeseries$
        ANOMALY DETECTOR($traffictimeseries$)
        ST FILTER($anomalytimeseries$)
        EXCHANGE INFORMATION($filteroutputs$)
end function

function ST FILTER($anomalytimeseries$)
    Form summaries
    Do ST Filtering
return $filteroutputs$
end function

function EXCHANGE INFORMATION($filteroutputs$)
    Send $filteroutputs$ to neighbors
    Receive $filteroutputs$ from neighbors
    Forward $L^v_{i,J,1}$ and $L^d_{i,J,1}$ in direction of anomalies
    if $boundarynode$ then
        Forward $L^v_{I,J,|J|}$, $L^d_{I,J,|J|}$, $L^v_{J,I,|I|}$ and $L^d_{J,I,|I|}$ to other boundary nodes
    end if
end function
```

FIGURE 6.22. Algorithm for real-time anomaly monitoring system

and $L^d_{J,I,|I|}(k)$ are required. Properties of responding anomalies are calculated locally without any information from the region. Thus, the above set is sufficient to compute all region model parameters.

6.3.5. CONCLUSIONS AND FUTURE WORK. A model that successfully characterizes Internet traffic anomalies by characterizing the different anomaly features and integrating them to obtain anomaly characteristics was proposed. The proposed model characterizes various spatiotemporal properties of anomaly behaviours with separate random models. As shown by KS-test based verifications, this approach accurately captures anomaly behaviours. The model parameters were derived using the outputs of a multi-scale anomaly analysis framework called the spatial-temporal filter. It analyzes various duration and volume properties of anomalies at difference scales. Compared to graph wavelets, the proposed spatial-temporal filter is more flexible in extracting anomaly properties. Further, the model parameters for Internet2 nodes and a number of subnets thereof were derived and validated. These results reveal interesting features of the Internet2 network, and more importantly provides information for designing future networks. The model is shown to be viable for anomaly characterization at nodal level and at subnet level, enabling hierarchical analysis of large and sophisticated networks. A real-time monitoring system was proposed and used for learning regional model parameters.

Future extensions of the model could add second order information by means such as covariances between model parameters, and elaborate the current simpler features. For example, the parameters described with probability values, such as probability of absorption or response, can be converted into distributions over the distance to the considered node. As the proposed model facilitates re-synthesizing statistically similar anomalies, an anomaly simulator can be developed using the model. Such a simulator can be used along with a typical traffic generator to produce more realistic traffic traces.

6.3.6. Appendix - Fourier Analysis Based Anomaly Extraction Scheme.

Network data is viewed as a combination of the following: (1) baseline behaviour that accounts for traffic trends, e.g., weekly and daily trends, and load variations, (2) noise like random fluctuations superimposed on baseline that account for normal traffic variations, and (3) anomalies that significantly deviate from tolerable traffic variations. The aim here is to extract the anomaly component of a traffic trace. The proposed scheme is capable of overcoming the major periodic fluctuations of traffic that naturally occur daily and weekly. The traffic patterns exhibit clear weekly trends that are consistent over time, and internal to weekly patterns one can also observe more regular periodic behaviours, such as diurnal patterns. Due to this periodicity, a Fourier analysis based scheme, described below, is employed to extract anomalies [7]. Other Fourier analysis based anomaly detections and characterizations such as in [138] operate on a few frequency bands. But the approach employed in this work operates on individual frequency components, as explained below.

The anomaly extraction scheme is described in Fig. 6.23. The algorithm takes in a time-series of network traffic and outputs a time-series containing only the anomalies. The process first de-trends the time-series by removing the significant Fourier components. The implication is that the prominent Fourier coefficients capture the baseline trend of the time-series. A weekly measurement trace has 2016 samples, collected every five minutes. The baseline component of this 2016 point time-series is found to lie in the largest 20 Fourier coefficients. Thus, 20 Fourier coefficients with largest magnitude are set to zero for de-trending. The number 20 has been established by studying the energy distribution in the spectrum of traffic traces. The Fourier components are also selected symmetrically, to maintain realness of the de-trended data. For a different time-frame or for a different network, Fourier coefficients would be selected to capture over 80% of the energy. The severity sought in the deviating samples to be labelled as anomalous is controlled by a tunable threshold. Typical settings for threshold are in the range of two to three times the standard deviation of the de-trended time-series.

Figure 6.24 demonstrates the stages of anomaly detection using a week's worth of data. The baseline estimated with the 20 most significant Fourier coefficients is shown in Fig. 6.24(a) plotted over the raw data. Once the baseline is removed, anomalies become more apparent and then a threshold is applied as shown in Fig. 6.24(b) to separate anomalies. The amount over the threshold is marked as anomalous as shown in Fig. 6.24(c). As can be

```
function ANOMALY DETECTOR($trafficimeseries$)
    $spectrum \leftarrow$ FFT($traffictimeseries$)
    $blI \leftarrow$ index set of largest Fourier coefficients
    $spectrum[blI] \leftarrow 0$
    $anomalytimeseries \leftarrow$ iFFT($spectrum$)
    for all element $k$ do
        if $|anomalytimeseries[k]| < threshold$ then
            $anomalytimeseries[k] \leftarrow 0$
        end if
    end for
return $anomalytimeseries$
end function
```

FIGURE 6.23. Anomaly detection algorithm

noted, Anomaly 2 and Anomaly 3 have peaks below the traffic trend and they would not have been detected using a threshold only approach. Two temporal properties, the anomaly duration and the inter anomaly gap are also identified in Fig. 6.24(c).

6.4. CONCLUSIONS

Network traffic anomalies are modelled component-wise to overcome the limitations posed by their sporadic behaviors. The presented model concisely capture the anomaly behavior with a limited set of parameters. The anomaly behavior of Internet2 network is closely described with a table of 10 columns. Furthermore, the model parameters can be calculated locally. But they do describe the global anomaly behavior. The anomaly description produced by the model is used for two real-time applications: (1) real-time computation of regional anomaly behavior, and (2) real-time communication of anomaly propagation for intelligent anomaly tracking.

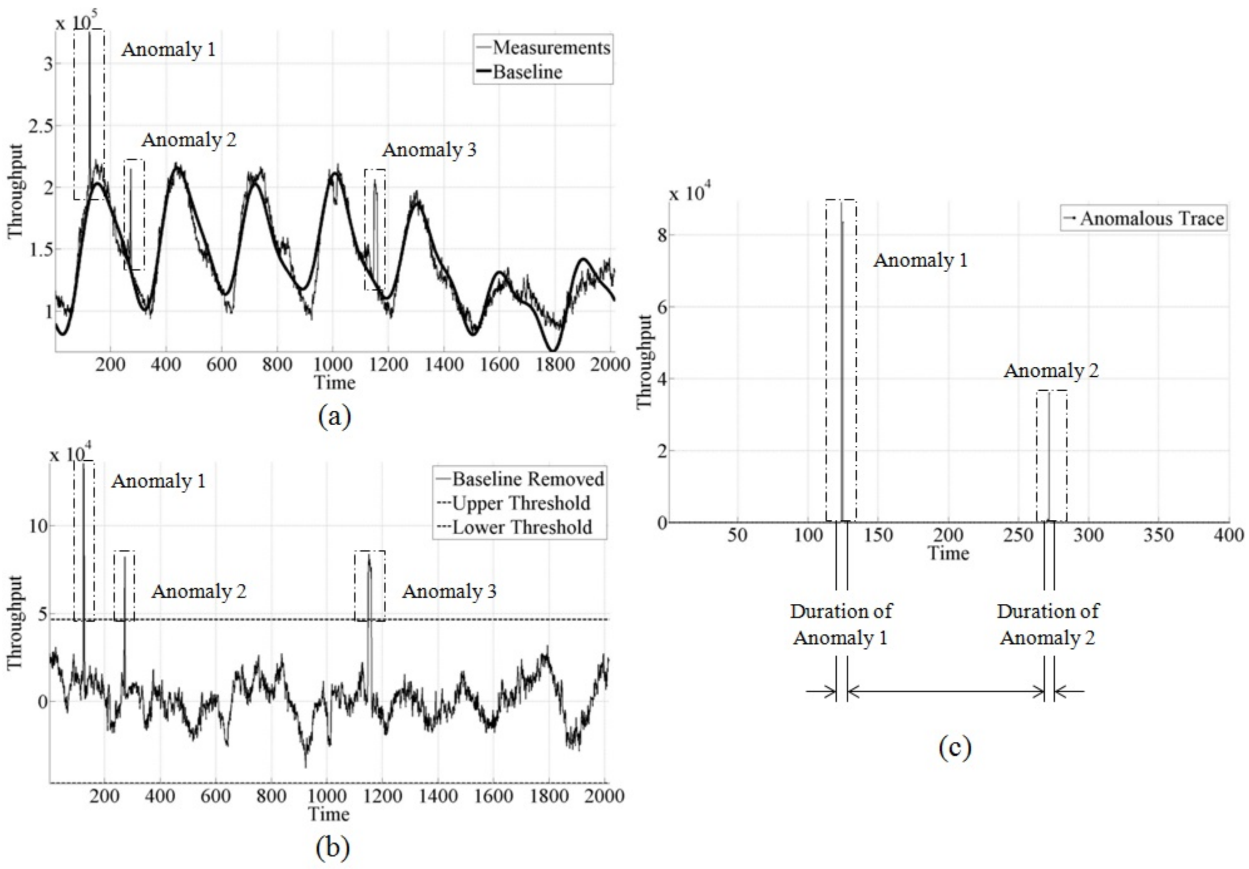


FIGURE 6.24. Anomaly extraction process: (a) Anomalies indicated on a trace of raw data, (b) Anomalies indicated over de-trended data, and (c) The extracted trace of anomalies.

CHAPTER 7

Network Traffic Baseline

Robust BaseLine (RBL) is a formal technique for extracting the baseline of network traffic to capture the underlying traffic trend. A range of applications such as anomaly detection and load balancing rely on baseline estimation. Once the fundamental period of the pattern for analysis is recognized, e.g., based on user interest or a period detector such as Autocorrelation Function (ACF), the basic extraction is carried out in two steps. First, the common component across the dataset is separated using Robust Principal Component Analysis (RPCA). The fundamental pattern in the common component is extracted using Principal Component Analysis (PCA) in the second step. Scaling factors required to fit the base-pattern back into the data are returned automatically by PCA. Two types of traffic baselines may be extracted: RBL-L captures the common behavior across time on a single link, and RBL-N captures the common behavior across a network of links, i.e., in space. RBL-N is particularly useful for specifying traffic matrices more efficiently over time, which normally requires multiple updates to follow baseline trends. The derived base-patterns for a single link or a single time period is then extended over the entire network or through the entire observation period with a compressive analysis. The compressed base-pattern provides a smoother baseline and also a filter to separate baseline traffic and the deviations on the fly from traffic measurements. When compared against BLGBA (Baseline for Automatic Backbone Management) the proposed scheme provides a less noisy, more precisely fitting baseline. It is also more effective in revealing anomalies.

7.1. Introduction

Trends in traffic such as peaks during busy hours and valleys during inert hours are natural occurrences. Traffic baselines represent such general trends. These baselines, which often are repetitive and perhaps deterministic, carry a large fraction of information about the traffic and play a vital role in traffic engineering, network design, load balancing and pricing. Extracting the baseline behavior from a traffic trace is a subjective task, based on how the baseline is perceived.

The fundamental structure of the baseline is termed "baseline pattern." The baseline of a traffic trace may be viewed as a series of scaled baseline patterns. The baseline pattern (or base-pattern) captures the following ideas: (1) it represents a segment that cannot be broken down to smaller similar segments, (2) it is repeated persistently on a trace, and (3) it contains most of the energy of the signal. The base-pattern in effect captures the most prominent features of the traffic trace such as modes, trends and gradients. Having a simple and compact representation for the base-pattern is useful for applications such as characterization of network traffic in terms of traffic matrices, which otherwise would require frequent updates.

7.1.1. Contribution.

This dissertation develops a novel formal scheme for extracting a Robust Base-line (RBL) of a traffic trace. Given a traffic trace, the scheme returns the most common and prominent base-pattern in the dataset, along with the scaling coefficients to construct the baseline. The scheme is developed formalizing features an expert would perceive as constituents of a baseline: a common, prominent and perhaps smooth extraction for the data trace. The novelty of the work lies here. Mathematical tools are applied in order to realize the perceptions of a baseline. The scheme employs Robust Principal Component

Analysis (RPCA) [5], a technique that recently has received much interest for separating the common component, i.e., the low rank component, across the dataset. The most salient component comprised of significant principal components in the common component is then extracted using classical PCA. Following that, a compressed representation for the base-pattern is proposed. The compressed representation builds a smoother version of the baseline. Furthermore, compressed representation is also used to build a filter to separate baseline from traffic intensity in real-time.

The formal scheme presented is dataset independent. The scheme operates on a data matrix which could either be multiple time windows on a single link or a data for single time window on multiple links. In the former arrangement, the baseline behavior over a link across time is found, and referred to as RBL-Link (RBL-L). While the latter will deliver the baseline behavior across the network over the considered time window, and referred to as RBL-Network (RBL-N). The scheme is also capable of revealing more subtle patterns that are buried in noisy data segments. The tunable parameters and the optimal settings for each parameter are also discussed.

7.1.2. Related Work. The importance of traffic characterization is emphasized in [170]. However, the widely used random-process based traffic models overlook deterministic baseline behaviors [171]. An extensive survey of traffic identification can be found in [172]. Lack of a proper definition of a baseline has challenged traffic characterization attempts. A simulated network is used in [173] for the baseline characterization in a tactical security architecture. In much of the literature, the baseline behavior of traffic is mostly characterized rather than being extracted. The difference here is that, characterization is not constructive, i.e., the returned properties are not sufficient to build a baseline trace, whereas extraction filters out the baseline trace from the data trace. A more simplistic approach in [174] uses average and variance to characterize traffic, and uses daily variation to account for dynamics. Such approaches are convenient for implementation. The characterization proposed in [174] is tuned for QoS routing. A statistical approach in [175] uses marginal and multi-variate histograms of traffic features to characterize baseline behavior. Use of Principal Component Analysis (PCA) for classifying baseline and anomalous traffic is discussed in [176]. In [177], entropy based clustering is used on a five-tuple characterization (source and destination; IP and port; and the protocol). This work is further extended in [178] by developing a real time traffic filter. An entropy based profiling scheme for attack detection is presented in [179]. A Hidden Markov Model (HMM) is used in [180], which also presents a good survey emphasizing the need of an effective traffic model. The approach in [181] uses a Gaussian mixture model for baselining network traffic. Some models are driven by the nature of traffic, such as burstiness and self-similarity [4]. An alternative is to use a token bucket scheme to meter bursty traffic traces [182]. A seven-tuple characterization in [183] uses a self-similar model parameterized with the minimum, the maximum and the degree of self-similarity using the Hurst exponent. In a more cross field approach to classify network traffic, Grey Level Co-occurrence Matrices (GLCM) are used in [184]. Here, the idea is to interpret the nature of traffic as the texture of an image. BLGBA proposed in [185] serves as the baseline scheme for GBA tool (Gerenciamento de Backbone Automatizado : Automatic Backbone Management). Two types of baseline sets were used in BLGBA: a set labeled bl-7 having separate baseline for each day of the week, and a set labeled bl-3 having a baseline for the week days, one for Saturday and one for Sunday. As an extension, [186] uses BLGBA based baseline and k-means clustering for anomaly detection.

Rest of the section is as follows. Section 7.2 explains the theoretical basis related to separating the base-pattern from a data trace. The derived scheme is applied to real data and the results are shown in Section 7.3. Section 7.4 addresses some applications of the proposed scheme. Section 7.5 provides a discussion on the scheme and concludes the section.

7.2. Scheme for Base-Pattern Separation

This section explains the formal scheme used to separate base-patterns from traffic patterns. The base-pattern is expected to capture a significant fraction of the traffic behavior, and stand as a good representation for the traffic trend. Therefore the base-pattern has to be (1) always present in the trace, (2) common to all links, (3) a prominent component in the trace, and preferably (4) has a compact representation. Below we consider two arrangements for data; one extracting baseline behavior over time and the other over space - the network. The following are discussed in this section:

(1) Finding an optimal period to break time-series

(2) RPCA based common component separation

(3) PCA based salient component extraction

(4) Compressed analysis on the extracted baseline is addressed

(5) A filter to separate baseline from traffic measurements on the fly

7.2.1. Data Arrangement. The scheme extracts the baseline of a traffic trace, arranged in a matrix, referred as Y. Two arrangements are possible: data traces of multiple links over the same period of time, or data on a single link broken into windows. When data traces of multiple links over an arbitrary period N is arranged into rows, the scheme returns a base-pattern valid for all the considered links over the period. Due to its validity over the space, it is referred as a "spatial" base-pattern. If M links are considered, then the data matrix $Y = \{Y_{mn}\}_{M\times N}$ with links $m = 0, \ldots, (M-1)$ and sample indices $n = 0, \ldots, (N-1)$. The goal behind analyzing time windows on a single link is to identify a base-pattern valid over time for the considered link - therefore is referred as the “temporal” base-pattern. Here, the time window N is chosen as described in the next section. The choice of the arrangement is application dependant. For example, anomaly detection may be more effective with temporal arrangements; whereas traffic characterization may prefer a spatial arrangement.

7.2.2. The Fundamental Period. To best capture temporal properties of the baseline, time-series has to be broken into cyclostationary periods [187] referred to here as the fundamental period. If the entire time-series is T samples long, and fundamental period is N, then time-series in broken into M windows, where $M = \lfloor T/N \rfloor$. Then the time series $y[t]$ is arranged row-wise on a matrix $Y = \{Y_{mn}\}_{M\times N}$; time window $m = 0, \ldots, (M-1)$ and sample index $n = 0, \ldots, (N-1)$ s.t. $t = mN + n$ and $Y_{mn} = y[t]$. A poorly selected period N will mis-align and truncate patterns, hampering recognition of the best base-pattern.

While Internet traffic in general exhibit trends that repeat week after week, such a period may not necessarily be obvious or clear in other networks. Corresponding time-series may not have well-defined frequency properties. Therefore, alternative methods have to be employed in identifying the fundamental period (N) of the trace. Autocorrelation Function (ACF) can be used to estimate the period by posing the candidate period as the lag of the function [188].

$$R_y[\tau] = \frac{\mathbb{E}\left[\langle y[k]-\mu, y[k+\tau]-\mu\rangle\right]}{\sigma^2} \tag{7.1}$$

where $y[k]$ is the time-series, μ is the mean of the series, σ^2 is the variance of the series, and τ is the lag. Then the optimal estimate for period N is given by:

$$N = \min \underset{\tau}{\arg\max}\; R_y[\tau] \tag{7.2}$$

If there happened to be multiple τ values that will maximize the ACF, then the least is taken. More efficient cycle detectors can be employed when the search space N is large.

7.2.3. The Common Component Across Time. The most common component in the dataset is identified using Robust-PCA [5]. Different from the classical PCA, RPCA breaks a given matrix Y into a low rank component L and a sparse component S as in (7.3).

$$Y = L + S \tag{7.3}$$

$$\text{s.t.} \quad \underset{L,S}{\arg\min} \quad \|L\|_* + \lambda \|S\|_1 \tag{7.4}$$

where $\|\cdot\|_*$ is the nuclear-norm (the sum of the singular values), $\|\cdot\|_1$ is the 1-norm, and typically, is chosen. This optimization problem is solved as an Augmented Lagrange Multiplier problem [189] with linear convergence.

The rank deficient component L carries elements common to all rows (i.e, periods or links). The rank deficiency often is interpreted as follows: the pattern in each row is a linear combinations of a few contributing sources. Since the few sources are common across the matrix, this low rank matrix represents the common component in the data. Traffic on a network on the other hand is the result of a large number of traffic sources. However, there are certain underlying repetitive phenomena, such as work hours and work patterns, when aggregated over a large number of users, result in equivalent logical effect on network traffic.

Our interest is on the low rank L matrix. Each row of L is the component in the corresponding row of Y common with rest of the rows in Y. Thus, L carries the common component across the dataset. The amount of details of traffic split between the sparse and the low rank components are balanced by λ. Here, λ can be treated as tunable parameter to make the low rank matrix (which is of our interest) detailed or less.

7.2.4. The Salient Component. Next, the salient patterns in the common component of the dataset are identified. They formulate the base-patterns of the dataset. Here we apply PCA [190] to the L matrix. In a Singular Value Decomposition (SVD) approach to solve PCA, L decomposes to:

$$L = U\Sigma V^T \tag{7.5}$$

and the principal components (PCs) of L projected on the basis of U^T are given by:

$$U^T L = \Sigma V^T = \{L_i\}\,,\ i = 0, \ldots, (M-1) \tag{7.6}$$

Each principal component (L_i) can be expressed as

$$L_i = \sigma_i V_i^T \tag{7.7}$$

where $\sigma_i = \Sigma_{ii}$.

Then the least number of PCs that will satisfactorily capture the information in the trace is selected. They are summed to form the salient component (p).

$$p = \{L_i\}\,,\ i \in I \tag{7.8}$$

where I is an index selected, s.t.,

$$\sum_{i \in I} \|L_i\|_2 \geq \alpha \sum_{\forall i} \|L_i\|_2 \tag{7.9}$$

where $\arg\min |I|$, and $\alpha < 1$, but close to 1. The selection criteria for the index set I is the least number of principal components to represent most of the variance of L. It is important to note that computing the PCs of the L matrix need not be done explicitly as a part of singular value shrinkage in RPCA the SVD of L is computed, and therefore PCs of L can be directly tapped from the algorithm. In selecting PCs to build the base-pattern the parameter α determines the amount of energy to be captured. With a higher α a base-pattern that resembles the low rank component much closely can be derived. However that will require selecting more PCs which also calls for maintaining more weights for each realization.

The resulting time-series $p[n]$ is present in all rows of Y and captures much of the behavior of the dataset. Thus, we call $p[n], n = 0, \ldots, (N-1)$ as the base-pattern of the dataset Y. The base pattern can be scaled back into the dataset using the coefficients in U, constructing a baseline for the dataset. For example the baseline for the m^{th} row of Y is:

$$p_m = \sum_{i \in I} U(m,i) L_i \tag{7.10}$$

This baseline is robust against contaminations such as anomalies in the dataset. Thus, is referred to as the Robust Base-Line (RBL). The RBL constructed with temporal base-patterns yields a robust baseline capturing the baseline behavior of a link; therefore is referred to as RBL-L (RBL for a link). Similarly RBL constructed with spatial base-pattern represent the baseline behavior over the network, and is referred to as RBL-N (RBL for the network).

7.2.5. Compact Representation. The base-pattern extracted above has a dimensionality of N. Also the analysis done so far is data independent. It is quite likely that further analysis is possible due to the nature of the Internet traffic data, which will allow a compressed description. Here we propose a compression strategy to reduce the dimensionality of the base-pattern. When long time windows are considered, e.g., weeks-long traces with minutes of sampling intervals, a certain degree of periodicity can be expected, which in turn will allow for a compressed representation of in the Fourier domain. Here, for demonstration purpose the compressibility in the Fourier basis is used. But we acknowledge the possibility of employing other basis appropriate for a given dataset. The dc component of Fourier domain corresponds to the mean of the time-series. It is automatically included in the base-pattern. Next, to select the other frequency components, consider the Discrete Fourier Transform (DFT) of the base-pattern.

$$P[k] = \sum_{n=0}^{N-1} p[n] e^{-\frac{2\pi\imath}{N}nk} \tag{7.11}$$

We seek a minimum cardinality subset of frequency indices (zeroth frequency is automatically included) that will capture the required energy (variance/information) of the system. Since network traffic traces are real valued data, we also need to maintain the symmetry in the frequency description. Therefore, we select a subset of frequencies K (referred as the dominant frequency set here forth); s.t. $\arg\min |K|, k \in K, (N-k) \in K$ and $k \neq 0$.

$$\sum_{k \in K} (P[k])^2 \geq \alpha\prime \sum_{k=1}^{N-1} (P[k])^2 \tag{7.12}$$

where $\alpha\prime < 1$, and preferable > 0.75. The resulting set K of frequency components inclusive of the zeroth frequency is a good smooth approximation to the base-pattern.

7.2.6. Extending over the Secondary Dimension. To make the compressed description obtained for a single link valid for the entire network or the compressed description obtained for the network over a single time window valid for multiple time windows, the union of the dominant frequency sets is taken. If the dominant frequency set of link-i (or window i) is K_i, and if the network has some q links (or q time windows), the dominant frequency set K for the entire network over a selected time history is,

$$K = K_0 \cup K_1 \cup \ldots \cup K_{q-1}. \tag{7.13}$$

The analysis above permits designing an FFT-filter for base-pattern separation in real-time. The goal of the FFT-filter is to break a given time series ($y[t]$) into its base-pattern component ($y_B[t]$) and the deviation from the base-pattern component ($y_D[t]$) on the fly. The most interesting feature of the filter is, it by-passes all the processing required to obtain the baseline and returns an approximation for the baseline (and the deviation) only with a frequency filtering operation.

$$y[t] = y_B[t] + y_D[t] \tag{7.14}$$

$$y_B[t] = \frac{1}{N} \sum_{k \in \{0 \cup K\}} e^{\frac{2\pi\imath}{N}kt} \sum_{j=0}^{N-1} y[j] e^{-\frac{2\pi\imath}{N}jk} \tag{7.15}$$

Similar to (15), $y_D[t]$ is found by replacing $\{0 \cup K\}$ with its complement.

7.3. Results

In this section we present results of applying the proposed baseline detection for Internet traffic. Results related to compact representation of base-patterns are also included.

7.3.1. Dataset. The nature of the dataset plays a key role over the extracted baseline. The baseline extracts the prominent and common structures across the dataset. Ideally, it should avoid sudden impulses which may arise due to unusual customer behaviors, attacks, network failures and other causes of traffic anomalies. If the basic structure of traffic - where the peaks and valleys lie, is common, irrespective of increasing, decreasing or even oscillating traffic trends, a common baseline is extracted. However, if either the different links or the different fundamental periods have disparate traffic trends and structure, there would be a few salient traces characterizing these different groups.

The publicly available dataset from Abilene network [152] is used in this work. It consists of 28 links spanning the United States. The dataset provides throughput on the links sampled at five minute intervals, starting from October 16^{th}, 2005. At sampling period of five minutes, each link produces 2016 samples a week. For the results presented below, 50 weeks of data on the 28 links are used.

The Abilene dataset however is fairly stable and has similar traffic structures across time. The traffic structure between links is also quite similar. Such a nature results in a single or perhaps a very few salient components. As discussed above this may not be the case with any general traffic dataset. If a dataset of heterogeneous or disparate traffic structures were used, we expect to see more salient components as explained in (7.8). These components will capture the principal structures of different groups. The weights $U(m, i)$ in (7.10) will express how each sample is biased to each salient component. Thus, even under heterogeneous traffic trends, the scheme will extract the salient trends as the baseline.

7.3.2. Estimating the Period.

The fundamental period N of data is found using the ACF as explained Section 7.2. Figure 7.1 shows the variation of ACF for all 28 links over lags up to 10,000. The ACF is maximized at lag 2016, which correspond to a period of a week. Therefore, a week (2016 samples) is selected as the fundamental period N of the dataset. The ACF for a period of a day (288 samples) is also indicated in the plot. In fact, choosing a week as the duration of the base-pattern can be supported visually as in Fig. 7.2. Figure 7.2 shows throughput over a period of four months (16 weeks). A pattern that repeats 16 times can be noted in the figure.

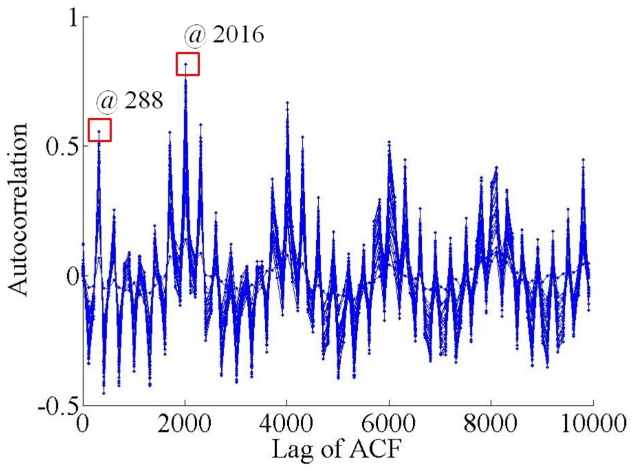


FIGURE 7.1. Autocorrelation function at different lags for all links.

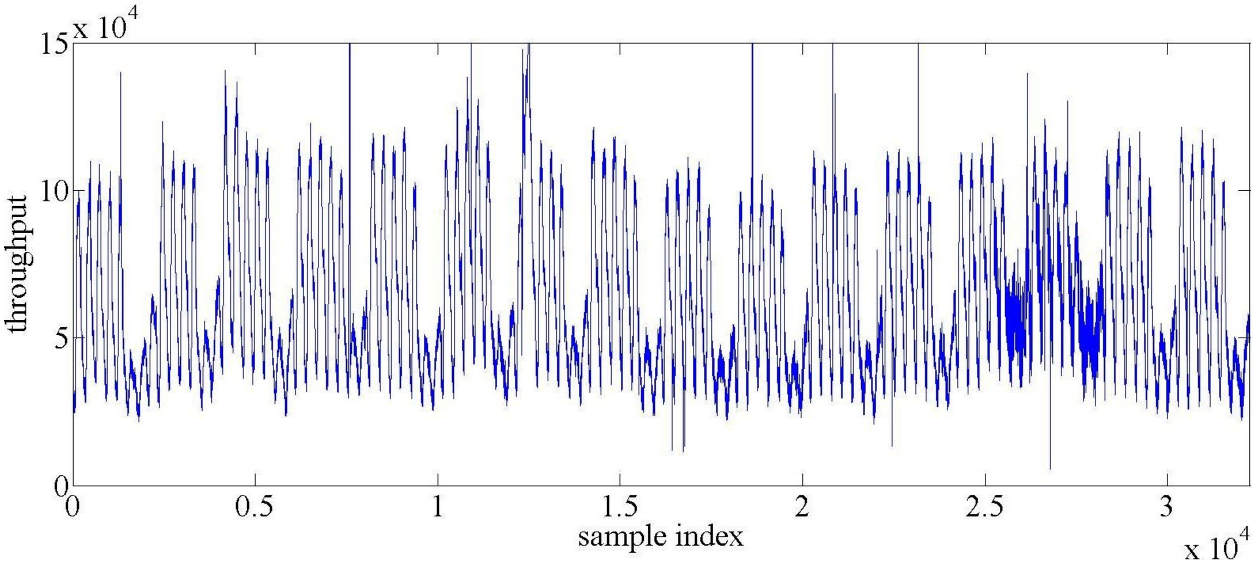


FIGURE 7.2. Traffic observed over a period of four months.

7.3.3. IDENTIFYING THE BASE-PATTERN. Data arranged as per Section 7.2 are decomposed using RPCA to a low rank L component and a sparse S component. Figure 7.3 shows an example of temporal base-pattern extracted for a 10 week long trace. The 10 week data block is broken into separate weeks and arranged as a 10×2016 matrix Y (five rows of which are shown in Fig. 7.3(a)). Then Y is decomposed to L and S (shown in Figs. 7.3(b) and (c) respectively) as per (7.3) and (7.4). As expected L turns out rather similar in all the rows, and differences are pushed to S. The rank of L is reduced to 5 from 10 in Y. Then PCA is applied to the low rank component L to identify the most salient time-series common to all time windows. A sufficient number of PCs has to be selected to capture much of the details. Figure 7.4 shows the decay of principal components of raw dataset and of RPCA low rank component, for both spatial and temporal arrangements. The keys in the legend are to be interpreted as, temporal: time-series on a single link, spatial: time-series of multiple links, PCA: PCs of Y, and base-pattern: PCs of L. The low rank component of the temporal arrangement has the fastest decaying PCs. In fact the first PC is over an order of magnitude larger than rest of the PCs. Therefore only the first PC is enough to obtain the temporal base-pattern for this dataset.

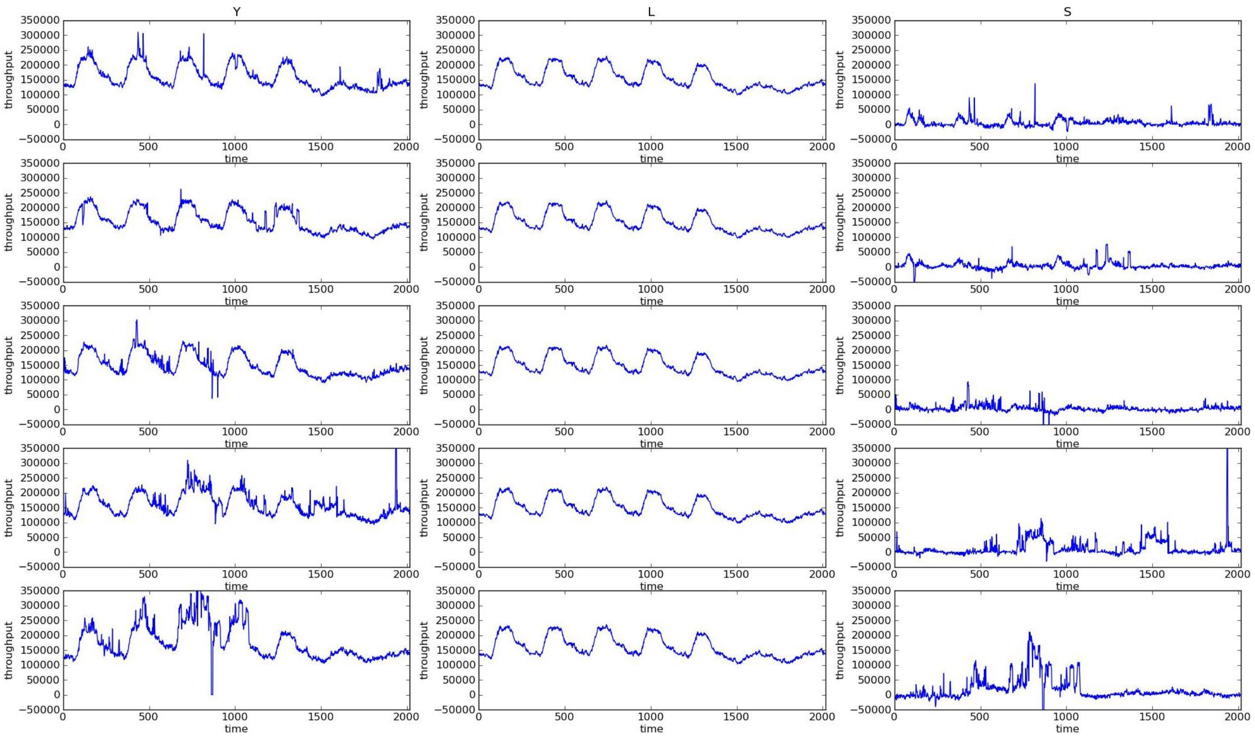


FIGURE 7.3. Sparse and low rank decomposition of a 10 week window: (a) data matrix Y, (b) low rank component L, and (c) sparse component S (decomposed throughput vs sample index).

7.3.4. FFT-FILTER. The base-pattern for the Abilene dataset shows certain periodicity suggesting the applicability of the proposed compressed analysis, and the FFT-filter. Fourier

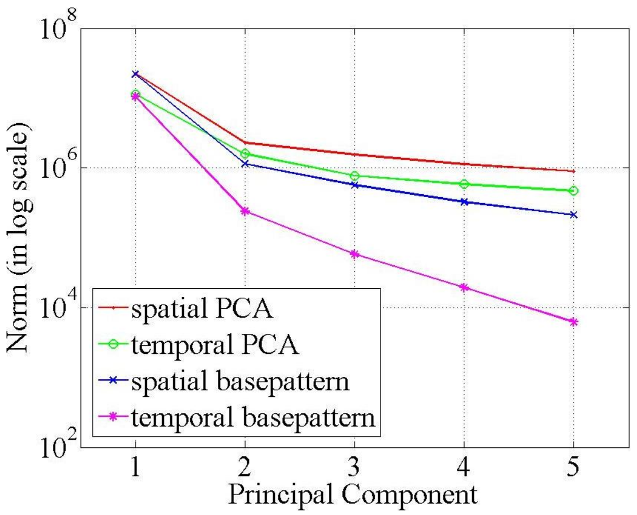


FIGURE 7.4. Decay of magnitude of principal components.

analysis revealed the magnitude of the zeroth frequency for any link is at least an order of magnitude greater than other frequency components. After the mean, about 85% of energy on average of all links is concentrated in a subset of 10 frequency components. Thus, we build an FFT-filter to select the ten frequency components with largest magnitudes. It was also found that, the superset of these frequencies, even across the entire network consisted of only 13 unique frequencies. This results in a compression from 2016 frequency component base-pattern characterization to a 13 frequency component characterization. Figure 7.5 shows the filtering performed using the 13 frequency component filter on five sample time series. The resulting filter separates baseline from traffic by-passing the tedious processing required in the scheme. Also, once the FFT-filter is designed, it may be valid over a long period of time due to the stability of the base-pattern as discussed later.

7.3.5. COMPARISON. To evaluate the effectiveness of the proposed scheme, we observe how closely the baselines capture raw traffic traces and avoid anomalies. Here we compare RBL-L and RBL-N with temporal and spatial PCA based baselinings [176], bl-7 baseline in BLGBA proposed in [185] and subsequently used in [186, 191], FFT-filter output baseline

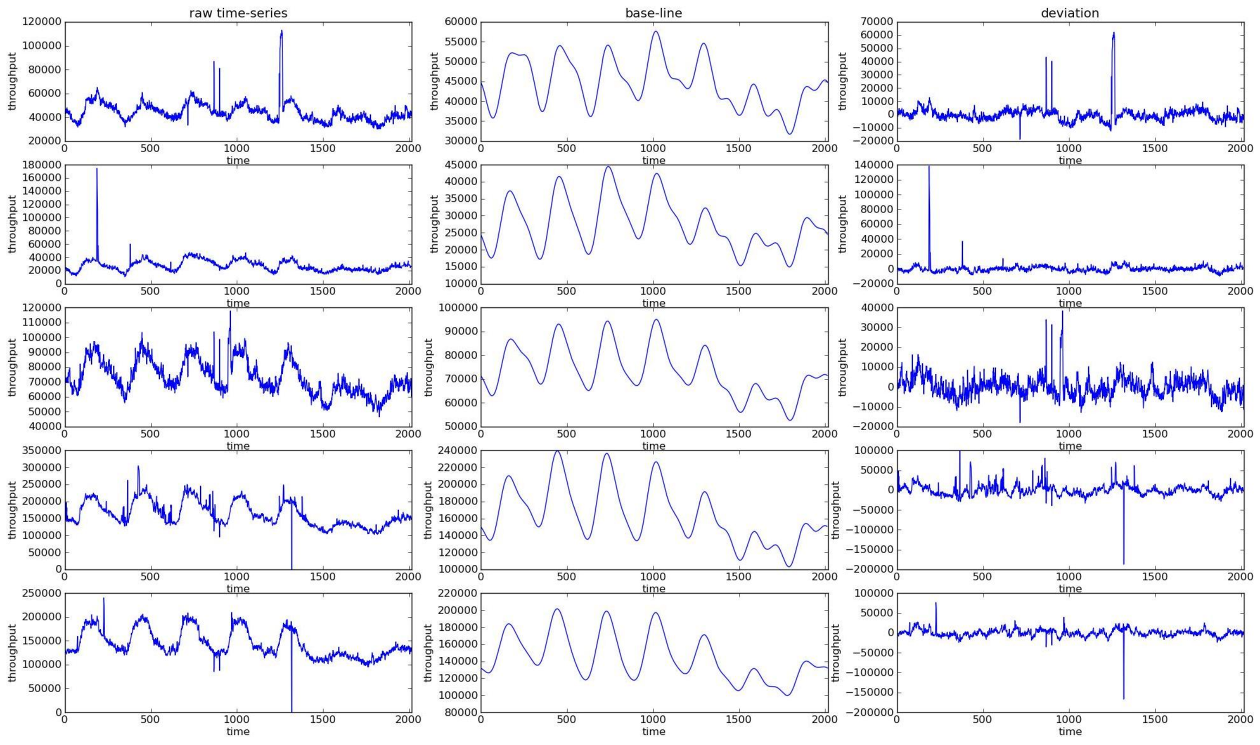


FIGURE 7.5. Sample operation of FFT-filter: (a) raw time-series, (b) filtered baseline, (c) deviation of from baseline (decomposed throughput vs sample index).

described in Section 7.2, and with low-pass filter baselining, where traffic is filtered thru a low-pass filter to extract the baseline. In BLGBA bl-7 each day of the week is modeled individually. Realizations for each sample is classified by BLGBA into five classes and the maximum value in the class greater than or equal to the 80% percentile is taken to be the baseline. Owing to the nature of the scheme, BLGBA baseline lies above the traffic trace most of the time, i.e., it over-estimates, whereas RBL follows the fundamental trace in the data. BLGBA independently looks at realizations of individual sample points and selects one; therefore it loses the relationship between adjacent samples. This leads to a time-wise uncorrelated noisy base-line. In RBL, relationships between samples are maintained.

Further, BLGBA returns one fixed pattern for the entire dataset. How that pattern should be scaled and shifted to fit to individual realizations, i.e., individual links or weeks, is not addressed. In RBL, as the final pattern is the output of PCA, the scaling coefficients $U(m, i)$ are automatically available. As can be seen in Fig 7.6, BLGBA is rather noisy defeating the very purposes of having a baseline. RBL baselines are consistent and also display less variance. We do acknowledge the simplicity of BLGBA and the ease of implementation. However, with the availability of hardware such as FPGAs and monitoring devices with significant computation and communication capability, the accuracy gains of RBL is not hard to realize.

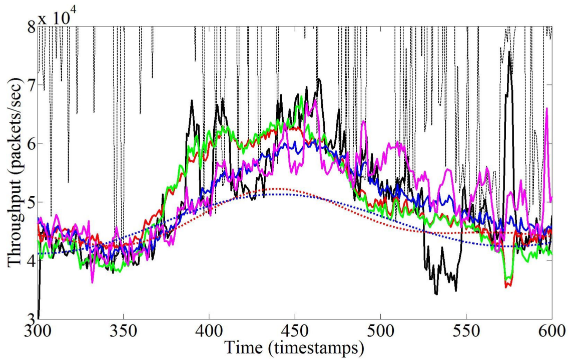
In Fig. 7.6 we demonstrate anomaly detection in three selected traces, using each of the baselines. The first (Fig. 7.6a and 7.6b) is an example of a typical trace. The second (Fig. 7.6c and 7.6d) has a large anomaly. The third trace (Fig. 7.6e and 7.6f) has a long-lasting anomaly. Figures 7.6a, 7.6c and 7.6e show the baselines generated by various methods super-imposed on traffic, in an attempt to see how closely each baseline follows traffic. Figures 7.6b, 7.6d and 7.6f show the anomalies detected using each method. Anomalies are detected by observing large deviations from baselines. Here the goal is recognize the effectiveness of each baselining technique for anomaly detection. Observing Fig. 7.6a BLGBA shows a noisy over-estimating baseline.

The frequency based methods provide a smoother baseline and PCA based methods closely follow the traffic. RBL shows the best fit to the traffic, while avoiding anomalies. As pointed out in [28], PCA based methods can sometimes be vulnerable to large anomalies and certain noise conditions. Figure 7.6d shows an example where PCA methods and RBL-N fail to detect a large anomaly. However, the RBL-L was capable of detecting this anomaly. Anomalies with long durations deform the frequency based baselines, as they form large low frequency descriptions. The long anomaly in Fig. 7.6f goes undetected by frequency based methods.

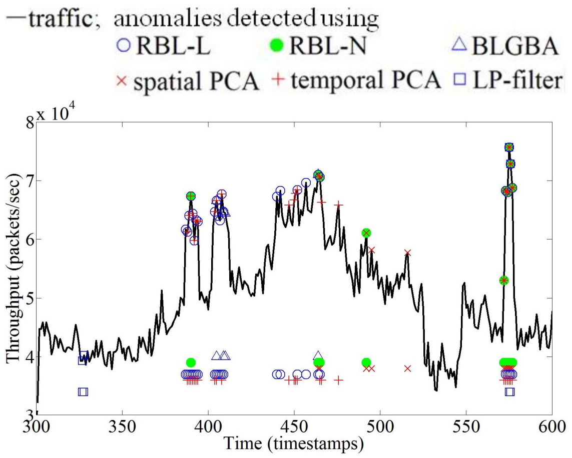
The strength in robust principal component based methods is that they are fairly immune to abrupt changes, such as unexpected customer behaviors. But methods based on classical PCA, and methods that consider every sample with equal importance, such as BLGBA, gain a bias from such. As the baseline is expected to capture the typical behavior, such biases are undesirable. Furthermore, when there are trends in traffic, RPCA identifies them quite effectively even if it is hidden. This is discussed shortly.

7.3.6. Stability. Here, we investigate the stability of the base-pattern. In Fig. 7.7 two extractions of base-patterns are considered: using 10 week blocks, and using 5 week blocks. Then the cross correlation between base-pattern of each of the blocks against the base-pattern of the initial block is observed to assess the stability. Furthermore, cross correlation of the data for the same time intervals is also presented, to assess degree of similarity. As can be noticed, with the exception of 5-week base-pattern for the weeks from 10 to 15, others show a high correlation to the initially derived base-pattern. This indicates that the base-pattern of traffic is fairly stable for this dataset, and it requires less realizations to calculate, as few as five weeks. The cross correlations for baselines always lie above cross correlations for raw data - a verification that the baselines capture the common components in data.

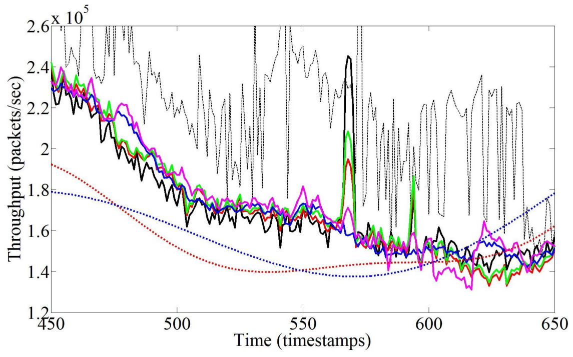
7.3.7. Patterns Buried in Noise. The advantage of baseline detection with a formal scheme is that it may reveal underlying patterns that are more subtle and buried in noise. An example of such is previewed in Fig. 7.8. Figure 7.8(a) shows weekend traffic on five different weeks. It is not much clear whether an underlying pattern exists between Saturday afternoon and Sunday morning (indicated by the box). However, when RBL is applied, it reveals a pattern in Figure 7.8(b) that can be verified by super-imposing back on the traffic trace as in Figure 7.8(c).



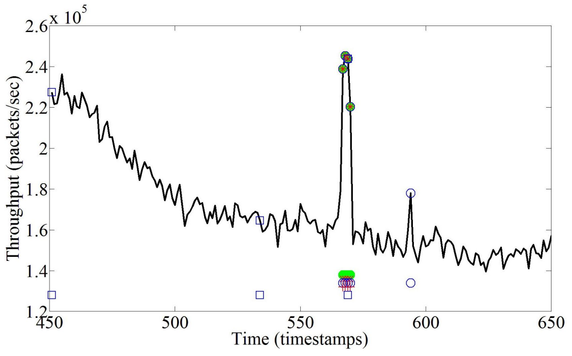
(A) a typical traffic trace



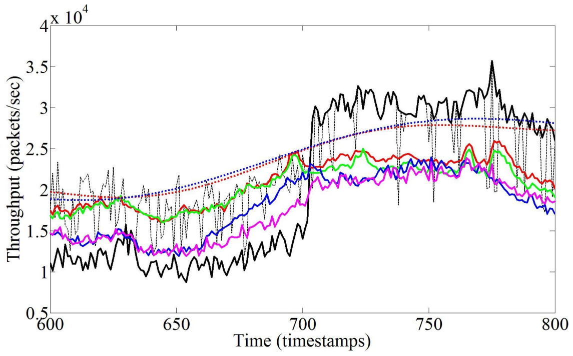
(B) anomalies detected on a typical traffic trace



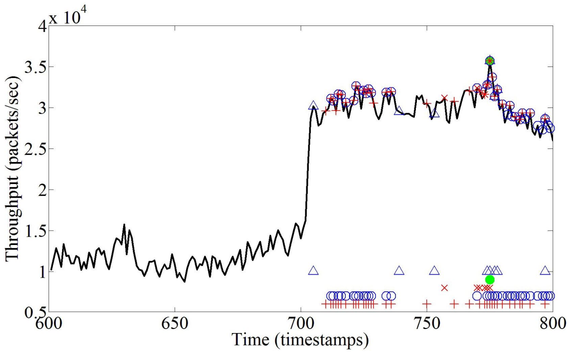
(C) a trace with a large anomaly



(D) detecting large anomalies



(E) a trace with a long anomaly



(F) detecting a long anomaly

Figure 7.6. Comparison of baselines and application on anomaly detection (throughput vs sample index). The instances at which anomalies have been detected by different strategies are also identified at the bottom of Figures to enhance clarity.

7.4. Applications

Utilities of the base-pattern $p[n]$ and its compact version $y_D[t]$ include a number of applications in traffic engineering. Range of applications includes traffic characterization, load

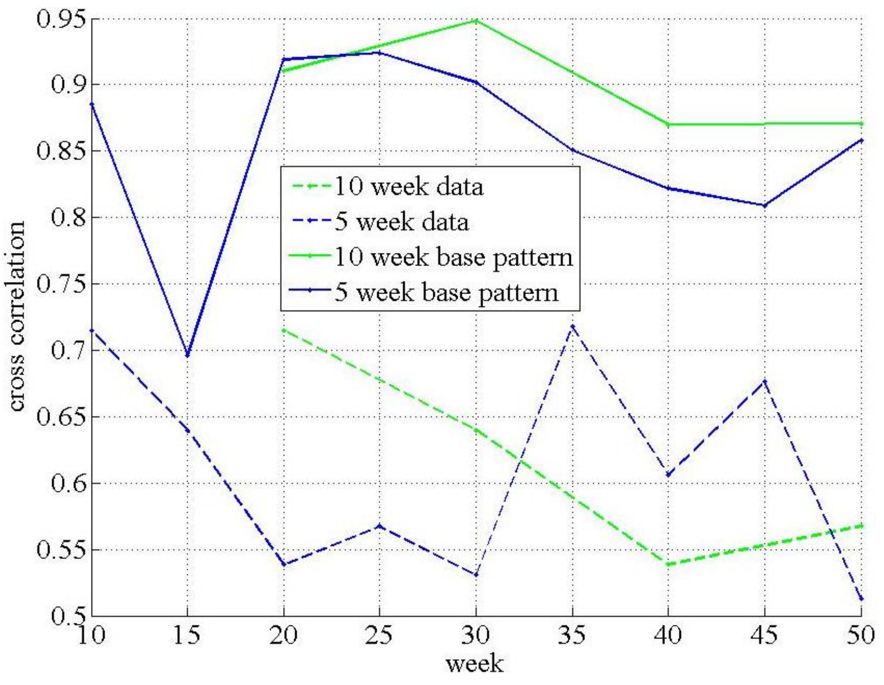


FIGURE 7.7. Variation of correlation of the base-pattern and traffic

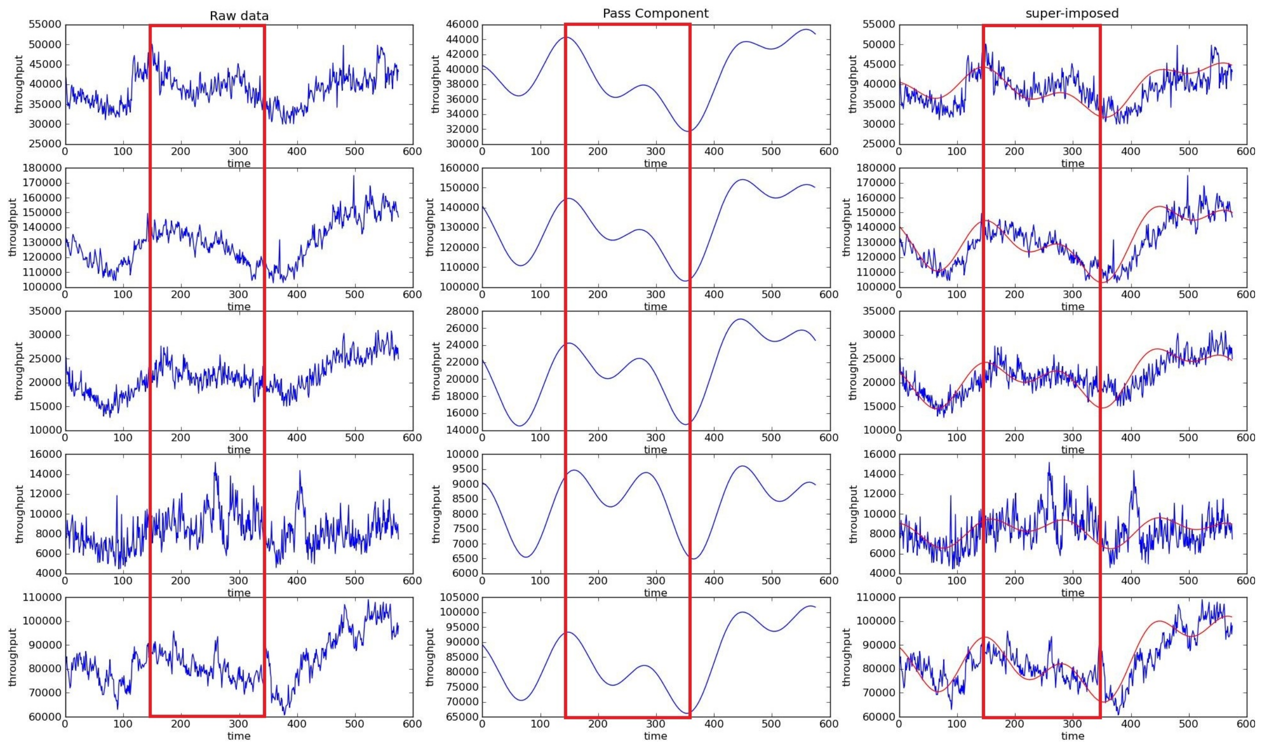


FIGURE 7.8. Example revealing subtle patterns; (a) raw time series (b) FFT-filtered smooth baseline; (c) baseline super-imposed on data (decomposed throughput vs sample index)

balancing, building robust networks, and pricing. The traffic baseline also may be related to the utilization and the availability of resources required by load balancing applications. In green computing applications, energy demanding resources can be scheduled to be online based on the demand/utility/base pattern.

Anomalies are large deviations of traffic from the expected. Baselining is an obvious choice for defining the expected traffic. When the difference between the observed traffic is beyond a certain threshold, say, a few times the standard deviation, then the observation is marked as anomalous. With an implementation such as the FFT-filter's deviation component as explained in Section 7.2, the deviation from the expected traffic can be readily extracted from the trace. The output here is de-trended for the baseline, and therefore can be thresholded with a fixed value to mark anomalies. Figure 7.6 also shows few examples demonstrating the effectiveness of using RBL for anomaly detection. Here, all the methods are tuned the same to declare measurements above two standard deviations as anomalous. BLGBA misses a few of the prominent anomalies. PCA based methods overlook large anomalies; while frequency based methods miss long duration anomalies. RBL effectively captures most of the major anomalies in the trace, which escape most other techniques.

Knowledge of baseline traffic behavior is useful in network planning and operations. Features such as resource demand dynamics, maximum and minimum loadings are well described by traffic baselines. Therefore baselines facilitate decisions regarding allocating and scheduling resources. Traffic Matrices (TMs) are a widely used in applications such as QoS routing and network capacity provisioning [192]. Methods such as Kalman filtering are often used to capture and predict the dynamics of TMs [193]. By providing the base pattern together with the appropriate scaling factor, TM specification can be simplified. Instead of the frequently updating the entire traffic matrix elements, the few scaling coefficients are needed to be updated only once a fundamental period.

7.5. Conclusions

A scheme for extracting baselines of network traffic traces was presented. The method was illustrated with the spatial and temporal arrangements of traffic to captures common trends in space and time respectively. It consists of two phases. The first uses RPCA to find the most common component of the dataset. Next, PCA was used to extracts the most salient trace in the common component. By identifying common frequency components of the baseline, we constructed an approximate smoother version of the baseline. An FFT filter is then designed to separate such a baseline from a traffic trace in real-time. RBL was found to be more effective in anomaly detection over number of other existing methods.

CHAPTER 8

Conclusions

This dissertation is centered around extracting, analyzing and modeling network data. We began with efficient network monitoring and fault localization. The proposed schemes monitor the network with a minimal load. Then upon detection of a fault, adaptive schemes based on compressive sensing rapidly localize the the faulty links. The schemes are tested on realistic topologies with realistic data models for accuracy, scalability and cost. Then we focused our attention to recovering data from a subset of samples. We generalized recovery bounds of compressive sensing to account for any sampling measure and applied CS for phenomena discovery in WSNs. Capabilities of wavelet transform is used for distributively compress data in a sensor network in order to minimize the communication cost. Matrix completion methods are used in conjunction with compressive sensing to reconstruct sensor field data from a small subset of samples. Thereafter, feature extraction from the data is taken into account. Recovery regions of RPCA are established empirically. Further, a cross validation principle is also developed to determine a successful recovery. A scheme to extract and derive network data features is developed and applied to detect a few network attack types. A method to detect subtle pattern is developed and applied for PCB testing. Then we proceed to network traffic behavior modeling. A component-wise approach is taken to model network traffic anomalies which do not adhere to common random processes. This method resulted in a concise description of traffic anomalies of a Internet2 network. Further, the model is used for efficient real-time communication of anomaly information. Finally, network traffic baselines are modelled. Techniques to extract baselines from highly contaminated data is devised. The tools developed as a result of these work are made publicly available.

Bibliography

[1] W. Xu, E. Mallada, and A. Tang, "Compressive sensing over graphs," in *Proceedings the 30th Annual IEEE International Conference on Computer Communications (IEEE INFOCOM 2011)*, pp. 2087–2095, April 2011.

[2] "The Internet2 network observatory data views." `http://www.internet2.edu/observatory/archive/data-views.html`, 2013.

[3] T. Karagiannis, M. Molle, and M. Faloutsos, "Long-range dependence ten years of internet traffic modeling," *IEEE Internet Computing*, vol. 8, pp. 57–64, September 2004.

[4] G. Terdik and T. Gyires, "Lèvy flights and fractal modeling of internet traffic," *IEEE/ACM Transactions on Networking*, vol. 17, pp. 120–129, February 2009.

[5] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?," *Journal of the ACM*, vol. 58, pp. 11:1–11:37, June 2011.

[6] V. Chandrasekaran, S. Sanghavi, P. Parrilo, and A. Willsky, "Rank-sparsity incoherence for matrix decomposition," *SIAM Journal on Optimization*, vol. 21, no. 2, pp. 572–596, 2011.

[7] V. Bandara, A. Pezeshki, and A. Jayasumana, "Modeling spatial and temporal behavior of internet traffic anomalies," in *Proceedings of the 35th Annual IEEE Conference on Local Computer Networks (IEEE LCN 2010)*, pp. 384–391, October 2010.

[8] N. Duffield, "Simple network performance tomography," in *Proceedings of the 3rd Annual ACM SIGCOMM Conference on Internet Measurement (IMC 2003)*, pp. 210–215, 2003.

[9] Y. Vardi, “Network tomography: Estimating source-destination traffic intensities from link data,” *Journal of the American Statistical Association*, vol. 91, no. 433, pp. 365–377, 1996.

[10] K. Naidu, D. Panigrahi, and R. Rastogi, “Detecting anomalies using end-to-end path measurements,” in *Proceedings of the 27th Annual IEEE Conference on Computer Communications (IEEE INFOCOM 2008)*, April 2008.

[11] E. Candès and T. Tao, “Decoding by linear programming,” *IEEE Transactions on Information Theory*, vol. 51, pp. 4203–4215, December 2005.

[12] J. Haupt, W. Bajwa, M. Rabbat, and R. Nowak, “Compressed sensing for networked data,” *IEEE Signal Processing Magazine*, vol. 25, pp. 92–101, March 2008.

[13] V. Bandara, A. Jayasumana, and R. Whitner, “An adaptive compressive sensing scheme for network tomography based fault localization,” in *Proceedings of the Annual IEEE International Conference of Communication (IEEE ICC 2014)*, June 2014.

[14] V. Bandara and A. Jayasumana, “Adaptive compressive sensing for network fault localization,” IN-PREPARATION.

[15] H. Li, “Research and implementation of an anomaly detection model based on clustering analysis,” in *Proceedings of the International Symposium on Intelligence Information Processing and Trusted Computing (IPTC)*, pp. 458–462, October 2010.

[16] I. Jolliffe, *Principal Component Analysis*. Springer Verlag, 1986.

[17] R. Baraniuk, “Compressive sensing [lecture notes],” *IEEE Signal Processing Magazine*, vol. 24, pp. 118–121, July 2007.

[18] R. Baraniuk, “Compressive sensing,” in *Proceedings of the 42nd Annual Conference on Information Sciences and Systems (CISS 2008)*, pp. iv–v, March 2008.

[19] E. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory*, vol. 52, pp. 489–509, February 2006.

[20] D. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, pp. 1289–1306, April 2006.

[21] H. Rauhut, "Compressive sensing and structured random matrices," in *Theoretical Foundations and Numerical Methods for Sparse Recovery* (M. Fornasier, ed.), vol. 9 of *Radon Series Computational Applied Mathematics*, pp. 1–92, deGruyter, 2010. Corrected Version (April 4, 2011): Noncommutative Khintchine inequality for Rademacher chaos (Theorem 6.22) corrected, Proof of Theorem 9.3 corrected. Proof of Lemma 8.2 corrected.

[22] "Joint photographic experts group." `http://www.jpeg.org/`, 2007.

[23] V. Bandara, A. Jayasumana, A. Pezeshki, T. Illangasekare, and K. Barnhart, "Subsurface plume tracking using sparse wireless sensor networks," *Electronic Journal of Structural Engineering, Special Issue: Sensor Network on Building Monitoring: from Theory to Real Application*, pp. 1–10, 2010.

[24] E. Candès and B. Recht, "Exact matrix completion via convex optimization," *Communications of the ACM*, vol. 55, pp. 111–119, June 2012.

[25] M. Boniface, B. Nasser, J. Papay, S. Phillips, A. Servin, X. Yang, Z. Zlatev, S. Gogouvitis, G. Katsaros, K. Konstanteli, G. Kousiouris, A. Menychtas, and D. Kyriazis, "Platform-as-a-service architecture for real-time quality of service management in clouds," in *Proceedings of the 5th International Conference on Internet and Web Applications and Services (ICIW)*, pp. 155–160, May 2010.

[26] I. Ari, B. Hong, E. Miller, S. Brandt, and D. Long, "Managing flash crowds on the internet," in *Proceedings of the 11th modeling, analysis & simulation of computer & telecommunication systems*, pp. 246–249, 2003.

[27] E. J. Candès, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on Pure and Applied Mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.

[28] H. Ringberg, A. Soule, J. Rexford, and C. Diot, "Sensitivity of pca for traffic anomaly detection," *SIGMETRICS Performance Evaluation Review*, vol. 35, pp. 109–120, June 2007.

[29] P. Barford, N. Duffield, A. Ron, and J. Sommers, "Network performance anomaly detection and localization," in *Proceedings of the 28th IEEE Conference on Computer Communications (INFOCOM)*, pp. 1377–1385, April 2009.

[30] H. X. Nguyen and P. Thiran, "Network loss inference with second order statistics of end-to-end flows," in *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement (IMC '07)*, pp. 227–240, 2007.

[31] D. Ghita, H. Nguyen, M. Kurant, K. Argyraki, and P. Thiran, "Netscope: Practical network loss tomography," in *Proceedings of the 29th IEEE Conference on Computer Communications (INFOCOM)*, pp. 1–9, March 2010.

[32] Y. Qiao, G. Wang, X.-s. Qiu, and R. Gu, "Network loss tomography using link independence," in *Proceedings of the IEEE Symposium on Computers and Communications (ISCC)*, pp. 569–574, July 2012.

[33] G. Fei and G. Hu, "Accurate and effective inference of network link loss from unicast end-to-end measurements," *IET Communications*, vol. 6, pp. 2989–2997, November 2012.

[34] W. Zhu, "An efficient loss rate estimator in multicast tomography and its validity," in *Proceedings of the 3rd IEEE International Conference on Communication Software and Networks (ICCSN)*, pp. 90–94, May 2011.

[35] A. Krishnamurthy and A. Singh, "Robust multi-source network tomography using selective probes," in *Proceedings of the 31st Annual IEEE International Conference on Computer Communications (IEEE INFOCOM 2012)*, pp. 1629–1637, March 2012.

[36] Y. Tsang, M. Coates, and R. Nowak, "Network delay tomography," *IEEE Transactions on Signal Processing*, vol. 51, pp. 2125–2136, August 2003.

[37] A. Miyamoto, K. Watanabe, and K. Ikeda, "Packet loss rate estimation with active and passive measurements," in *Proceedings of the Signal Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 1–4, December 2012.

[38] S. Agrawal, K. V. M. Naidu, and R. Rastogi, "Diagnosing link-level anomalies using passive probes," in *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM)*, pp. 1757–1765, May 2007.

[39] M. H. Raza, B. Robertson, W. J. Phillips, and J. Ilow, "Network tomography by non negative matrix factorization (NNMF)," in *Proceedings of the International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*, pp. 58–64, July 2010.

[40] M. H. Raza, B. Robertson, W. J. Phillips, and J. Ilow, "Multimetric network tomography," in *Proceedings of the International Conference on Data Communication Networking (DCNET)*, pp. 1–5, July 2010.

[41] W. Xu, M. Wang, E. Mallada, and A. Tang, "Recent results on sparse recovery over graphs," in *Conference Record of the 45th Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, pp. 413–417, November 2011.

[42] M. Firooz and S. Roy, "Network tomography via compressed sensing," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM 2010)*, pp. 1–5, December 2010.

[43] J. Haupt, R. Nowak, and R. Castro, "Adaptive sensing for sparse signal recovery," in *Proceedings of the 13th IEEE Digital Signal Processing Workshop and 5th IEEE Signal Processing Education Workshop*, pp. 702–707, January 2009.

[44] A. Gilbert and P. Indyk, "Sparse recovery using sparse matrices," *Proceedings of the IEEE*, vol. 98, pp. 937–947, June 2010.

[45] E. Lawrence, G. Michailidis, and V. N. Nair, "Network delay tomography using flexicast experiments," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 68, no. 5, pp. 785–813, 2006.

[46] E. Candès and J. Romberg, "ℓ1-magic." `http://users.ece.gatech.edu/~justin/l1magic/`, 2005.

[47] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-Point Algorithms for Inverse Problems in Science and Engineering* (H. H. Bauschke, R. S. Burachik, P. L. Combettes, V. Elser, D. R. Luke, and H. Wolkowicz, eds.), Springer Optimization and Its Applications, pp. 185–212, Springer New York, 2011.

[48] D. Needell and J. Tropp, "Cosamp: Iterative signal recovery from incomplete and inaccurate samples," *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 301–321, 2009.

[49] B. Quoitin, V. Van den Schrieck, P. Franois, and O. Bonaventure, "IGen: Generation of router-level internet topologies through network design heuristics," in *Proceedings of the 21st International Teletraffic Congress (ITC)*, pp. 1–8, September 2009.

[50] G. Haßlinger and O. Hohlfeld, "The gilbert-elliott model for packet loss in real time services on the internet," in *Proceedings of the 14th GI/ITG Conference - Measuring, Modelling and Evaluation of Computer and Communication Systems (MMB)*, pp. 1–15, March 2008.

[51] "Planetlab." `http://www.planet-lab.org/`.

[52] "Packet portal." `http://www.jdsu.com/en-us/Test-and-Measurement/campaigns/Pages/packetportal.aspx`.

[53] "SF probe." `http://www.jdsu.com/en-us/Optical-Communications/Products/optical-transceivers/packetportal-sfprobes/Pages/default.aspx`.

[54] P. Sattari, A. Markopoulou, C. Fragouli, and M. Gjoka, "A network coding approach to loss tomography," *IEEE Transactions on Information Theory*, vol. 59, pp. 1532–1562, March 2013.

[55] E. Candès and T. Tao, "The power of convex relaxation: Near-optimal matrix completion," *IEEE Transactions on Information Theory*, vol. 56, no. 5, pp. 2053–2080, 2010.

[56] A. Soni and J. Haupt, "Learning sparse representations for adaptive compressive sensing," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2097–2100, March 2012.

[57] R. Calderbank, S. Howard, and S. Jafarpour, "Construction of a large class of deterministic sensing matrices that satisfy a statistical isometry property," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, pp. 358–374, April 2010.

[58] J.-J. Fuchs, "On sparse representations in arbitrary redundant bases," *IEEE Transactions on Information Theory*, vol. 50, no. 6, pp. 1341–1344, 2004.

[59] J. Tropp, "Recovery of short, complex linear combinations via l1ℓ_1 minimization," *IEEE Transactions on Information Theory*, vol. 51, no. 4, pp. 1568–1570, 2005.

[60] B. Edwards, S. A. Hofmeyr, G. Stelle, and S. Forrest, "Internet topology over time," *CoRR*, vol. abs/1202.3993, 2012.

[61] F. Gont, R. Atkinson, and C. Pignataro, "Recommendations on filtering of ipv4 packets containing ipv4 options." RFC7126, February 2014.

[62] N. Duffield, "Network tomography of binary network performance characteristics," *IEEE Transactions on Information Theory*, vol. 52, pp. 5373–5388, December 2006.

[63] G. Samorodnitsky and M. S. Taqqu, *Stable non-Gaussian random processes : stochastic models with infinite variance.* Stochastic modeling, New York: Chapman & Hall.

[64] J. Andersson and J.-O. Strömberg, "On the theorem of uniform recovery of random sampling matrices," *ArXiv e-prints*, June 2012.

[65] D. C. Dhanapala, *Anchor Centric Virtual Coordinate Systems in Wireless Sensor Networks: From Self-Organization to Network Awareness.* dissertation, Colorado State University, 2012.

[66] M. Zhang, M. C. Chan, and A. Ananda, "Connectivity monitoring in wireless sensor networks," in *Proceedings of the 5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS 2008)*, pp. 69–79, September 2008.

[67] Y. Wang, H. Wu, and N.-F. Tzeng, "Cross-layer protocol design and optimization for delay/fault-tolerant mobile sensor networks (DFT-MSN's)," *IEEE Journal on Selected Areas in Communications*, vol. 26, pp. 809–819, June 2008.

[68] J. Haupt, W. Bajwa, M. Rabbat, and R. Nowak, "Compressed sensing and network monitoring," *The Next Wave*, vol. 18, no. 3, pp. 16–25, 2010.

[69] M. McCutcheon, H. Farahani, J. Stednick, G. Buchleiter, and T. Green, "Effect of soil water on apparent soil electrical conductivity and texture relationships in a dryland field," *Biosystems Engineering*, vol. 94, pp. 19–32, 2006.

[70] J. Burrell, T. Brooke, and R. Beckwith, "Vineyard computing: sensor networks in agricultural production," *IEEE Pervasive Computing*, vol. 3, pp. 38–45, January 2004.

[71] W. Zhao, M. Ammar, and E. Zegura, "A message ferrying approach for data delivery in sparse mobile ad hoc networks," in *Proceedings of the 5th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '04)*, pp. 187–198, 2004.

[72] R. Shah, S. Roy, S. Jain, and W. Brunette, "Data mules: modeling a three-tier architecture for sparse sensor networks," in *Proceedings of the 1st IEEE International Workshop on Sensor Network Protocols and Applications*, pp. 30–41, May 2003.

[73] "When cars can talk." `http://www.scientificcomputing.com/news-DS-When-Cars-can-Talk-032211.aspx`.

[74] F. Fazel, M. Fazel, and M. Stojanovic, "Random access compressed sensing in underwater sensor networks," in *Proceedings of the 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 768–774, September 2010.

[75] J. Meng, H. Li, and Z. Han, "Sparse event detection in wireless sensor networks using compressive sensing," in *Proceedings of the 43rd Annual Conference on Information Sciences and Systems (CISS 2009)*, pp. 181–185, March 2009.

[76] G. Quer, R. Masiero, D. Munaretto, M. Rossi, J. Widmer, and M. Zorzi, "On the interplay between routing and signal representation for compressive sensing in wireless sensor networks," in *Information Theory and Applications Workshop*, pp. 206–215, February 2009.

[77] U. Lee, B. Zhou, M. Gerla, E. Magistretti, P. Bellavista, and A. Corradi, "Mobeyes: smart mobs for urban monitoring with a vehicular sensor network," *IEEE Wireless Communications*, vol. 13, pp. 52–57, October 2006.

[78] S. Lee, S. Pattem, M. Sathiamoorthy, B. Krishnamachari, and A. Ortega, "Spatially-localized compressed sensing and routing in multi-hop sensor networks.," in *GSN* (N. Trigoni, A. Markham, and S. Nawaz, eds.), vol. 5659 of *Lecture Notes in Computer Science*, pp. 11–20, Springer, 2009.

[79] M. Sartipi and R. Fletcher, "Energy-efficient data acquisition in wireless sensor networks using compressed sensing," in *Proceedings of the Data Compression Conference (DCC)*, pp. 223–232, March 2011.

[80] C. Feng, S. Valaee, and Z. Tan, "Localization of wireless sensors using compressive sensing for manifold learning," in *Proceedings of the 20th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, pp. 2715–2719, September 2009.

[81] B. Zhang, X. Cheng, N. Zhang, Y. Cui, Y. Li, and Q. Liang, "Sparse target counting and localization in sensor networks based on compressive sensing," in *Proceedings of the 30th IEEE International Conference on Computer Communications (IEEE INFOCOM 2011)*, pp. 2255–2263, April 2011.

[82] N. Ahmed, T. Natarajan, and K. Rao, "Discrete cosine transform," *IEEE Transactions on Computers*, vol. C-23, pp. 90–93, January 1974.

[83] D. Braginsky and D. Estrin, "Rumor routing algorthim for sensor networks," in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA '02)*, pp. 22–31, 2002.

[84] PRISM Group, "Climatewizard." `http://www.climatewizard.org/index.html`, February 2007.

[85] K. Romer and F. Mattern, "The design space of wireless sensor networks," *IEEE Wireless Communications*, vol. 11, pp. 54–61, December 2004.

[86] F. Shu, M. Halgamuge, and W. Chen, "Building automation system using wireless sensor networks: Radio characteristics and energy efficient communication protocols," *Electronic Journal of Structural Engineering (EJSE) - Special Issue: Sensor Network for Building Monitoring: From Theory to Real Application*, pp. 66–73, 2009.

[87] M. Halgamuge, T. Chan, and P. Mendis, "Experimental study of link quality distribution in sensor network deployment for building environment," *Electronic Journal of Structural Engineering (EJSE) - Special Issue: Sensor Network for Building Monitoring: From Theory to Real Application*, pp. 28–34, 2009.

[88] L. Peralta, L. Brito, and B. Gouveia, "The WISE-MUSE project: Environmental monitoring and controlling of museums based on wireless sensor networks," *Electronic Journal of Structural Engineering (EJSE) - Special Issue: Sensor Network for Building Monitoring: From Theory to Real Application*, pp. 46–57, 2009.

[89] S. Alahakoon, D. Preethichandra, and E. Ekanayake, "Sensor network applications in structures - a survey," *Electronic Journal of Structural Engineering (EJSE) - Special Issue: Sensor Network for Building Monitoring: From Theory to Real Application*, pp. 1–10, 2009.

[90] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, pp. 102–114, August 2002.

[91] L. Porta, T. Illangasekare, P. Loden, Q. Han, and A. Jayasumana, "Continuous plume monitoring using wireless sensors: Proof of concept in intermediate scale tank," *Journal of Environmental Engineering*, vol. 135, no. 9, pp. 831–838, 2009.

[92] P. Loden, Q. Han, L. Porta, T. Illangasekare, and A. P. Jayasumana, "A wireless sensor system for validation of real-time automatic calibration of groundwater transport models," *Journal of System Software*, vol. 82, pp. 1859–1868, November 2009.

[93] R. W. Puls and C. J. Paul, "Multi-layer sampling in conventional monitoring wells for improved estimation of vertical contaminant distributions and mass," *Journal of Contaminant Hydrology*, vol. 25, no. 12, pp. 85–111, 1997.

[94] A. Kaya and H. Fang, "Identification of contaminated soils by dielectric constant and electrical conductivity," *Journal of Environmental Engineering*, vol. 123, no. 2, pp. 169–177, 1997.

[95] A. Jayasumana, Q. Han, and T. Illangasekare, "Virtual sensor networks - a resource efficient approach for concurrent applications," in *Proceedings of the 4th International Conference on Information Technology (ITNG '07)*, pp. 111–115, April 2007.

[96] Q. Han, A. Jayasumana, T. Illangaskare, and T. Sakaki, "A wireless sensor network based closed-loop system for subsurface contaminant plume monitoring," in *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing (IPDPS 2008)*, pp. 1–5, April 2008.

[97] T. ElBatt, "On the trade-offs of cooperative data compression in wireless sensor networks with spatial correlations," *IEEE Transactions on Wireless Communications*, vol. 8, pp. 2546–2557, May 2009.

[98] J. Zhuo, C. Meng, and M. Zou, "A task scheduling algorithm of single processor parallel test system," in *Proceedings of the 8th ACIS International Conference on Software*

Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD 2007), vol. 1, pp. 627–632, July 2007.

[99] H. Jin, Q. Ren, J. Li, and Y. Shi, "An approximate query processing method based on data correlation in sensor networks," in *Embedded Software and Systems, 2008. ICESS '08. International Conference on*, pp. 13–18, July 2008.

[100] A. Deligiannakis, Y. Kotidis, and N. Roussopoulos, "Dissemination of compressed historical information in sensor networks," *The VLDB Journal*, vol. 16, no. 4, pp. 439–461, 2007.

[101] A. Guitton, A. Skordylis, and N. Trigoni, "Utilizing correlations to compress time-series in traffic monitoring sensor networks," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC 2007)*, pp. 2479–2483, March 2007.

[102] M. Vetterli and J. Kovačevic, *Wavelets and Subband Coding.* Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1995.

[103] K. Barnhart, *Re-conceptualizing groundwater transport models 2 in the wireless sensor network data context.* dissertation, Colorado School of Mines, 2010.

[104] K. Barnhart, I. Urteaga, Q. Han, A. Jayasumana, and T. Illangasekare, "On integrating groundwater transport models with wireless sensor networks," *Ground Water*, vol. 48, no. 5, pp. 771–780, 2010.

[105] C. Zheng and P. P. Wang, "MT3DMS: A modular three-dimensional multi-species transport model for simulation of advection, dispersion, and chemical reactions of contaminants in ground-water systems. documentation and user's guide," in *Contract Report SERDP-99-1, U.S. Army Engineer Research and Development*, 1999.

[106] A. Harbaugh, E. Banta, M. Hill, and M. McDonald, "The USGS modular ground-water model user guide to modular-ization concepts and the groundwater flow process," *USGS Open File Report 00-92*, 2000.

[107] R. Paffenroth, P. Du Toit, L. Scharf, A. Jayasumana, V. Bandara, and R. Nong, "Space-time signal processing for distributed pattern detection in sensor networks," *Selected Topics in Signal Processing*, vol. 6, pp. 839309–839309–15, January 2013.

[108] R. Paffenroth, P. Du Toit, L. Scharf, A. Jayasumana, V. Bandara, and R. Nong, "Distributed pattern detection in cyber networks," pp. 84080J–84080J–13, 2012.

[109] P. Du Toit, R. Paffenroth, and R. Nong, "Semi-definite program for solving augmented lagrange multipliers for rpca," IN-PREPARATION.

[110] Z. Zhang, X. Liang, A. Ganesh, and Y. Ma, "Tilt: Transform invariant low-rank textures," in *ACCV (3)'10*, pp. 314–328, 2010.

[111] P. Yigang, A. Ganesh, J. Wright, X. Wenli, and Y. Ma, "Rasl: Robust alignment by sparse and low-rank decomposition for linearly correlated images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 763–770, 2010.

[112] G. Zhu, S. Yan, and Y. Ma, "Image tag refinement towards low-rank, content-tag prior and error sparsity," in *Proceedings of the international conference on Multimedia (MM '10)*, pp. 461–470, 2010.

[113] X. Hou, J. Harel, and C. Koch, "Image signature: Highlighting sparse salient regions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 1, pp. 194–201, 2012.

[114] H. Xu, C. Caramanis, and S. Sanghavi, "Robust pca via outlier pursuit," *IEEE Transactions on Information Theory*, vol. 58, no. 5, pp. 3047–3064, 2012.

[115] Z. Zhou, X. Li, J. Wright, E. Candes, and Y. Ma, "Stable principal component pursuit," in *Proceedings of the IEEE International Symposium on Information Theory Proceedings (ISIT)*, pp. 1518–1522, 2010.

[116] M. Tao and X. Yuan, "Recovering low-rank and sparse components of matrices from incomplete and noisy observations," *SIAM Journal on Optimization*, vol. 21, no. 1, pp. 57–81, 2011.

[117] Å. Björck and G. Golub, "Numerical Methods for Computing Angles Between Linear Subspaces," *Mathematics of Computation*, vol. 27, no. 123, pp. 579–594, 1973.

[118] G. Zuccon, L. Azzopardi, and C. Rijsbergen, "Semantic spaces: Measuring the distance between different subspaces," in *Proceedings of the 3rd International Symposium on Quantum Interaction (QI '09)*, pp. 225–236, 2009.

[119] Z. Lin, M. Chen, and Y. Ma, "The Augmented Lagrange Multiplier Method for Exact Recovery of Corrupted Low-Rank Matrices," *ArXiv e-prints*, September 2010.

[120] L. Foster, "San jose state university singular matrix database." `http://www.math.sjsu.edu/singular/matrices/`.

[121] P. Hansen, "Regularization tools." `http://www.imm.dtu.dk/~pcha/Regutools/`.

[122] M. L. Baart, "The use of auto-correlation for pseudo-rank determination in noisy ill-conditioned linear least-squares problems," *IMA Journal of Numerical Analysis*, no. 2, pp. 241–247, 1982.

[123] R. A. Jr., W. Boland, V. Faber, and G. Wing, "Singular values and condition numbers of Galerkin matrices arising from linear integral equations of the first kind," *Journal of Mathematical Analysis and Applications*, vol. 109, no. 2, pp. 564–590, 1985.

[124] G. M. Wing and J. D. Zahrt, "A primer on integral equations of the first kind," *SIAM, Philadelphia*, p. 109, 1991.

[125] M. Tehranipoor and F. Koushanfar, "A survey of hardware trojan taxonomy and detection," *IEEE Design Test of Computers*, vol. 27, pp. 10–25, January 2010.

[126] "Finding counterfeits." `http://www.eetimes.com/discussion/break-points/4216656/Finding-counterfeits`.

[127] M. Tehranipoor and B. Sunar, "Hardware trojan horses," in *Towards Hardware-Intrinsic Security*, pp. 167–187, 2010.

[128] H. Salmani, M. Tehranipoor, and J. Plusquellic, "A novel technique for improving hardware trojan detection and reducing trojan activation time," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, pp. 112–125, January 2012.

[129] P. C. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology (CRYPTO '99)*, pp. 388–397, 1999.

[130] A. Moradi, M. Kasper, and C. Paar, "Black-box side-channel attacks highlight the importance of countermeasures," in *Topics in Cryptology CT-RSA 2012* (O. Dunkelman, ed.), vol. 7178 of *Lecture Notes in Computer Science*, pp. 1–18, Springer Berlin Heidelberg, 2012.

[131] R. Chakraborty, S. Narasimhan, and S. Bhunia, "Hardware trojan: Threats and emerging solutions," in *Proceedings of the IEEE International High Level Design Validation and Test Workshop (HLDVT 2009)*, pp. 166–171, November 2009.

[132] X. He, Y. Malaiya, A. Jayasumana, K. Parker, and S. Hird, "Principal component analysis-based compensation for measurement errors due to mechanical misalignments in pcb testing," in *Proceedings of the International Test Conference (ITC 2010)*, pp. 1–10, November 2010.

[133] D. Donoho, "For most large underdetermined systems of equations, the minimal l1ℓ_1-norm near-solution approximates the sparsest near-solution," tech. rep., Communications on Pure and Applied Mathematics, 2004.

[134] G. W. Stewart, "On the early history of the singular value decomposition," *SIAM Review*, vol. 35, pp. 551–566, December 1993.

[135] X. He, Y. Malaiya, A. Jayasumana, K. Parker, and S. Hird, "An outlier detection based approach for pcb testing," in *Proceedings of the International Test Conference (ITC 2009)*, pp. 1–10, November 2009.

[136] P. Barford, J. Kline, D. Plonka, and A. Ron, "A signal analysis of network traffic anomalies," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment (IMW '02)*, pp. 71–82, 2002.

[137] P. Barford and D. Plonka, "Characteristics of network traffic flow anomalies," in *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement (IMW '01)*, pp. 69–73, 2001.

[138] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," in *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '04)*, pp. 219–230, 2004.

[139] R. Kompella, J. Yates, A. Greenberg, and A. Snoeren, "Detection and localization of network black holes," in *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM 2007)*, pp. 2180–2188, May 2007.

[140] L. Huang, M. Jordan, A. Joseph, M. Garofalakis, and N. Taft, "In-network PCA and anomaly detection," in *Advances in Neural Information Processing Systems*, pp. 617–624, MIT Press, 2006.

[141] M. Thottan and C. Ji, "Anomaly detection in ip networks," *IEEE Transactions on Signal Processing*, vol. 51, pp. 2191–2204, August 2003.

[142] B. Raghunath and S. Mahadeo, "Network intrusion detection system (NIDS)," in *Proceedings of the 1st International Conference on Emerging Trends in Engineering and Technology (ICETET '08)*, pp. 1272–1277, July 2008.

[143] R. Cullingford, "Correlation and collaboration in anomaly detection," in *Proceedings of the Conference For Homeland Security - Cybersecurity Applications Technology (CATCH '09)*, pp. 251–254, March 2009.

[144] S. Kandula, D. Katabi, and J.-P. Vasseur, "Shrink: A tool for failure diagnosis in ip networks," in *Proceedings of the ACM SIGCOMM Workshop on Mining Network Data (MineNet '05)*, pp. 173–178, 2005.

[145] R. Kompella, J. Yates, A. Greenberg, and A. Snoeren, "Ip fault localization via risk modeling," in *Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation - Volume 2 (NSDI '05)*, pp. 57–70, 2005.

[146] M. Steinder and A. Sethi, "A survey of fault localization techniques in computer networks," *Science of Computer Programming - Topics in System Administration*, vol. 53, no. 2, pp. 165–194, 2004.

[147] M. Natu and A. Sethi, "Probabilistic fault diagnosis using adaptive probing," in *Proceedings of the Distributed Systems: Operations and Management 18th IFIP/IEEE International Conference on Managing Virtualization of Networks and Services (DSOM '07)*, pp. 38–49, 2007.

[148] P. Chhabra, C. Scott, E. Kolaczyk, and M. Crovella, "Distributed spatial anomaly detection," in *Proceedings of the 27th Conference on Computer Communications (IEEE INFOCOM 2008)*, April 2008.

[149] M. Crovella and E. Kolaczyk, "Graph wavelets for spatial traffic analysis," in *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications (IEEE INFOCOM 2003)*, vol. 3, pp. 1848–1857, March 2003.

[150] R. Coifman and M. Maggioni, "Diffusion wavelets," *Applied and Computational Harmonic Analysis - Special Issue: Diffusion Maps and Wavelets*, vol. 21, no. 1, pp. 53–94, 2006.

[151] M. Coates, Y. Pointurier, and M. Rabbat, "Compressed network monitoring for ip and all-optical networks," in *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement (IMC '07)*, pp. 241–252, 2007.

[152] T. Oetiker and D. Rand, "Internet2 network noc - historical abilene data." `http://noc.net.internet2.edu/i2network/live-network-status/historical-abilene-data.html`, 2011.

[153] I. Daubechies, *Ten Lectures on Wavelets.* Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1992.

[154] M. DeGroot, *Probability and Statistics.* Addison-Wesley, 1991.

[155] Z. A. Lomnicki, "On the distribution of products of random variables," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 29, no. 3, pp. 513–524, 1967.

[156] I. Paschalidis and G. Smaragdakis, "Spatio-temporal network anomaly detection by assessing deviations of empirical measures," *IEEE/ACM Transactions on Networking*, vol. 17, pp. 685–697, June 2009.

[157] P. Huang, A. Feldmann, and W. Willinger, "A non-instrusive, wavelet-based approach to detecting network performance problems," in *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement (IMW '01)*, pp. 213–227, 2001.

[158] D. Best, R. Hafen, B. Olsen, and W. Pike, "Atypical behavior identification in large-scale network traffic," in *Proceedings of the IEEE Symposium on Large Data Analysis and Visualization (LDAV)*, pp. 15–22, October 2011.

[159] L. Jun and C. Manikopoulos, "Network fault detection: classifier training method for anomaly fault detection in a production network using test network information," in *Proceedings of the 27th Annual IEEE Conference on Local Computer Networks (LCN 2002)*, pp. 473–482, November 2002.

[160] S. Oshima, Y. Ichimura, T. Nakashima, and T. Sueyoshi, "An anomaly detection system based on chi-square method with dynamic bin algorithm," in *Proceedings of the International Conference on Broadband and Wireless Computing, Communication and Applications (BWCCA)*, pp. 549–554, October 2011.

[161] L. Tao, Q. Ailing, H. Yuanbin, and C. Xintan, "Method for network anomaly detection based on bayesian statistical model with time slicing," in *Proceedings of the 7th World Congress on Intelligent Control and Automation (WCICA 2008)*, pp. 3359–3362, June 2008.

[162] K. Tabia, S. Benferhat, and Y. Djouadi, "A two-stage aggregation/thresholding scheme for multi-model anomaly-based approaches," in *Proceedings of the 33rd IEEE Conference on Local Computer Networks (LCN 2008)*, pp. 919–926, October 2008.

[163] G. Thatte, U. Mitra, and J. Heidemann, "Parametric methods for anomaly detection in aggregate traffic," *IEEE/ACM Transactions on Networking*, vol. 19, pp. 512–525, April 2011.

[164] Y. Yasami, M. Farahmand, and V. Zargari, "An arp-based anomaly detection algorithm using hidden markov model in enterprise networks," in *Proceedings of the 2nd*

International Conference on Systems and Networks Communications (ICSNC 2007), pp. 69–69, August 2007.

[165] X. Hoang and J. Hu, "An efficient hidden markov model training scheme for anomaly intrusion detection of server applications based on system calls," in *Proceedings of the 12th IEEE International Conference on Networks (ICON 2004)*, vol. 2, pp. 470–474, November 2004.

[166] W. Xiaotao, H. Houkuan, T. ShengFeng, i. Y. Xiaohu, and X. Baomin, "An online adaptive network anomaly detection model," in *Proceedings of the International Joint Conference on Computational Sciences and Optimization (CSO 2009)*, vol. 2, pp. 365–368, April 2009.

[167] Z. Kui, "A danger model based anomaly detection method for wireless sensor networks," in *Proceedings of the 2d International Symposium on Knowledge Acquisition and Modeling (KAM '09)*, vol. 1, pp. 11–14, November 2009.

[168] K. Bartos, M. Rehak, and V. Krmicek, "Optimizing flow sampling for network anomaly detection," in *Proceedings of the 7th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pp. 1304–1309, July 2011.

[169] D. Hongmei and R. Xu, "Model selection for anomaly detection in wireless ad hoc networks," in *Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining (CIDM 2007)*, pp. 540–546, March 2007.

[170] K. Claffy and T. Monk, "What's next for internet data analysis? status and challenges facing the community," *Proceedings of the IEEE*, vol. 85, pp. 1563–1571, October 1997.

[171] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao, "Overview and principles of internet traffic engineering." RFC3272, April 1969.

[172] A. Callado, C. Kamienski, G. Szabo, B. Gero, J. Kelner, S. Fernandes, and D. Sadok, "A survey on internet traffic identification," *IEEE Communications Surveys Tutorials*, vol. 11, no. 3, pp. 37–52, 2009.

[173] A. Peng, B. Eickhoff, T. Hey, and D. Lilja, "Toward consolidated tactical network architecture: A modeling and simulation study," in *Proceedings of the IEEE Military Communications Conference (MILCOM 2008)*, pp. 1–7, November 2008.

[174] K. Kalapriya, B. R. Raghucharan, A. Lele, and S. K. Nandy, "Dynamic traffic profiling for efficient link bandwidth utilization in qos routing," in *Proceedings of the 9th Asia-Pacific Conference on Communications (APCC 2003)*, vol. 2, pp. 486–493 Vol.2, September 2003.

[175] Y. Kim, J.-Y. Jo, and K. K. Suh, "Baseline profile stability for network anomaly detection," in *Proceedings of the 3rd International Conference on Information Technology: New Generations (ITNG '06)*, pp. 720–725, 2006.

[176] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. D. Kolaczyk, and N. Taft, "Structural analysis of network traffic flows," *SIGMETRICS Performance Evaluation Review*, vol. 32, pp. 61–72, June 2004.

[177] K. Xu, Z.-L. Zhang, and S. Bhattacharyya, "Internet traffic behavior profiling for network security monitoring," *IEEE/ACM Transactions on Networking*, vol. 16, pp. 1241–1252, December 2008.

[178] K. Xu, F. Wang, S. Bhattacharyya, and Z.-L. Zhang, "A real-time network traffic profiling system," in *Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN '07)*, pp. 595–605, June 2007.

[179] T.-H. Lee and J.-D. He, "Entropy-based profiling of network traffic for detection of security attack," in *Proceedings of the IEEE Region 10 Conference (TENCON 2009)*, pp. 1–5, January 2009.

[180] J. Maia and R. Holanda Filho, "Internet traffic classification using a hidden markov model," in *Proceedings of the 10th International Conference on Hybrid Intelligent Systems (HIS)*, pp. 37–42, August 2010.

[181] H. Hajji, "Baselining network traffic and online faults detection," in *Proceedings of the IEEE International Conference on Communications (ICC '03)*, vol. 1, pp. 301–308 vol.1, May 2003.

[182] X. Yang, "Designing traffic profiles for bursty internet traffic," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM '02)*, vol. 3, pp. 2149–2154 vol.3, November 2002.

[183] N. Garcia, P. Monteiro, and M. Freire, "Measuring and profiling ip traffic," in *Proceedings of the 4th European Conference on Universal Multiservice Networks (ECUMN '07)*, pp. 283–291, February 2007.

[184] T. Kisner, A. Essoh, and F. Kaderali, "Visualisation of network traffic using dynamic co-occurrence matrices," in *Proceedings of the 2nd International Conference on Internet Monitoring and Protection (ICIMP 2007)*, pp. 7–7, July 2007.

[185] J. Proença, MarioLemes, C. Coppelmans, M. Bottoli, and L. Souza Mendes, "Baseline to help with network management," in *e-Business and Telecommunication Networks* (J. Ascenso, L. Vasiu, C. Belo, and M. Saramago, eds.), pp. 158–166, Springer Netherlands, 2006.

[186] M. Lima, B. Zarpelao, L. Sampaio, J. Rodrigues, T. Abrao, and M. Proena, “Anomaly detection using baseline and k-means clustering,” in *Proceedings of the International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pp. 305–309, September 2010.

[187] W. Gardner and L. Franks, “Characterization of cyclostationary random signal processes,” *IEEE Transactions on Information Theory*, vol. 21, pp. 4–14, January 1975.

[188] A. Groth, “Estimation of periodicity in time series by ordinal analysis with application to speech,” November 2002.

[189] B. Fares, P. Apkarian, and D. Noll, “An augmented lagrangian method for a class of lmi-constrained problems in robust control theory,” in *Proceedings of the American Control Conference*, vol. 5, pp. 3702–3706, 2000.

[190] S. Wold, K. Esbensen, and P. Geladi, “Principal Component Analysis,” *Chemometrics and Intelligent Laboratory Systems*, vol. 2, no. 1, pp. 37–52, 1987.

[191] B. B. Zarpelão, L. D. S. Mendes, and M. L. Proença, Jr., “Anomaly detection aiming pro-active management of computer network based on digital signature of network segment,” *Journal of Network and Systems Management*, vol. 15, pp. 267–283, June 2007.

[192] A. Mansy and C. Dovrolis, “The mythical traffic matrix,” 2006.

[193] L. Yong, C. Yuanli, and H. Yongxuan, “Kalman filtering based dynamic od matrix estimation and prediction for traffic systems,” in *Proceedings of the IEEE Intelligent Transportation Systems*, vol. 2, pp. 1515–1520 vol.2, October 2003.

[194] T. Goldstein and S. Osher, “The split bregman method for L1-regularized problems,” *SIAM Journal on Imaging Sciences*, vol. 2, no. 2, pp. 323–343, 2009.

[195] W. Yin, S. Osher, D. Goldfarb, and J. Darbon, “Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing,” *SIAM Journal on Imaging Sciences*, vol. 1, no. 1, pp. 143–168, 2008.

APPENDIX A

Algorithm Implementations

A.1. Bregman L_1 minimization

Bregman iterations [194] is applied to the problem

$$\min_u J(u) + H(u). \tag{A.1}$$

Then the iterations are shown in Fig. A.1 where D is the Bregman distance given by

```
Initialize: $k \leftarrow 0, u^0 \leftarrow 0, p^0 \leftarrow 0$
while not converged do
    $u^{k+1} \leftarrow \arg\min_u D_J^{p^k}\left(u, u^k\right) + H(u)$
    $p^{k+1} \leftarrow p^k - \nabla H\left(u^{k+1}\right) \in \partial J\left(u^{k+1}\right)$
    $k \leftarrow k + 1$
end while
```

Figure A.1. Bregman iterations

$D_J^p(u, v) = J(u) - J(v) - < p, u - v >$.

As shown in [195] for L_1 minimization $J(u) = \mu\|u\|_1$ and $H(u) = \frac{1}{2}\|Au - f\|^2$.

A.2. Doughlas-Rachford L_1 minimization

$$\min_x F(x) + G(x) \quad \text{where} \quad F(x) = i_C(x), G(x) = \|x\|_1, C = \{x : \Phi x = y\}. \tag{A.2}$$

Then the Douglas-Rachford iterations are

$$\tilde{x}_{k+1} = \left(1 - \frac{\mu}{2}\right)\tilde{x}_k + \frac{\mu}{2} r Prox_{\gamma G}\left(r Prox_{\gamma F}\left(\tilde{x}_k\right)\right) \tag{A.3}$$

$$x_{k+1} = Prox_{\gamma F}\left(\tilde{x}_{k+1}\right) \tag{A.4}$$

Then

$$rProx_{\gamma F}(x) = 2Prox_{\gamma F}(x) - x \tag{A.5}$$

$$Prox_{\gamma F}(x) = \arg\min_y \frac{1}{2}\|x - y\|^2 + \gamma F(y) \tag{A.6}$$

A.3. Douglas-Rachford L_* minimization

Using [55] nuclear norm minimization can be written as

$$\min_x F(x) + G(x) \quad \text{where} \quad F(x) = i_C(x), G(x) = \|x\|_*, C = \{x : \Phi x = y\}. \tag{A.7}$$

Then the Douglas-Rachford iterations are

$$\tilde{x}_{k+1} = \left(1 - \frac{\mu}{2}\right)\tilde{x}_k + \frac{\mu}{2} rProx_{\gamma G}\left(rProx_{\gamma F}\left(\tilde{x}_k\right)\right) \tag{A.8}$$

$$x_{k+1} = Prox_{\gamma F}\left(\tilde{x}_{k+1}\right) \tag{A.9}$$

Then

$$rProx_{\gamma F}(x) = 2Prox_{\gamma F}(x) - x \tag{A.10}$$

$$Prox_{\gamma F}(x) = \arg\min_y \frac{1}{2}\|x - y\|^2 + \gamma F(y) = x + \Phi^*(y - \Phi x) \tag{A.11}$$

A.4. Principal Component Pursuit

The following algorithm implements RPCA via inexact ALM [119]. It decomposes the input matrix D into low-rank A and sparse E.

```
Initialize: $Y_0 = D/J(D), E_0 = 0, \mu_0 > 0, \rho > 1, k = 0$
**while** not converged **do**
    $(U, S, V) \leftarrow svd\left(D - E_k + \mu_k^{-1} Y_k\right)$
    $A_{k+1} \leftarrow U(S)_{\mu_k^{-1}}(S)V^T$
    $E_{k+1} \leftarrow \mathcal{S}_{\lambda\mu_k^{-1}}\left(D - A_{k+1} + \mu_k^{-1} Y_k\right)$
    $Y_{k+1} \leftarrow Y_k + \mu_k\left(D - A_{k+1} - E_{k+1}\right)$
    Update $\mu_k$ to $\mu_{k+1}$
    $k \leftarrow k + 1$
**end while**
```

FIGURE A.2. RPCA via the inexact ALM method

A.5. MATCHING PURSUIT L_0 MINIMIZATION

CoSaMP algorithm listed in [48] takes in a sampling matrix Φ, noisy sample vector u and a sparsity level s and perform Algorithm A.3.

```
Initialize: $a^0 \leftarrow 0, v \leftarrow u, k \leftarrow 0$
**while** not converged **do**
    $k \leftarrow k + 1$
    $y \leftarrow \Phi^* v$
    $\Omega \leftarrow supp(y_{2s})$
    $T \leftarrow \Omega \cup supp(a^{k-1})$
    $b_{|T} \leftarrow \Phi^{\dagger} u$
    $b_{|T^c} \leftarrow 0$
    $a^k \leftarrow b_s$
    $v \leftarrow u - \Phi a^k$
**end while**
```

FIGURE A.3. CoSaMP algorithm

APPENDIX B

Software Tools

B.1. GUI for Robust PCA Recoverability Experiments

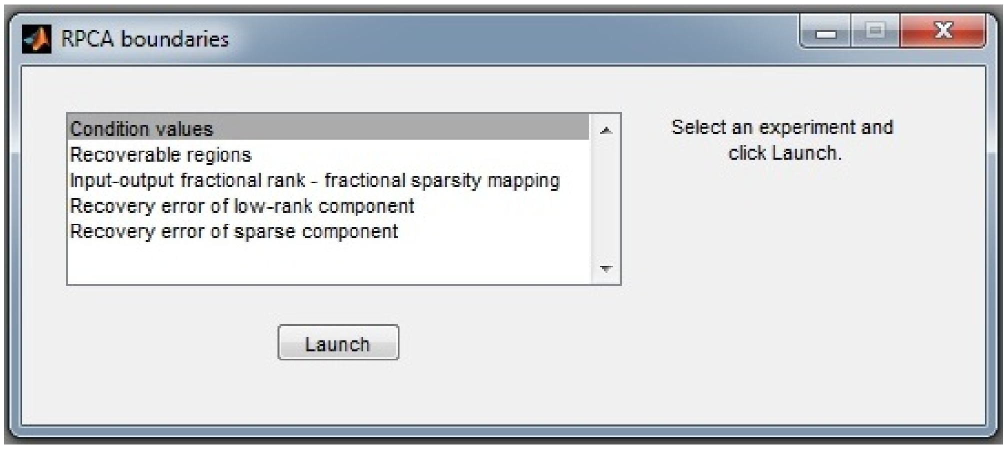
`http://www.cnrl.colostate.edu/Projects/NetworkDataAnalysis/software1.html`

Permanent URL: `http://hdl.handle.net/10217/89321`

The GUI provides the following functionalities:

- Evaluate sufficient conditions for recovery over a selected range of ranks and sparsities, size, low-rank and sparse matrix types.
- Recoverable region for a selected range fractional sparsities, size, low-rank and sparse matrix types.
- Input - output mapping between fractional-ranks fractional-sparsities.
- Recovery error of the low-rank component.
- Recovery error of the sparse component.

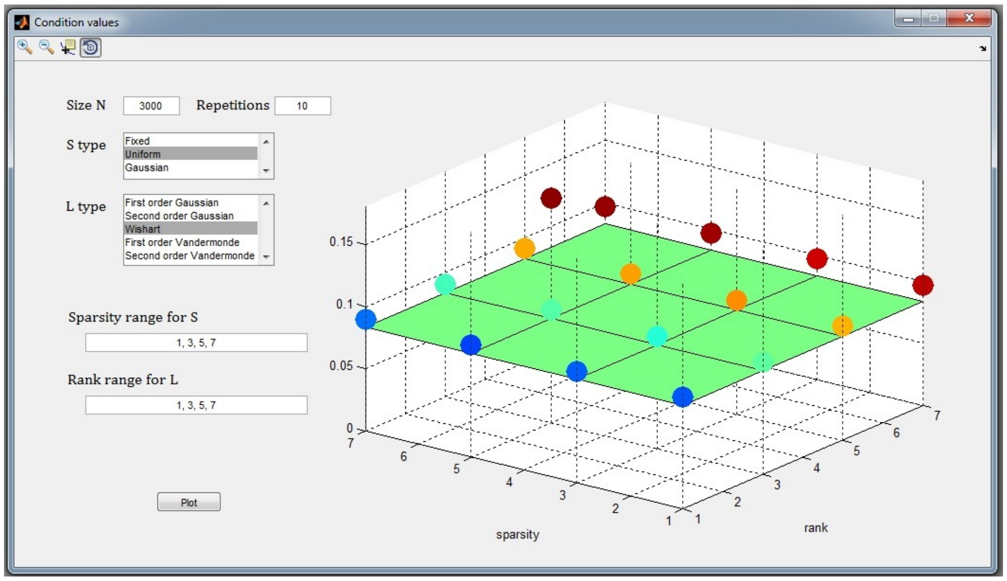
The software reproduces experiments listed in [6]. The sufficient conditions used are from [6]. Matrix constructions and further details of the experiments can be found in Section 5.2.3.1.


RPCA boundaries
Condition values
Recoverable regions
Input-output fractional rank - fractional sparsity mapping
Recovery error of low-rank component
Recovery error of sparse component
Launch

Select an experiment and click Launch.

Figure B.1. Main GUI

B.1.1. Main GUI. Figure B.1 is portal to the GUIs of each experiment. User will select experiments and click Launch open a separate GUI for the experiment. When clicked on an experiment, a brief description of the experiment will be displayed.



Condition values

Size N 3000 Repetitions 10

S type: Fixed, Uniform (selected), Gaussian

L type: First order Gaussian, Second order Gaussian, Wishart (selected), First order Vandermonde, Second order Vandermonde

Sparsity range for S: 1, 3, 5, 7

Rank range for L: 1, 3, 5, 7

Figure B.2. Sufficient condition values

B.1.2. Sufficient Condition Values. Dots in Fig. B.2 indicate the condition values of the sufficient conditions presented in [6]. The green surface indicate the condition threshold 1/12. Users select the matrix size to test, the number of realizations, sparse and low-rank matrix type and the ranges for sparsity and ranks.

B.1.3. Recoverable Regions. Fractional-rank and fractional-sparsity combinations below the curves in Fig. B.3 are 100% recoverable for corresponding matrix size. Users select the sparse and low-rank matrix type, the range of fractional-sparsities to test and the list of matrix sizes to test. The users also specify the number of repetitions to be made to verify 100% recoverability.

B.1.4. Input-Output Fractional-rank Fractional-sparsity Mapping. The mapping between the input pair of fractional rank of the low-rank component, fractional-sparsity

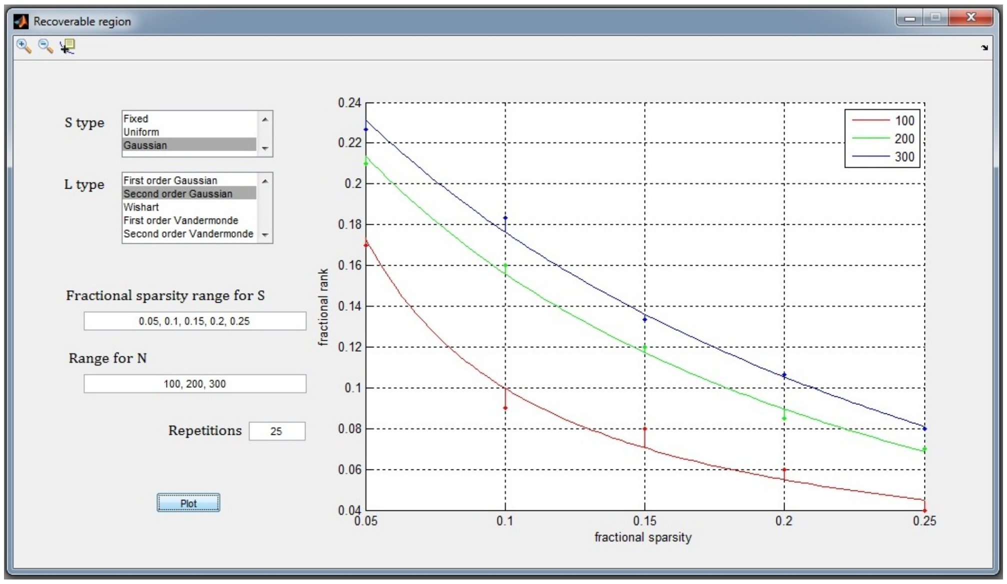


FIGURE B.3. Recoverable regions

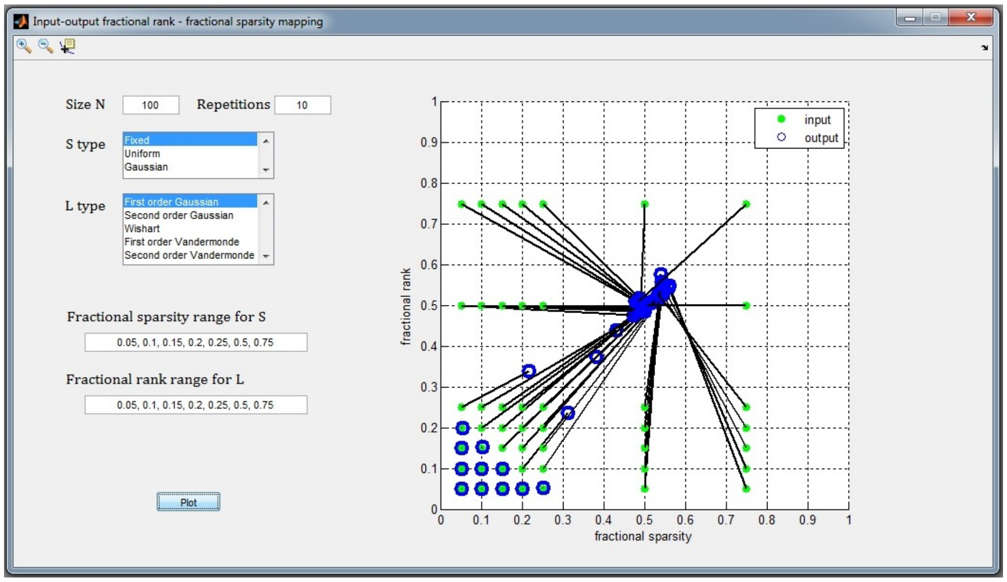
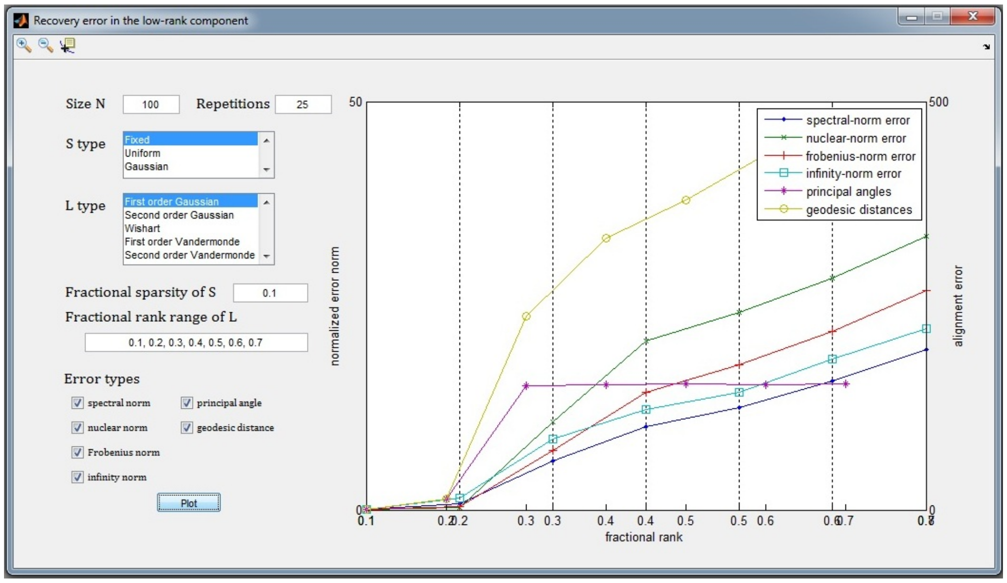


FIGURE B.4. Input-output fractional-rank fractional-sparsity mapping

of the sparse component and the output pair of fractional rank of the low-rank component, fractional-sparsity of the sparse component is shown in Fig. B.4. Results indicate that failed recoveries tend to result in a half-rank low-rank component and half-sparse sparse component. Users specify the matrix size to test, the number of repetitions, sparse and low-rank matrix types, and ranges of fractional-ranks and fractional-sparsities.


Recovery error in the low-rank component

Size N 100 Repetitions 25

S type: [✓ Fixed] Uniform Gaussian

L type: [✓ First order Gaussian] Second order Gaussian Wishart First order Vandermonde Second order Vandermonde

Fractional sparsity of S 0.1

Fractional rank range of L 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7

Error types

[✓] spectral norm [✓] principal angle

[✓] nuclear norm [✓] geodesic distance

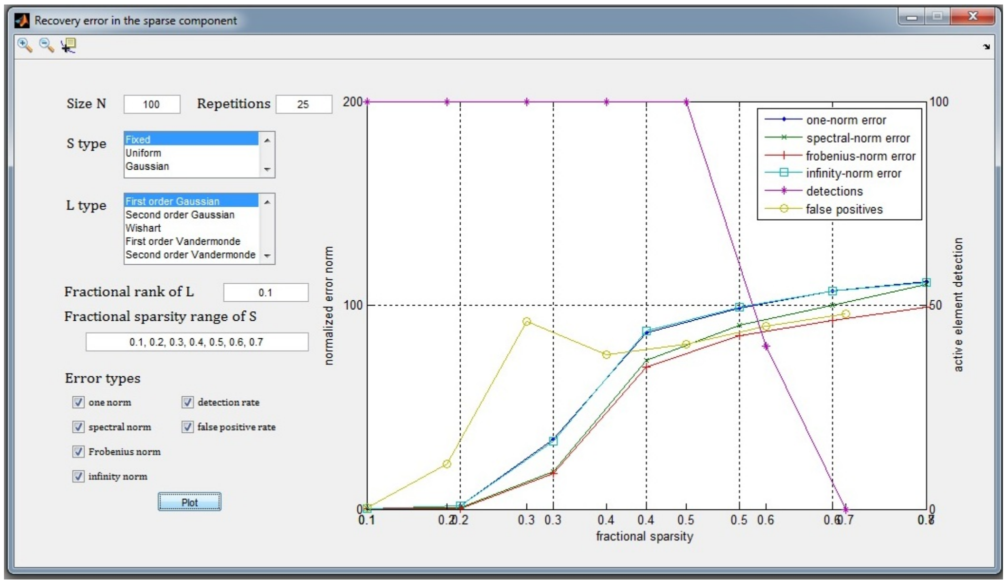
[✓] Frobenius norm

[✓] infinity norm

Figure B.5. Recovery error of the low-rank component

B.1.5. Recovery Error of the Low-rank Component. In Fig. B.5 the growth of error in recovering the low-rank component for a range of fractional-ranks with sparsity of the sparse component kept fixed. User specifies the matrix size to test, number of repetitions, type of the sparse and low-rank components, the fixed fractional-sparsity of the sparse component, the range of fractional-ranks of the low-rank components, and the types of the errors. Two types of errors can be found: normalized norm errors and alignment errors. Normalized norm errors use spectral, nuclear, Frobenius and infinity norms. Alignment errors check principal angles and geodesic distances.

B.1.6. Recovery Error of the Sparse Component. In Fig. B.5 the growth of error in recovering the sparse component for a range of fractional-sparsities with rank of the low-rank component kept fixed. User specifies the matrix size to test, number of repetitions, type of the sparse and low-rank components, the fixed fractional-rank of the low-rank component, the range of fractional-sparsities of the sparse components, and the types of the errors. Two types of errors can be found: normalized norm errors and detection errors.


Recovery error in the sparse component

Size N: 100 Repetitions: 25

S type: [✓ Fixed] Uniform Gaussian

L type: [✓ First order Gaussian] Second order Gaussian Wishart First order Vandermonde Second order Vandermonde

Fractional rank of L: 0.1

Fractional sparsity range of S: 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7

Error types

- [✓] one norm
- [✓] detection rate
- [✓] spectral norm
- [✓] false positive rate
- [✓] Frobenius norm
- [✓] infinity norm

FIGURE B.6. Recovery error of the sparse component

Normalized norm errors use one, spectral, Frobenius and infinity norms. Alignment errors check detection rate and false positive rate.

B.2. GUI for Robust Principal Component Analysis

`http://www.cnrl.colostate.edu/Projects/NetworkDataAnalysis/software2.html`

Permanent URL: `http://hdl.handle.net/10217/89321`

The GUI performs Robust PCA on:

- Synthesized low-rank and sparse matrix additions.
- Data from external experiments.

Matrix constructions details can be found in Section 5.2.3.1.

B.2.1. Generate test matrices.

Synthesizes test matrices of a specified sized. Test matrices are a sum of a low-rank matrix and a sparse matrix. The users set the types and fractional sparsity and ranks of the low-rank and sparse matrices.

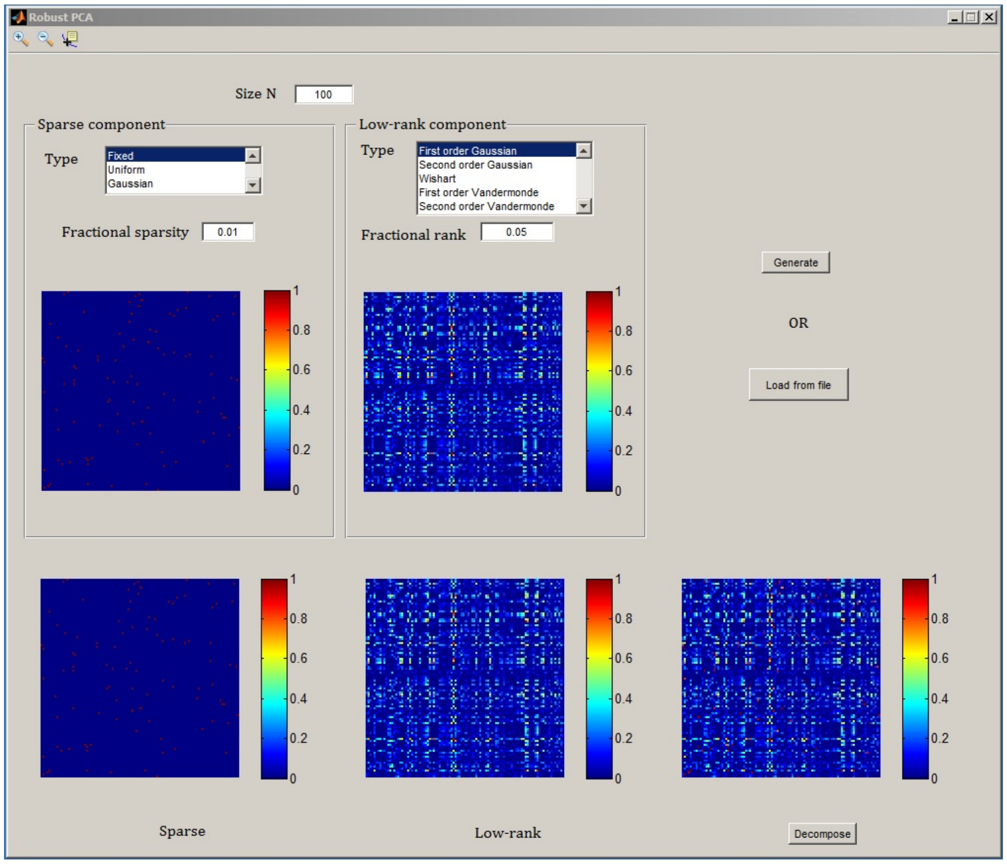


FIGURE B.7. GUI for Robust Principal Component Analysis

B.2.2. LOAD TEST MATRICES. Users may analyze test matrices from an external experiment. The matrices are to be formatted as follows.

- Separate row by new lines.
- Separate columns (entries of a row) by commas.

E.g.:

2.5, 3.0, 1.2

6.2, 2.1, -7.8

-9.7, 4.5, 2.0

B.3. RANDOM MATRIX GENERATOR

`http://www.cnrl.colostate.edu/Projects/NetworkDataAnalysis/software3.html`

Permanent URL: `http://hdl.handle.net/10217/89321`

The software sythesizes random realizations of low-rank and sparse matrices of specified types. These software is used in, and the matrix constructions details can be found inSection 5.2.3.1.

B.3.1. Synopsis. Input arguments: LS typ n frs reps fname

LS: specify either low-rank or sparse. Use L or l to indicate low-rank matrices, and S or s to indicate sparse matrices.

typ: indicates the type of the matrices. Types available for low-rank are:

(1) First order Gaussian
(2) Second order Gaussian
(3) Wishart
(4) First order Vandermonde
(5) Second order Vandermonde

Type available for sparse matrices are:

(1) Fixed
(2) Uniform
(3) Gaussian

n: size of the matrices.

frs: fractional rank or fractional sparsity of the matrix.

reps: number of realizations of the specified matrix.

fname: base file name of the matrices. reps many files will be produced each containing a comma-delimited realization of the specified matrix. E.g.: If fname is testfile and reps is 3, then three output files with names testfile.1, testfile.2, and testfile.3 will be produced.

B.3.2. Examples. DOS: genRandMats-1.0.exe L 1 20 0.1 3 testfile

Linux: ./run_genRandMatix_v01a.sh $MCR_DIRECTORY L 1 20 0.1 3 testfile

B.4. MATALB® Toolkit for Network Traffic Anomaly Analysis

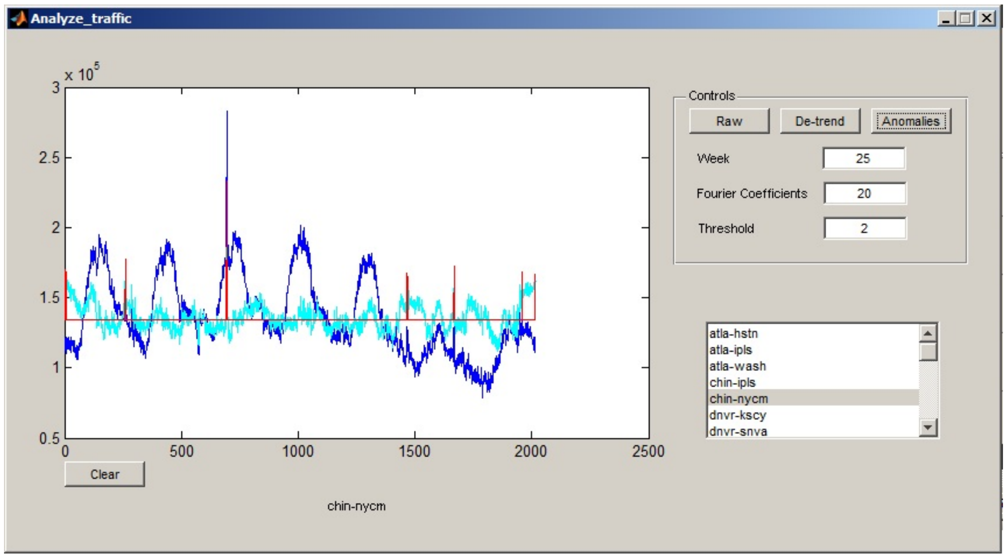
`http://www.engr.colostate.edu/~vwb/anom/`

Permanent URL: `http://hdl.handle.net/10217/89321`

The toolkit performs:

- De-trending and thresholding for anomaly detection
- Graph wavelet based summarizing and anomaly tracing
- Distribution fitting to spatial and temporal parameters
- Simulator/Emulator to regenerate statistically similar anomalies

The toolkit is developed for Internet2 dataset, but customizable for other datasets.



Week 25

Fourier Coefficients 20

Threshold 2

Figure B.8. Anomaly detection

B.4.1. Screenshots.

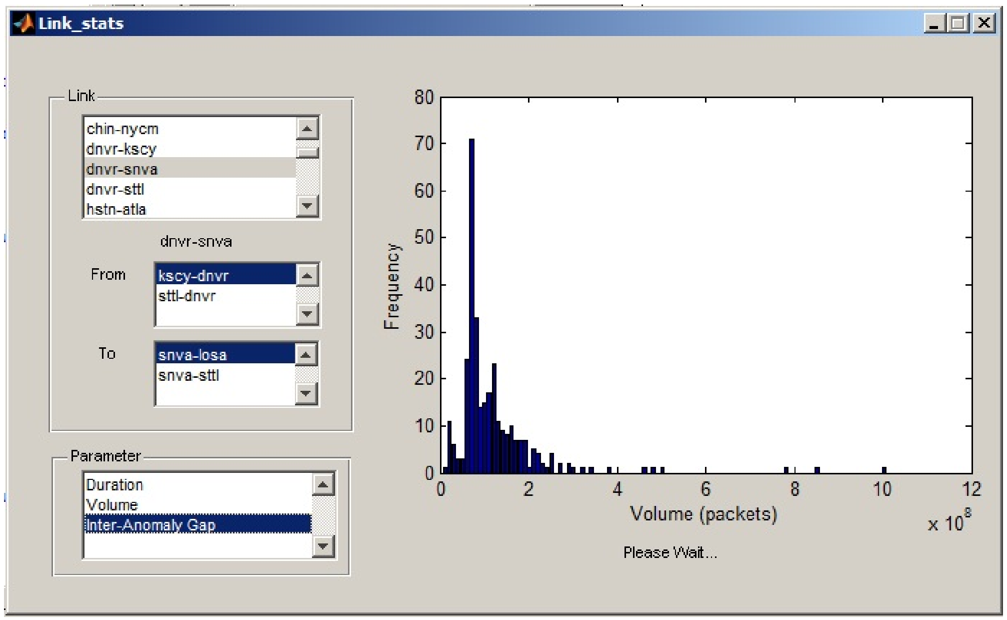


FIGURE B.9. Activity on a link

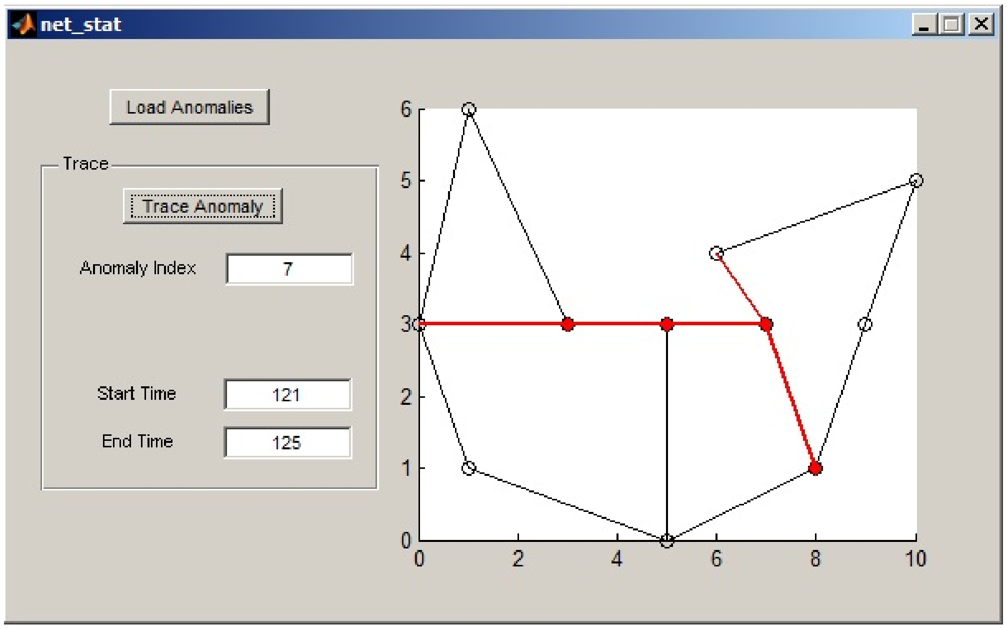
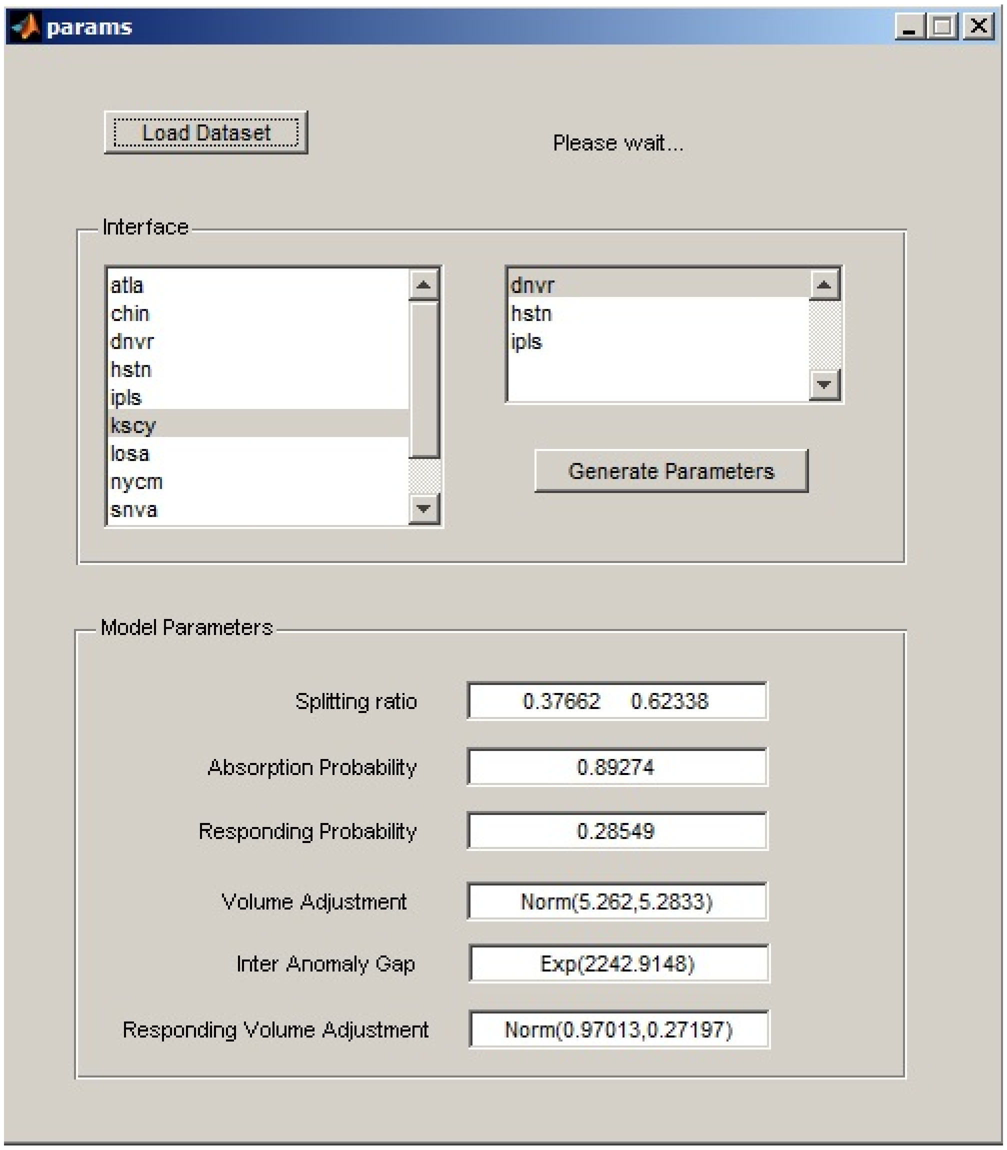


FIGURE B.10. Activity over the network



Splitting ratio	0.37662 0.62338
Absorption Probability	0.89274
Responding Probability	0.28549
Volume Adjustment	Norm(5.262,5.2833)
Inter Anomaly Gap	Exp(2242.9148)
Responding Volume Adjustment	Norm(0.97013,0.27197)

Figure B.11. Estimating parameters

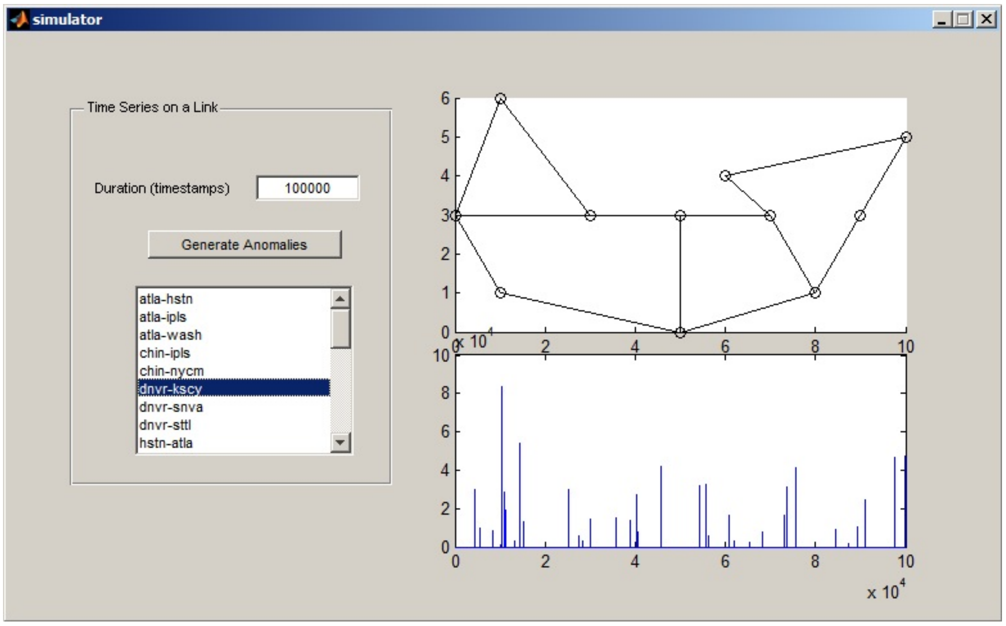


FIGURE B.12. Simulator

APPENDIX C

Source Codes

C.1. Source Codes for Robust PCA Experiments

C.1.1. lagrangian.m. The following code performs RPCA decomposition of input matrix. i.e., solves

$$\arg\min_{L,S} ||L||_* + \lambda||\mathcal{S}_\epsilon(S)||_1 \tag{C.1}$$

$$\text{s.t.} \quad M - L - S = 0 \tag{C.2}$$

Synopsis of usage is:

```
 [L,S,iter] =
lagrangian(M, epsilon, tol, maxIter, lambda-Scale, lambda, rho, mu)
```

where `M` is the only mandatory argument.

```
function [L,S,iter]=lagrangian(M,varargin)
% [L,S,iter] =
% LAGRANGIAN(M, [epsilon, tol, maxIter, lambda-Scale, lambda, rho, mu])
%
% Solve
% argmin_{L,S} ||L||_* + lambda.||shrink(epsilon,S)||_1
% s.t. M-L-S=0

% Copyrights (C) Vidarshana Bandara
% Computer Networking Research Laboratory,
% Department of Electrical & Computer Engineering,

% Colorado State University,

% Fort Collins, CO 80523, USA



% use defaults for unspecified arguments

switch nargin

    case 1

        epsilon=zeros(size(M));

        tol=1e-6;

        maxIter=100;

        lambdaScale=1.0;

        lambda=1.0/sqrt(max(size(M)));

        rho=1.1;

        mu=1.25/norm(M);


    case 2

        epsilon=varargin{1};

        tol=1e-6;

        maxIter=100;

        lambdaScale=1.0;

        lambda=1.0/sqrt(max(size(M)));

        rho=1.1;

        mu=1.25/norm(M);


    case 3
```

```
        epsilon=varargin{1};

        tol=varargin{2};

        maxIter=100;

        lambdaScale=1.0;

        lambda=1.0/sqrt(max(size(M)));

        rho=1.1;

        mu=1.25/norm(M);


    case 4

        epsilon=varargin{1};

        tol=varargin{2};

        maxIter=varargin{3};

        lambdaScale=1.0;

        lambda=1.0/sqrt(max(size(M)));

        rho=1.1;

        mu=1.25/norm(M);


    case 5

        epsilon=varargin{1};

        tol=varargin{2};

        maxIter=varargin{3};

        lambdaScale=varargin{4};

        lambda=1.0/sqrt(max(size(M)));

        rho=1.1;

        mu=1.25/norm(M);
```

```
case 6

    epsilon=varargin{1};

    tol=varargin{2};

    maxIter=varargin{3};

    lambdaScale=varargin{4};

    lambda=varargin{5};

    rho=1.1;

    mu=1.25/norm(M);


case 7

    epsilon=varargin{1};

    tol=varargin{2};

    maxIter=varargin{3};

    lambdaScale=varargin{4};

    lambda=varargin{5};

    rho=varargin{6};

    mu=1.25/norm(M);


case 8

    epsilon=varargin{1};

    tol=varargin{2};

    maxIter=varargin{3};

    lambdaScale=varargin{4};

    lambda=varargin{5};

    rho=varargin{6};

    mu=varargin{7};
    otherwise
        warning('error in arguments');
        return
end

% sanity check
% epsilon
if size(epsilon)~=size(M)
    warning('dimensions of epsilon should match M');
    return
end
% tol
if length(tol)~=1 || tol<0
    warning('tol should be a positive scalar');
    return
end
% maxIter
if length(maxIter)~=1 || maxIter<0
    warning('maximum iterations should be a positive scalar');
    return
end
% lambda-scale
if length(lambdaScale)~=1 || lambdaScale<0
    warning('lambda-Scale should be a positive scalar');
    return
```

```
end

% lambda

if length(lambda)~=1

    warning('lambda should be a scalar');

    return

end

% rho

if length(rho)~=1 || rho<0

    warning('rho should be a positive scalar');

    return

end

% mu

if length(mu)~=1 || mu<0

    warning('mu should be a positive scalar');

    return

end


% initialize

iter=0;

S=zeros(size(M));

Y=zeros(size(M));


% solve

while 1

    iter=iter+1;

    % alternating directions
    L = lagrangeL(M,S,mu,Y);

    S = lagrangeS(M,L,mu,Y,lambda,epsilon);


    % updates

    Y = Y + mu*(M-L-S);

    mu = mu * rho;


    % check convergence

    error = norm(M-L-S,'fro')/norm(M,'fro');

    if error < tol

        break

    end


    % break long loops

    if iter>maxIter

        break

    end

end

% shrink S before returning

S = sign(S) .* max( abs(S)-epsilon , zeros(size(S)) );




function L=lagrangeL(M,S,mu,Y)

% argmin_L {lagrangian}


X = M - S + (1.0/mu)*Y;
```

```
[U,sigma,V] = svd(X);

% shrink singular values
sigma = sign(sigma) .* max( abs(sigma)-(1.0/mu) , zeros(size(sigma)) );

% construct L
L = U*sigma*V';



function S=lagrangeS(M,L,mu,Y,lambda,epsilon)
% argmin_S {lagrangian}

X = M - L + (1.0/mu)*Y;
S1 = X -(lambda/mu);
S2 = X;
S3 = X + (lambda/mu);

[m,n] = size(M);
S = zeros(m,n);

for i=1:m
    for j=1:n
        if epsilon(i,j) < S1(i,j)
            S(i,j)=S1(i,j);
        elseif -epsilon(i,j) < S2(i,j) && S2(i,j) < epsilon(i,j)
            S(i,j) = S2(i,j);
        elseif S3(i,j) < -epsilon(i,j)
            S(i,j) = S3(i,j);
        else
            tmp1 = Y(i,j) * ( M(i,j) - L(i,j) - epsilon(i,j) ) + ...
                (mu/2.0) * ( M(i,j) - L(i,j) - epsilon(i,j) )^2;
            tmp2 = Y(i,j) * ( M(i,j) - L(i,j) + epsilon(i,j) ) + ...
                (mu/2.0) * ( M(i,j) - L(i,j) + epsilon(i,j) )^2;
            if tmp1 < tmp2
                S(i,j) = epsilon(i,j);
            else
                S(i,j) = -epsilon(i,j);
            end
        end
    end
end
```

C.1.2. GETSPARSE.M. The code below produces a sparse matrix of given sparsity.

Synopsis:

`S = getSparse(sp,N,[M,typ])`

where

- `typ` 1 : fixed (default)
- `typ` 2 : uniform
- `typ` 3 : Gaussian

First two arguments are mandatory.

```
function S = getSparse(sp,N,varargin)
% produce a sparse matrix of given sparsity
% S = getSparse_v02b(sp,N,[M,typ])
% typ 1 : fixed (default)
% typ 2 : uniform
% typ 3 : gaussian

% default
typ=1;

% input args
switch nargin
    case 2
        M = N;
    case 3
        M = varargin{1};
    case 4
        M = varargin{1};
        typ = varargin{2};
    otherwise
        S=[];
        return
end

% support
if isempty(M)
```

```
    M=N;
end
locs=randperm(N*M,int32(sp));
S = zeros(N,M);

% element values
if typ == 1
    % fixed
    S(locs)=1;
elseif typ == 2
    % unif
    S(locs)=2*rand(sp,1)-1;
elseif typ == 3
    % gaussian
    S(locs)=randn(sp,1);
else
    S = [];
end
```

C.1.3. GETLOWRANK.M. The code below constructs a low rank matrix.

Synopsis:

```
 L = getLowRank(n,r,typ,U1,Sig1h,U2,Sig2h)
```

where

- `typ` 1 : 1st order Gaussian
- `typ` 2 : 2nd order Gaussian
- `typ` 3 : Wishart

- `typ` 4 : 1st order Vandermonde
- `typ` 5 : 2nd order Vandermonde

All arguments are mandatory.

```
function L = getLowRank(n,r,typ,U1,Sig1h,U2,Sig2h)
% construct low rank matrices
% 1. 1st ord Gauss
% 2. 2nd ord Gauss
% 3. Wishart
% 4. 1st ord Vandermonde
% 5. 2nd ord Vandermonde

L=[];

% adjust Sigma_1
Sig1h(r+1:n)=0;
Sig1h=diag(Sig1h);
% Sigma_2 adjusted on demand

% 1. 1st ord Gauss
if typ==1
    L=U1*Sig1h*randn(n);
end

% 2. 2nd ord Gauss
if typ==2
```

```
    Sig2h(r+1:n)=0; Sig2h=diag(Sig2h);

    L=U1*Sig1h*randn(n)*randn(n)'*Sig2h'*U2';

end


% 3. Wishart

if typ==3

    X=U1*Sig1h*randn(n);

    L=X*X';

end


% U1 is Z for Vandermondes


% 4. 1st ord Vandermonde

if typ==4

    L=U1*Sig1h*randn(n);

end


% 5. 2nd ord Vandermonde

if typ==5

    X=U1*Sig1h*randn(n);

    L=X*X';

end
```

C.1.4. genG.m, genW.m, and genZ.m. The following codes produce row and columns spaces for Gaussian, Wishart and Vandermonde matrices.

```
% Generate Cov for 2nd ord. Gauss - don't re-run

n=100;  % size

% iid N(0,1)
X1=randn(n);
X2=randn(n);

% svd
[U1,~,~]=svd(X1);
[U2,~,~]=svd(X2);

% spectra
Sig1h=sqrt(rand(n,1));
Sig2h=sqrt(rand(n,1));

% DON'T RUN THIS LINE
save n100G U1 U2 Sig1h Sig2h



% Generate Cov for Wishart - don't re-run

n=100;  % size
fName=['n' num2str(n) 'W.mat'];

% iid N(0,1)
```

```
X1=randn(n);

% svd
[U1,~,~]=svd(X1);

% spectra
Sig1h=sqrt(rand(n,1));

% DON'T RUN THIS LINE
save(fName,'U1','Sig1h');



% Vandermonde matrices

clear all
close all
clc

n=500;

tmp1=2*pi*linspace(0,1,n+1);
tmp2=cos(tmp1)+sin(tmp1)*1i;
alphaI=tmp2(1:n);
Z=ones(n,n);
for i=1:n
    for j=1:n
```

```
        Z(i,j)=alphaI(j)^(i-1);
    end
end

% spectra
Sig1h=sqrt(rand(n,1));

save(['n' num2str(n) 'Z.mat'],'Z','Sig1h');
```

Source codes and binaries developed in this work, including the codes listed in Appendix C, are deposited on the permanent URL: `http://hdl.handle.net/10217/89321`