

Anomaly Detection in Heterogeneous Data Environments with Applications to Mechanical  
Engineering Signals & Systems

Michael W. Milo

Dissertation submitted to the faculty of the Virginia Polytechnic Institute and State University in  
partial fulfillment of the requirements for the degree of:

Doctor of Philosophy  
In  
Mechanical Engineering

Michael J. Roan  
Ricardo A. Burdisso  
Alfred L. Wicks  
Pang Du  
Martin E. Johnson

October 4, 2013  
Blacksburg, Virginia, United States of America

Keywords: Anomaly Detection, Mechanical Systems, Bayesian Statistics, Pattern Recognition,  
Simulation Model, Heterogeneous Data, Statistical Analysis, Data Mining

Copyright 2013

# Anomaly Detection in Heterogeneous Data Environments with Applications to Mechanical Engineering Signals & Systems

Michael W. Milo

## ABSTRACT

Anomaly detection is a relevant problem in the field of Mechanical Engineering, because the analysis of mechanical systems often relies on identifying deviations from what is considered “normal”. The mechanical sciences are represented by a heterogeneous collection of data types: some systems may be highly dimensional, may contain exclusively spatial or temporal data, may be spatiotemporally linked, or may be non-deterministic and best described probabilistically. Given the broad range of data types in this field, it is not possible to propose a single processing method that will be appropriate, or even usable, for all data types. This has led to human observation remaining a common, albeit costly and inefficient, approach to detecting anomalous signals or patterns in mechanical data.

The advantages of automated anomaly detection in mechanical systems include reduced monitoring costs, increased reliability of fault detection, and improved safety for users and operators. This dissertation proposes a hierarchical framework for anomaly detection through machine learning, and applies it to three distinct and heterogeneous data types: state-based data, parameter-driven data, and spatiotemporal sensor network data. In time-series data, anomaly detection results were robust in synthetic data generated using multiple simulation algorithms, as well as experimental data from rolling element bearings, with highly accurate detection rates (>99% detection, <1% false alarm). Significant developments were shown in parameter-driven data by reducing the sample sizes necessary for analysis, as well as reducing the time required for computation. The event-space model extends previous work into a geospatial sensor network

and demonstrates applications of this type of event modeling at various timescales, and compares the model to results obtained using other approaches.

Each data type is processed in a unique way relative to the others, but all are fitted to the same hierarchical structure for system modeling. This hierarchical model is the key development proposed by this dissertation, and makes both novel and significant contributions to the fields of mechanical analysis and data processing. This work demonstrates the effectiveness of the developed approaches, details how they differ from other relevant industry standard methods, and concludes with a proposal for additional research into other data types.

## **DEDICATION**

This work is dedicated to my parents, Bill and Janet Milo, whose love and support made my continuing education possible. Thank you so much for believing in me.

## **ACKNOWLEDGEMENTS**

I would like to acknowledge the contribution of several key colleagues and educators who made this work possible. First, I would like to thank my advisor Dr. Michael Roan for his mentorship and guidance during my Doctorate program. I recognize colleagues Bradley Harris, Benjamin Bjerke, Phillip Chin, Dr. Benjamin Smith, and Dr. Elizabeth Hoppe for their contributions as research team members on multiple projects. I would also like to thank those faculty not enumerated on my committee list, but whose intellectual contributions to this research were significant: Dr. Leanna House of Virginia Tech, and Dr's K. David Wilson and Lindamae Peck of the U.S. Army Cold Regions Research and Engineering Lab (CRREL). Dr. Amol Khatkhate of Penn State University offered assistance in the evaluation of data preprocessing methods and wavelet transforms for the dataset addressed in Chapter 3. The experimental bearing-fault data used in Chapter 3 was originally obtained by Dr. Kenneth A. Loparo of Case Western Reserve University. The US Army Cold Regions Research and Engineering Laboratory (CRREL) coordinated the experiment and performed data collection for the dataset addressed in Chapter 5. Funding for this experiment was provided by the US Army Engineer Research and Development Center (ERDC) Geospatial Research and Engineering business area. Finally, I recognize the support of my family and friends for their assistance in proofreading and discussion of elements contained in this dissertation.

# TABLE OF CONTENTS

ABSTRACT.....	ii
DEDICATION.....	iv
ACKNOWLEDGEMENTS.....	iv
TABLE OF CONTENTS.....	v
LIST OF FIGURES.....	x
LIST OF TABLES.....	xiii
LIST OF ABBREVIATIONS.....	xiv
LIST OF SYMBOLS.....	xv
1 INTRODUCTION.....	1
1.1 Anomaly Detection.....	1
1.1.1 Problem Definition.....	1
1.1.2 Applications.....	2
1.1.3 Presentation.....	3
1.1.4 Approaches.....	5
1.1.5 Evaluation.....	6
1.1.6 Stochastic Processes.....	8
1.2 General Techniques.....	9
1.2.1 Symbolic Aggregate Approximation.....	9
1.2.2 Principal Component Analysis & Eigenvalue Decomposition.....	11
1.2.3 Pattern Recognition using PCA and SVD.....	15
1.2.4 Clustering Algorithms.....	16
1.2.5 Bayesian Statistics.....	18

1.2.6	Markov Chains.....	19
1.2.7	Sequential Updating.....	23
1.2.8	Autoregressive Moving Average Models.....	25
1.2.9	Neural Networks.....	28
1.2.10	Hierarchical Approaches.....	32
2	PROBLEM DEFINITIONS AND LITERATURE REVIEW.....	34
2.1	Anomaly Detection in Time-Series Data.....	34
2.1.1	Previous approaches using Transition Matrices.....	34
2.1.2	Previous approaches using Neural Networks.....	40
2.2	Anomaly Detection in Parameter-Driven Data.....	41
2.2.1	Approach Background and Survey.....	42
2.2.2	CUSUM algorithm background.....	43
2.2.3	Limitations of Point Change Detection.....	44
2.3	Anomaly Detection in Spatiotemporal Data.....	45
2.3.1	Network Traffic Analysis.....	47
2.3.2	Discrete Convolution Approaches.....	47
2.4	Anomaly Detection in Rotating Machinery.....	48
3	ANOMALY DETECTION IN TIME-SERIES DATA.....	52
3.1	Introduction.....	52
3.2	Background.....	53
3.2.1	Discrete Wavelet Transform (DWT).....	53
3.2.2	Symbolic Aggregate Approximation (SAX).....	54
3.2.3	Hidden Markov Models (HMMs) and Transition Matrices (TMs).....	56

3.2.4	Markov Chain Monte Carlo (MCMC) data generation .....	58
3.3	Overview of Analytical Approach .....	59
3.3.1	Hierarchical Parameterization.....	59
3.3.2	Block diagram of approach.....	62
3.4	Data Types Obtained and Generated.....	64
3.4.1	Data Generated by ARMA.....	64
3.4.2	Synthetic Data from MCMC.....	65
3.4.3	Experimental Data .....	70
3.5	Comparison and other processing techniques .....	70
3.5.1	Discrete Wavelet Transform & Dimensionality Reduction.....	70
3.5.2	Comparison to Nearest Neighbor Analysis.....	71
3.5.3	Comparison to Raw Probability Calculation .....	72
3.6	Hierarchical Transition Matrix Model (HTMM).....	73
3.6.1	Formulaic Derivation .....	73
3.6.2	Appropriateness of Gaussian Model.....	74
3.6.3	Application of Confidence Intervals & Anomaly Detection .....	77
3.7	Results .....	79
3.7.1	Detection results in ARMA-generated data .....	79
3.7.2	Detection results in MCMC-generated data .....	81
3.7.3	Experimental Bearing Fault Data using HTMM .....	83
3.8	Performance Comparison.....	85
3.8.1	Demonstration of Computational Efficiency .....	85
3.8.2	Results in Experimental Data using Nearest Neighbors.....	86

3.8.3	Results in Synthetic Data using Nearest Neighbors .....	86
3.9	Discussion .....	88
3.9.1	Conclusions.....	88
3.9.2	Future Work .....	89
4	ANOMALY DETECTION IN PARAMETER-DRIVEN DATA.....	90
4.1	Background .....	90
4.2	Proposed method .....	91
4.2.1	Assumptions.....	92
4.2.2	Derivation of Joint Posterior & Sequential Updater.....	93
4.2.3	Derivation of Distance Function between Gamma Posterior Distributions.....	94
4.2.4	Hierarchical Parameterization of Difference Metric.....	95
4.2.5	Anomaly Detection Process.....	97
4.3	Data sets .....	98
4.3.1	Simulated Data.....	98
4.3.2	Experimental Data .....	100
4.4	Results .....	103
4.4.1	Performance Summary, Simulated Data.....	103
4.4.2	ROC Curves, Simulated Data .....	105
4.4.3	Experimental Data .....	106
4.5	Conclusions .....	107
5	ANOMALY DETECTION IN SPATIOTEMPORAL DATA.....	108
5.1	Introduction .....	108
5.2	Approach .....	111

5.3	Preprocessing .....	112
5.4	Poisson Distribution Data Model .....	113
5.5	Poisson Model Analysis and Results .....	114
5.5.1	Poisson Results: Simulation data.....	114
5.5.2	Poisson Results: RWSE Data.....	116
5.6	Discrete Convolution Event Extraction Model .....	116
5.7	Discrete Convolution Model Analysis and Results.....	118
5.7.1	Event-Space Results: Nearest Neighbor Method.....	119
5.7.2	Event-Space Results: Event Likelihood Model .....	122
5.8	Conclusions .....	125
5.8.1	Poisson probability data model.....	125
5.8.2	Spatiotemporal model & discrete convolution analysis.....	125
6	SUMMARY AND CONCLUSIONS .....	127
6.1	State Transition Anomaly Detection .....	127
6.2	Parameter-Driven Anomaly Detection.....	129
6.3	Spatiotemporal Network Anomaly Detection .....	131
6.4	Future Work .....	132
6.5	Final Conclusions.....	133
7	REFERENCES .....	134
	APPENDIX A: FULL DERIVATION OF GAMMA POSTERIOR .....	1477-QQQQQQ
	APPENDIX B: FULL DERIVATION OF DIFFERENCE METRIC $w$ .....	148
	APPENDIX C: ANNOTATED LIST OF FIGURES .....	150

## LIST OF FIGURES

Figure 1. Example of a point anomaly when tracking multiple dimensions .....	4
Figure 2. Example contextual anomaly in an ECG pattern. ....	4
Figure 3. ROC Curve Axis and Quadrant Labeling.....	8
Figure 4. Reduction of time-series data into symbolic bins.....	9
Figure 5. Markov chain with 5 discrete states (a), and corresponding transition matrix (b).....	20
Figure 6. Cumulative Density Function for determining MCMC transition probability.....	22
Figure 7. Transition Matrix for predator-prey scenario with two absorbing states .....	23
Figure 8. Flowchart of Sequential Updating & Parameter Refinement.....	24
Figure 9. Basic diagram of a physical neuron (a) and an analogous artificial neuron (b).....	28
Figure 10. Example Feed-Forward Artificial Neural Network.....	29
Figure 11. Digits 0 through 9 as 5x5 image arrays .....	31
Figure 12. An anomaly letter “A” relative to known pattern “8” .....	31
Figure 13. An example Bayesian hierarchical model for mean parameters with fixed variance .	32
Figure 14. Example transition matrix probability calculation scheme .....	36
Figure 15. D-Markov transition matrix where $D=2$ , $N=2$ .....	37
Figure 16. Nested hierarchy of pixels.....	39
Figure 17. Example ball bearing assembly, cross-section .....	48
Figure 18. ISO Velocity Chart from <a href="http://www.stiweb.com/TechNote117.html">http://www.stiweb.com/TechNote117.html</a> .....	50
Figure 19. Representation of symbolic bin assignment in SAX.....	55
Figure 20. Markov Chain with transition probabilities for a three-state system .....	57
Figure 21. Example Transition Matrix for a 3-state System.....	58
Figure 22. Hierarchical Parameterization Structure for Anomaly Detection .....	60

Figure 23 (a)-(h). Several examples of sparse Transition Matrices from normal data .....	61
Figure 24. Training Algorithm Flowchart .....	62
Figure 25. Testing Algorithm Flowchart .....	63
Figure 26. Time-series data generated by ARMA, normal and anomalous models .....	65
Figure 27. Power spectral density of normal (a) vs. anomalous (b) data.....	65
Figure 28. Normal and anomalous transition matrices, sparse diagonal case. ....	66
Figure 29. Normal and anomalous transition matrices, purely random case.....	66
Figure 30. Bistochastic Matrix of Normal Data (a) and Anomalous Data (b).....	68
Figure 31. Histogram (a) and Frequency Analysis (b) of Normal Bistochastic Data.....	69
Figure 32. Histogram (a) and Frequency Analysis (b) of Anomalous Bistochastic Data.....	69
Figure 33. Cumulative Distribution Functions for TM Z-scores vs. Gaussian Model .....	75
Figure 34. Histogram of Normal Data Transition Matrix Z-scores .....	78
Figure 35. Example Z-Score distribution of example HTMM analysis .....	79
Figure 36. ROC Curves for HTMM applied to ARMA-generated data .....	80
Figure 37. ROC Curves tested against 500 anomalous spans for various anomaly lengths. ....	82
Figure 38. ROC curves for diagonal (a), random (b) TMs .....	83
Figure 39. Z-Scores for DB1 / SAX window 1 experimental normal (a) and anomalous (b) data .....	84
Figure 40. Z-Scores for DB1 / SAX window 10 experimental normal (a) and anomalous (b) data .....	84
Figure 41. Nearest Neighbor detection for experimental normal (a) and anomalous (b) data .....	86
Figure 42. Nearest Neighbor detection for synthetic normal (a) and anomalous (b) data.....	87
Figure 43. Example production of parts $y_i$ with mean $\mu$ and precision $\phi$ .....	92

Figure 44. Flowchart for obtaining mean normal posterior CDF .....	96
Figure 45. (a) 100 distinct Gamma CDFs, (b) Mean CDF vs. CDF of means. ....	97
Figure 46. Flowchart for parameterizing difference metric $w$ . ....	97
Figure 47. Flowchart for Anomaly Detection Process.....	98
Figure 48. Simulated Precision Manufacturing Data with Anomalies, $L=20$ .....	100
Figure 49. Comparison of Experimental Measurement Parameters .....	102
Figure 50. Precision Tracking in Simulated Precision Process Data, $n=10$ .....	103
Figure 51. MOSUM Residuals in Simulated Precision Process Data, $n=10$ .....	104
Figure 52. ROC Curves for Simulated Data, Various Anomaly Lengths.....	106
Figure 53. Real World Sensor Environment (RWSE) Sensor Layout.....	109
Figure 54. Simulated Sensor Environment Layout.....	110
Figure 55. (a) Single Sensor and (b) Multi-Sensor Movement Increase Detected .....	115
Figure 56. Numerical Score ROC Curves.....	115
Figure 57. Sensor 5 vs. All Others: Correlation Result Matrix .....	118
Figure 58. "Arrival" Event, Nov. 8-28, Weekdays Only (Day 14 = Thanksgiving Day) (a) Arrival Event Convolution Results (Sensor #5 leads #16 by 12 to 18 s) (b) Total Distance from Nearest Neighbor in Training Set (Anomalousness Metric).....	120
Figure 59. Double-Trigger Event, Nov 8-28, Weekdays Only (a) Double-Trigger Event Convolution Results (Sensor #14 leads #14 by 6 to 12 s) (b) Total Distance from Nearest Neighbor in Training Set (Anomalousness Metric).....	121
Figure 60. Convolution Analysis Results – Nearest Neighbor, Simulated Data .....	122
Figure 61. Example distribution of expected “arrival event” activity .....	123
Figure 62. Short-term Anomalousness Metric for Sensor 1 vs. 2.....	124

## LIST OF TABLES

Table 1. Pseudocode for Symbolic Aggregate Approximation algorithm.....	10
Table 2. Pseudocode for PCA Classification Algorithm .....	15
Table 3. Pseudocode for K-Means Algorithm.....	17
Table 4. Pseudocode for Markov Chain Monte Carlo data generation algorithm.....	58
Table 5. Parameters for ARMA test model .....	64
Table 6. Performance Characteristics for HTMM applied to ARMA-generated data.....	80
Table 7. Synthetic ground-truth bistochastic data, using Nearest Neighbors.....	87
Table 8. Synthetic ground-truth bistochastic data, using HTMM .....	87
Table 9. Experimental bearing fault data, using Nearest Neighbors .....	88
Table 10. Experimental bearing fault data, using HTMM.....	88
Table 11. List of Empirically Measured Parameters .....	101
Table 12. Performance Characteristics Comparison.....	104
Table 13. RWSE Data Results.....	116

## LIST OF ABBREVIATIONS

All abbreviations in this work are defined the first time they are used in text. Below is a table of some commonly-used abbreviations for general reference:

Abbr.	Meaning
SAX	Symbolic Aggregate approximation
ANN	Artificial Neural Network
AR(I)MA	Autoregressive (Integrated) Moving Average
BPUSH	Bayesian Posteriors Updated Sequentially and Hierarchically
CUSUM	Cumulative Sum control chart
DFT	Discrete Fourier Transform
DWT	Discrete Wavelet Transform
EM	Expectation Maximization
FFT	(Fast) Fourier Transform
FPR	False Positive Rate
GEOINT	GEOspatial INTelligence
GMM	Gaussian Mixture Model
HTMM	Hierarchical Transition Matrix Model
HTMM	Hierarchical Transition Matrix Model
MCMC	Markov Chain Monte Carlo
ND or RS	Non-Discriminant or Random Selector
NN	Nearest Neighbor
PCA	Principal Component Analysis
PIR	Passive Infra-Red
ROC	Receiver Operating Characteristic
SAR	Synthetic Aperture Radar
SNR	Signal-to-Noise Ratio
SVD	Singular Value Decomposition
TM	Transition Matrix
TPR	True Positive Rate

## LIST OF SYMBOLS

All symbols in this work are defined the first time they are used in text. Below is a table of some commonly-used abbreviations for general reference:

Symbol	Meaning
$\alpha$	Rate parameter of the Gamma distribution
$\beta$	Shape parameter of the Gamma distribution, OR Breakpoint in Symbolic Aggregate Approximation
$\gamma$	Cutoff value (in y-axis) for Symbolic Aggregate Approximation
$\gamma(A, B\theta)$	Lower-incomplete Gamma function
$\varepsilon$	Random Gaussian noise for ARMA model, or Machine error (smallest floating point precision)
$\eta$	Learning coefficient for Artificial Neural Network
$\theta$	Angle of rotation, or Arbitrary parameter in a probabilistic system
$\lambda$	Eigenvalue
$\mu$	Mean
$\rho$	Density
$\sigma$	Standard Deviation of a Gaussian distribution
$\tau$	Time-delay
$\phi$	Precision parameter of a Gaussian distribution
$\Phi$	Coefficient matrix for AR portion of ARMA model
$\Psi$	Coefficient matrix for MA portion of ARMA model
$\omega$	Rotational frequency
$\Sigma$	Summation of a discrete set
$\Pi$	Multiplication of a discrete set
$\Gamma(A)$	Gamma function for constant $A$
$\Gamma(A, B\theta)$	Upper-incomplete Gamma Function
$\Delta$	Discrete difference
$A^T$	Transpose of matrix $A$

# 1 INTRODUCTION

This chapter will act as an introduction to the problem of anomaly detection in general, and a mathematical introduction to many of the techniques found later in this dissertation. A formal literature review is continued and expanded in Chapter 2, including in-depth discussions of previous approaches and how the methods developed in this dissertation differ from prior work. Some of the techniques discussed in this chapter are developed further as they are implemented in the text. This section serves as a conceptual introduction and lays the framework to help the reader understand these approaches in the context of this dissertation.

## 1.1 Anomaly Detection

This work focuses on the problem of anomaly detection as it pertains to mechanical processes and systems. The broader definitions of anomaly detection will be discussed, and applications provided for fields other than Mechanical Engineering. System characteristics, previous approaches, and methods for evaluation will also be explored. The goal of this section is to provide the reader with a broad understanding of the importance and applications of anomaly detection, how detection is performed and detection quality is evaluated, and why anomaly detection is highly relevant to mechanical systems.

### 1.1.1 Problem Definition

Anomalies have been defined differently across many fields, and as such there is no single absolute definition for what constitutes an “anomaly”. Because of the wide variety of metrics for assessing anomalousness, their application has become largely situation-specific and geared towards the parameters of individual systems and environments. The range of anomaly detection methods spans from proximity-based techniques (e.g., the Mahalanobis or Euclidian distance from a point to a central value), fuzzy or probabilistic techniques (e.g., a parameter

value assessed using Z-score), and other quantitative difference metrics. This work seeks to develop several novel and effective methods for anomaly detection in heterogeneous environments, which should extend to other applications that contain similar data types.

An anomaly can be described as any event that deviates from what is considered normal behavior. In the field of statistics, an anomaly is loosely defined as a significant deviation from what is thought to be normal data, based on previously obtained or expected results (Khatkhate, Ray, Keller, Gupta, & Chin, 2006) (Chandola, Banerjee, & Kumar, 2009). The detection of anomalies is a widely applicable practice across many fields, as the presence of anomalies in data often indicates some abnormal behavior in a system. This has made anomaly detection a topic of significant recent interest in the fields of data mining, knowledge discovery, and pattern recognition.

### **1.1.2 Applications**

The importance of accurate, rapid, reliable anomaly detection cannot be understated. In the biomedical sciences, the detection of anomalous ECG signals or brainwave functions may act as precursors, serving as “early warnings” before heart attacks, strokes, or other catastrophic conditions. For example, pattern recognition in Electrocardiographic data is still largely dependent on operator inference (instead of automated anomaly detection processes) as these processes have not yet reached the reliability of definitive diagnoses:

*“A fully automatic reading device with which there is no need for the operator to reinterpret the tracings looks promising at first glance because the automatic file contains many numbers, figures, and statistics, but all this accumulated data are not always correct and often do not reflect the truth. As the device is completely automatic, its interpretation value and efficiency diminish with the complexity of*

*the recording. In our opinion, fully automatic recordings without a human superimposed access should be totally avoided.” (Adamec & Adamec, 2008)*

Operator dependency is a consistent issue across many fields. High dimensionality data presents a number of unique problems to the analysis of systems for patterns and anomalies. The defense industry is particularly focused on improving operator accuracy by reducing the amount of non-relevant information which reaches the end-user. In systems which rely on the human monitoring of video processes (e.g., a camera surveillance network), an analyst may be tasked with the simultaneous observation of more than ten video sources for extended periods of time. This type of system leverages the visual processing ability of a human against the tendency of humans to become bored fatigued or desensitized, or to encounter visual issues such as target confusion or insufficient resolution. Modern high-dimensionality monitoring systems seek to reduce the negative impacts of human involvement, in part by aiding the operator in the automatic detection of “relevant” data, streamlining the user interface, and reducing the amount of data throughput being observed by the analyst. (Cheyer & Julia, 1998)

### **1.1.3 Presentation**

Anomalies can present themselves in a variety of different forms. Point anomalies, contextual anomalies, and collective anomalies are three of these classifications (Chandola, Banerjee, & Kumar, 2009). Point anomalies are single points that lie outside the expected domain of the data; they are essentially the outliers of a dataset (Das & Parthasarathy, 2009). Point anomalies generally do not occur in time-series data unless there is a data collection error or some alteration to the system.

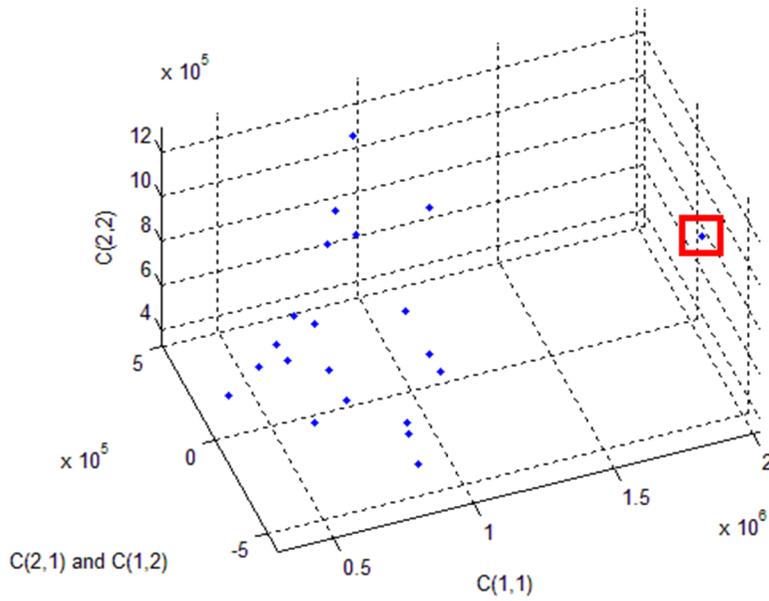


Figure 1. Example of a point anomaly when tracking multiple dimensions

Contextual anomalies are isolated events that do not follow the general trend of the dataset. Contextual anomalies may not seem anomalous on their own, but are considered anomalous given the presence of surrounding data (Chandola, Banerjee, & Kumar, 2009). An example of this in time-series data measuring a mechanical phenomenon, e.g., vibration, would be an unexpected peak or trough in otherwise trending data that occurs over a short duration of time before returning to the original trend. If a portion of a dataset is significantly different from the rest of the dataset, it is considered a collective anomaly. This portion of the dataset may not contain anomalies within itself but it may be anomalous with respect to the rest of the set (Chandola, Banerjee, & Kumar, 2009) (Chuah & Fu, 2007).

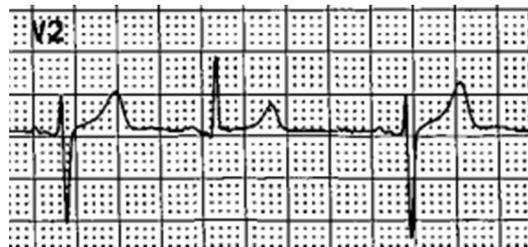


Figure 2. Example contextual anomaly in an ECG pattern.

An entire dataset may also be classified as a collective anomaly if it is significantly different what is considered a normal dataset. This work mainly addresses contextual anomalies as they pertain to their surrounding data, including sources. It should be noted that an “anomaly” is not necessarily an “outlier” – the latter is defined as a point (or set of points) that is measurably far from a mean value; the former describes a point or set of points that is measurably different from previous observations. For example, a system with a mean of 0 and standard deviation of 1 would not classify 0.1 as an outlier. However, if 100 sequential measurements of 0.1 occurred without any change, a human observer might consider this to be anomalous.

#### **1.1.4 Approaches**

Determining if data is anomalous depends first on establishing ground truth. Ground truth pertains to the known quality of training data, both in the presence and presentation of “normal” system behavior. Other datasets can be seen as either anomalous or normal based on their comparisons to ground truth (Chandola, Banerjee, & Kumar, 2009). If there are significant differences between some candidate data and normal training data, the test data is considered anomalous, while if the test data is similar to ground truth it is considered normal. The threshold of what is considered anomalous becomes an issue due to the possibility of anomalous patterns being hidden beneath noise in the system being tested (Rajagopalan & Ray, 2006) (Eftekharijad, Carrasco, Charnley, & Mba, 2011). This is often the case in practical mechanical systems and can result in problems with the accuracy of anomaly detection metrics when applied to this experimental data.

The process of anomaly detection often relies on supervised training: the selection of known-normal and known-anomalous segments of data. This is an appropriate approach if the

characteristics of the anomalous pattern are known in advance – the classification algorithm is trained to identify both normal and anomalous data independently. Even if the presence of anomalies is a rare occurrence, this approach performs very well relative to unsupervised methods, because it utilizes a twofold criterion for categorizing data. Unsupervised learning is a more practical approach, because this type of prior knowledge is unavailable in many cases. The goal of this approach is to automatically classify data as normal or anomalous based only on an unlabeled ground truth set of “normal” data. While more difficult, this method is most useful in cases where it is desirable to detect *all* anomalous data, instead of anomalies of a specific *type*.

*... an anomaly detection system has the ability to detect previously unknown attacks. This is due to the fact that a profile of intrusive activity is not based on specific signatures representing known intrusive activity. An intrusive activity generates an alarm because it deviates from normal activity, not because someone configured the system to look for a specific attack signature. (Patcha & Park, 2007)*

The difficulty in this approach is that anomalies must be significantly different from the normal data, demonstrating distinct signals, parameters, patterns, or trends that can be used to classify the anomalous segments. (Tan, Kumar, & Steinback, 2005)

### **1.1.5 Evaluation**

The effectiveness of an anomaly detector can be measured based on the rate of true positive and false positive detection. True positives occur when data is correctly flagged as anomalous. A false positive occurs when normal data is incorrectly flagged as anomalous. The effectiveness of an anomaly detector is often displayed using a Receiver Operating Characteristic (ROC) curve. The ROC curve is a function of the rates of true positive (TPR) and false positive

(FPR) detection (Trevor & Tibshirani, 2009). The horizontal axis of a ROC curve is typically the FPR while the vertical axis is typically the TPR.

$$TPR = \frac{\textit{True Positives}}{\textit{True Positives} + \textit{False Negatives}} \quad (1)$$

The TPR, sometimes called the Recall of a classifier, is a fraction of how many anomalies were correctly identified by the model. This fraction must range between 0 and 1, shown above in Equation (1). The FPR, on the other hand, is the fraction of how many normal instances were incorrectly identified as being anomalous. This fraction must also range between 0 and 1, shown in Equation (2).

$$FPR = \frac{\textit{False Positives}}{\textit{False Positives} + \textit{True Negatives}} \quad (2)$$

Another metric for assessing performance is the precision,  $p$  of a system. This precision is similar to the TPR, but instead acts as a relationship between true positive and false positive, shown in Equation (3) (Hand, 2009).

$$p = \frac{\textit{True Positives}}{\textit{True Positives} + \textit{False Positives}} \quad (3)$$

Anomaly detectors are compared based on where they appear on a ROC curve, with areas above the Non-Discriminant (ND) line (sometimes referred to as the Random Selector threshold) being classified as “positive” detection, and areas below the ND line being classified as “negative” detection. The more area between a metric’s ROC curve and the Random Selector line, the better that metric’s performance (Hanley & McNeil, 1982). This allows the objective comparison between the performance characteristics of multiple classification models at the same time, relative to a “break even” value.

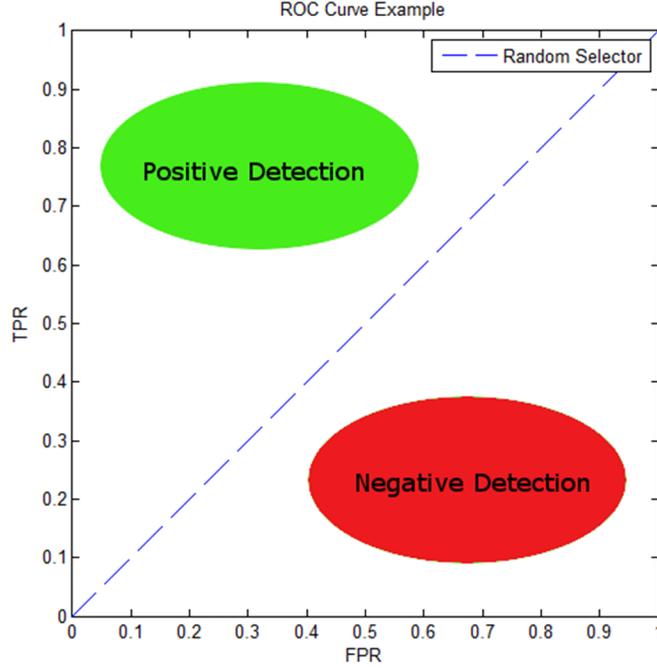


Figure 3. ROC Curve Axis and Quadrant Labeling

### 1.1.6 Stochastic Processes

A stochastic process is often defined as a system whose behavior evolves over time such that, given some initial starting condition, the system condition at some arbitrary time  $t$  cannot be definitely known (Karlin, 1998). Stochastic processes are found in a wide range of applications such as computer science, linguistics modeling, business & finance, and process analysis. Some specific examples of time-series data include seismographic data obtained following earthquake or nuclear events, and electrocardiogram data obtained from heart attacks or other cardiomyopathy events (Billinger, 2001). In stochastic processes, future data is often partially dependent on prior data (except in the case of a purely random walk). In a discrete stochastic process, in which data are indexed by a discrete time variable  $k$ , a data point  $y_k$  typically follows:

$$y_k = f(y_{k-1}, y_{k-2}, \dots, y_{k-p}) + \Omega(\theta_1, \theta_2, \dots, \theta_q) \quad (4)$$

where  $f$  is a function that encompasses some finite number  $p$  of past data points  $y_{k-p}$ , and  $\Omega$  is a probability space with parameters  $\theta_q$  that apply to  $q$  previous observations. These types of

processes require analysis using techniques combining statistical inference and often incorporate hierarchical modeling.

## 1.2 General Techniques

This section describes several of the key techniques applied throughout this work. These techniques may be defined in greater detail as they are used. This section serves as a general overview of the mathematical, statistical, signal processing, and parameterization approaches which contribute to the overall goal of anomaly detection as it pertains to mechanical systems.

### 1.2.1 Symbolic Aggregate Approximation

The process of Symbolic Aggregate Approximation (or SAX) is described in greater detail in sections 3.2.2 and 5.3 of this work; this section provides a broad overview of the algorithm. SAX is a dimensionality reduction process which partitions time-series data into equiprobable, symbolically-labeled bins (Keogh, Lin, & Fu, 2005). The approximation of time-series data in this way effectively reduces the amount of data required to represent the data, while preserving shape trends. An example of this process is shown below in Figure 4.

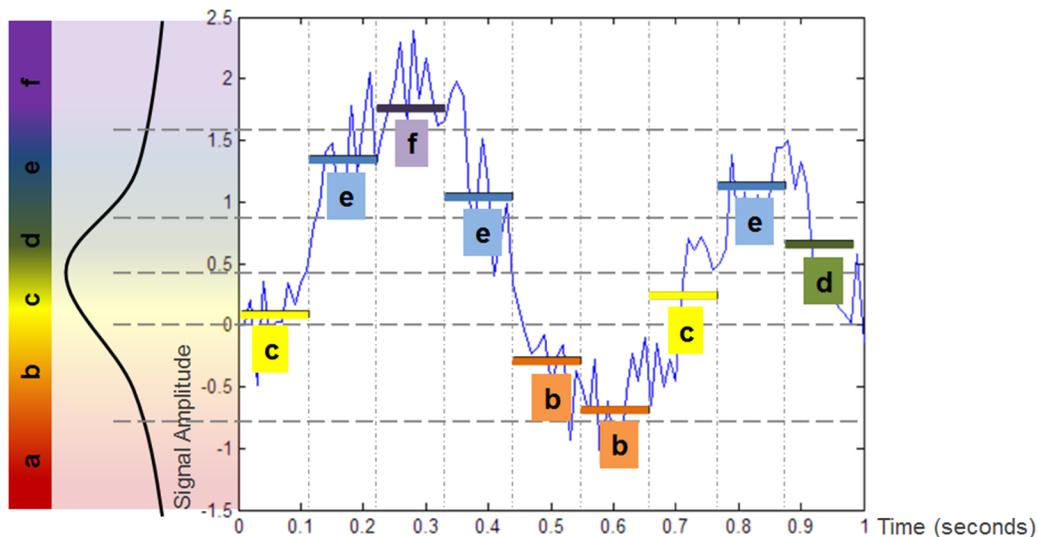


Figure 4. Reduction of time-series data into symbolic bins.

This reduction is computationally inexpensive, and runs quickly on large datasets as there is no inter-data dependency (e.g., no need to lookup nearest neighbors). Pseudocode for the SAX algorithm is shown below in Table 1:

Table 1. Pseudocode for Symbolic Aggregate Approximation algorithm.

```

(User specifies alphabet length  $N$  and window length  $L$ )
Calculate first-order statistics of the data (mean  $\mu$  and standard deviation  $\sigma$ )
Use lookup tables or integrals to calculate breakpoint Z-scores  $\beta_1, \beta_2, \dots, \beta_{N-1}$ 
for all data, indexed by  $k$  {
    Separate data into equal window lengths  $L$ 
    Average each of these windows, store as  $W_k$ 
    Calculate Z-score of window  $W_k$ 
    if Z-score  $\leq \beta_1$  {
        assign window  $W_k$  symbolic label "a"
    }
    else if Z-score  $> \beta_{N-1}$  {
        assign window  $W_k$  symbolic label "N"
    }
    else for all breakpoints  $\beta_1, \beta_2, \dots, \beta_{N-1}$  {
        if Z-score  $> \beta_n$  AND Z-score  $\leq \beta_{n+1}$  {
            assign window  $W_k$  symbolic label "n"
            break
        }
    }
}

```

In order for SAX to work correctly, the data should be:

- Stationary
- Ergodic
- Approximately normally distributed

SAX makes the assumption that the data are not lower-bounded (as indicated by the assumption of Gaussian distribution), but lower-bounded data does not invalidate SAX as an approach

(Patel, Keogh, Lin, & Lonardi, 2002). This makes SAX useful for a wider variety of systems, and is applied throughout this dissertation to multiple data types.

## 1.2.2 Principal Component Analysis & Eigenvalue Decomposition

Principal Component Analysis (PCA) is a common data processing technique utilized in many modern applications of machine learning and anomaly detection, such as the “Eigenfaces” algorithm (Zhang & Turk, 2008). In PCA, a set of data are reduced into the principal vectors that best describe the variability in that data. If all of the vectors are preserved, the data are fully recoverable; if only the principal vectors are preserved, the data are best represented by these vectors in reduced dimensionality. The process of PCA is typically performed through an eigenvalue / eigenvector decomposition of a dataset’s covariance (or correlation) matrix. This makes PCA an equivalent process to the Karhunen–Loève theorem, which states that a dataset  $X$  may be represented by a linear summation of orthogonal functions, as in Equation (5).

$$X = \sum_{n=1}^{\infty} a_n f_n \quad (5)$$

where  $a_n$  are weighting coefficients, and  $f_i$  and  $f_j$  are orthogonal functions (or vectors) such that

$$f_i^T f_j = f_j^T f_i = 0 \quad (6)$$

Similar to a Fourier series, this eigenvalue decomposition reveals both the orthogonal functions and their weighting coefficients, which allow the data to be reconstructed. For eigenvalue decomposition this series is not performed to infinity, but is instead bounded by  $R$ , the number of dimensions in the data. In order to obtain a truly orthonormal set of eigenvectors, the data should be distributed as a multivariate Gaussian model, as in Equation (7).

$$X \sim N(\mu, \sigma) \quad (7)$$

This makes PCA an excellent complement to SAX, which inherently processes the data into discrete equiprobable bins based on a Gaussian distribution. The two approaches differ in the way that dimensionality reduction is performed: SAX reduces the number of discrete values needed to adequately represent the amplitude of the data, while PCA reduces the number of discrete vectors needed to adequately represent the data.

In order to perform PCA on one-dimensional time-series signals, the data should first be spaced into windows of equal length. For image data, these windows are typically the length of the unwrapped pixels of the image. For one-dimensional signal data, such as vibration data from bearing drives, the windows are specified by the user. First, a training set  $X$  is separated into  $M$  windows ( $W_m$ ) of equal length  $R$ . Then, these windows are averaged using Equation (8) to obtain the mean window.

$$\overline{W} = \frac{1}{M} \sum_{m=1}^M W_m \quad (8)$$

The mean window  $\overline{W}$  is subtracted from each window  $W_m$  which yields difference windows  $\hat{W}_m$ . These difference windows are concatenated (without respect to order as the next process is time-invariant) orthogonally to the window length, to form the mean-averaged matrix  $A$  which is of size  $R \times M$ .

$$\hat{W}_m \Rightarrow A = \begin{bmatrix} \hat{W}_{1,1} & \hat{W}_{1,2} & \cdots & \hat{W}_{1,R} \\ \hat{W}_{2,1} & \hat{W}_{2,2} & \cdots & \hat{W}_{2,R} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{W}_{M,1} & \hat{W}_{M,2} & \cdots & \hat{W}_{M,R} \end{bmatrix} \quad (9)$$

Normally, the covariance matrix  $C$  is calculated by the simple operation on the matrix  $A$  shown in Equation (10). This is equivalent to obtaining the outer product matrix of  $A$  against

itself. After obtaining the covariance matrix, the eigenvalues and eigenvectors of  $C$  are fairly straightforward to obtain.

$$COV \langle X_{windowed} \rangle = C = AA^T \quad (10)$$

However, the covariance matrix obtained from  $AA^T$  is of size  $R \times R$ , containing up to  $R^2$  elements. Depending on the size of the dataset, and the length of the window chosen, this operation may take a significant computational toll. If the QR algorithm is used (a common component to computing the eigenvalue problem) (Trefethen & Bau, 1997), this approach may have an operations cost of Equation (11):

$$\#ops = \left[ \frac{10}{3}R^3 + O(R^2) \right] + [6R^2 + O(R)] \quad (11)$$

where the first bracketed term represents the cost of transforming  $C$  into upper Hessenberg form, and the second bracketed term is the cost of a QR decomposition of an upper Hessenberg matrix (Demmel, 1997) (Trefethen & Bau, 1997). As the size of  $R$  is user-specified, and may need to be quite large in order to encompass sufficient data for pattern extraction, it is at this point that the calculation for PCA takes one of two paths.

Generally, if  $(M \gg R)$  – that is, if the number of windows  $M$  (training segments) is much larger than the window size  $R$  (number of dimensions in the data) – it is more efficient to continue with the traditional approach of calculating the eigenvalues and eigenvectors of  $C$ . Otherwise, if  $R \gg M$ , a more computationally-efficient approach is shown below. Because PCA by its nature focuses on the principal eigenvectors of the covariance matrix, it is convenient to isolate the eigenvalue problem to a limited subset of the entire eigenspace.

By isolating analysis to the  $M^{th}$  eigenvectors of  $C$ , it is possible to reduce the resulting solution to an  $M$ -dimensional approximation of the eigenvalue problem. Instead of calculating the outer product matrix of  $A$  ( $AA^T$ ), it is computationally simpler to calculate the inner product

matrix  $L$  of  $A$  such that  $L = A^T A$ . In this way, the eigenvalues and eigenvectors of  $L$  are instead calculated from the standard eigenvalue problem shown in Equation (12). The eigenvalues  $\lambda_m$  are consistent with the  $M$  eigenvalues of the original covariance matrix,  $C$ .

$$L v_m = \lambda_m v_m \quad (12)$$

Using these  $v$  and  $\lambda$ , it is then possible to recover the true eigenvectors through a simple transform using  $A$ , in Equation (13):

$$\bar{u}_m = A \bar{v}_m \quad (13)$$

where  $u_m$  and  $\lambda_m$  are eigenvectors and eigenvalues of  $C$ . By scaling the resulting eigenvectors such that their inner product matrix  $u_m^T u_m$  equals unit length, a semi-complete *orthonormal* set of eigenvectors  $U = [\bar{u}_1, \bar{u}_2, \dots, \bar{u}_M]$  and eigenvalues  $\Lambda = [\lambda_1, \lambda_2, \dots, \lambda_M]$  are obtained. These are only the  $M$  principal eigenvectors of  $C$ , which significantly reduces the computation time necessary to obtain and sort the set while still maintaining a good representation of the system. This approach can be further extended by isolating the  $J$  principal eigenvectors (those with the largest associated eigenvalues) of  $U$  such that  $J < M$ . This yields the  $J$  principal components of the system, which can then be used for pattern recognition algorithms and dimensionality reduction. This sorted and truncated matrix  $\tilde{U}$  is structured as Equation (14).

$$\tilde{U} = \begin{bmatrix} u_{1,1} & u_{2,1} & \cdots & u_{J,1} \\ u_{1,2} & u_{2,2} & \cdots & u_{J,2} \\ \vdots & \vdots & \ddots & \vdots \\ u_{1,R} & u_{2,R} & \cdots & u_{J,R} \end{bmatrix} \quad (14)$$

To transform new data into these principal coordinates, it is a straightforward procedure to subtract the mean window  $\overline{W}$  from the new candidate window  $W_{test}$ , and then premultiply by  $\tilde{U}^T$  to obtain a weighting coefficient matrix,  $P$  as shown in Equation (15).

$$P = \tilde{U}^T (W_{test} - \overline{W}) = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_J \end{bmatrix} \quad (15)$$

The resulting coefficient values of  $p_1, p_2$ , etc. are the contributions of each principal eigenvector to the candidate window. In this way, it is possible to perform *detection* and *classification* of training data by comparing these difference vectors  $P$ .

### 1.2.3 Pattern Recognition using PCA and SVD

In classification, a candidate set is compared against the training set, and matched to the training vector through a distance classifier. In this process, the difference vector  $P_i$  for each training window is found and stored. The same approach is applied to find  $P_{test}$  for a window of test data. Then, the test windows are compared to each training window and a minimum distance is found. This process is detailed as pseudo-code in Table 2.

Table 2. Pseudocode for PCA Classification Algorithm

$P_{test} = \tilde{U}^T (W_{test} - \overline{W})$ <p>for each training window <math>W_i</math> {</p> $P_i = \tilde{U}^T (W_i - \overline{W})$ $\varepsilon(i) = \ P_{test} - P_i\ $ <p>}</p> $\text{bestMatch} = \text{INDEX\_MIN}\langle \varepsilon \rangle$
---

In this classification example,  $\varepsilon$  is an array of best matches for the difference weighting coefficients between a candidate window and the training set. The index of the minimum of

these matches, *bestMatch*, is the training window that provides the overall best match to the test window.

In detection, it is useful to discern whether or not a candidate set has been generated from the same process that generated the training data. This is particularly useful in anomaly detection and pattern recognition in machine learning. To accomplish this, the test data must be compared against itself when projected onto the eigenspace of the training data. This projection occurs through Equation (16) below, where  $P_{test}$  is separated into its element components  $p_{test,j}$  and projected onto the eigenvectors  $u_j$ .

$$P_{projected} = \sum_{j=1}^J p_{test,j} u_j \quad (16)$$

A difference metric  $\varepsilon$  is then obtained by comparing the original vector  $P_{test}$  against the projection  $P_{projected}$ , through Equation (28).

$$\varepsilon = \|P_{test} - P_{projected}\| \quad (17)$$

These difference metrics can then be parameterized, or thresholded, and incorporated into anomaly detection programs and machine learning applications.

#### 1.2.4 Clustering Algorithms

In the field of data mining, the “clustering” of data refers to the process of assigning individual non-labeled data points into logical groups of points. Like PCA, clustering’s advantages include the description of data using only a few key parameters, the classification of data into logical groups, and the ability to represent the data with significantly reduced information (Bo, Zhang, & Yu, 2010). However, because PCA (and similar approaches) often require a high degree of computational complexity, clustering algorithms are often more suited to large-scale datasets. One shortfall of clustering is its high degree of subjectivity: what one

human observer may define as a “cluster” may be found by a different observer to be multiple clusters. The definition of what constitutes a “cluster” is non-rigorous, leading to a wide variety of algorithms aimed at logical grouping (Estivill-Castro, 2002). Clustering is particularly useful as it applies to the finding of Nearest Neighbors (NN’s). Finding the NN of a candidate data sample requires a comparison against all other data in the set. By applying clustering algorithms as a preprocessing technique, this processing requirement can be significantly reduced by only comparing candidate data to other points within the same cluster.

The K-Means algorithm is a straightforward method to determining the location and point-assignments of the cluster centroids of a dataset. Given a user-specified value for  $K$ , the K-Means algorithm iterates by assigning each data point to its nearest centroid  $k$ . When all points have been assigned, each centroid  $k$  is recalculated to be the centroid of the points assigned to it in the previous step. In this way, the K-Means algorithm is a form of Expectation Maximization (EM), in that data are assigned to the centroids during the Expectation step, and the cluster centroids are optimized during the Maximization step (MacQueen, 1967). A pseudocode representation of the K-Means algorithm is shown below in Table 3:

Table 3. Pseudocode for K-Means Algorithm

```

User inputs input data  $X$ , number of means  $K$ 
Set initial centroid locations  $MU_k$ 
Initialize loop counter  $l$ , and max number of loops  $L_{max}$ ;
Initialize centroid motion  $e$  and tolerance  $tol$ 
while  $e < tol$  AND  $l < maxL$  {
  (Expectation step)
  for  $i=1, i \leq \text{length}(X), i++$  {
     $dMUs = X(i) - MUs$ ;
     $[V, I] = \min(dMUs)$ ;
     $Xk(i) = I$ ;
  }
  // Maximization step

```

```
for k=1, k<K, k++ {
    Xsubset = X(find(Xk==k));
    newMUs(k) = mean(Xsubset);
}
e = sum(newMUs-MUs)/sum(MUs);
n++;
MUs = newMUs;
}
```

The K-Means algorithm is best suited to data which are not sparse, which are regularly shaped, and which contain well-separated clusters (Tan, Kumar, & Steinback, 2005). The algorithm is susceptible to noise, outliers, and may not always converge to a globally-correct clustering solution (i.e., the end result may be a local minima, but not a global minima). The initial centroid locations are also subject to error, as they may result in empty clusters (centroids to which no points are assigned) during the K-Means iteration. There are many approaches to address these issues not discussed here; for the purposes of this work, the K-means algorithm is used primarily as a tool for assessing the number, locations, and approximate parameters of data as a preprocessing tool.

**1.2.5 Bayesian Statistics**

Bayesian statistics is a field of statistics that focuses on conditional probabilities, and differs from “classical” (or “frequentist”) statistics in several key ways:

- Bayesian statistics incorporates prior knowledge about system parameters
- Bayesian Statistics supports updating a parameter’s model as more information becomes available over time
- Traditional statistics typically reports confidence measures, whereas Bayesian statistics reports credible intervals with measured uncertainty

In classical statistics, a common confidence measure is the p-value: the probability of some data or event given a null hypothesis. In Bayesian Statistics, relationships between prior and posterior distributions are used to estimate the parameters that drive a system (Everitt, 2002). Bayes' law states that the probabilities of two conditionally-linked instances are related through the following equation:

$$P(B | A) = \frac{P(A | B)P(B)}{P(A)} \quad (18)$$

In classical statistics, the reported probability of a hypothesis test that some parameter  $\theta$  is exactly equal to  $\theta_0$ , based on a measurement  $x_0$ , is given by a confidence interval. This confidence interval is highly dependent on sampling procedure, and how the data are obtained. The p-value for an observation  $x$  to be larger than  $x_0$  given the null hypothesis regarding  $\theta$  is given by:

$$p = P(x \geq x_0 | \theta = \theta_0) \quad (19)$$

Conversely, in Bayesian statistics, the reported probability of a system parameter  $\theta$  existing within an interval  $I$  given some data  $x$  is shown in Equation (20). This is referred to as the credible interval:

$$P(\theta \in I | x) \quad (20)$$

Frequentist statistics tends to report the probability that a hypothesis is supported (or not supported), while Bayesian statistics tends to report the probability that a parameter lies within some range of known values given the prior observation of data (Huang & Darwiche, 1996)

### 1.2.6 Markov Chains

A Markov Chain is defined as a mathematical process that governs transitions between discrete system states. In Markov Chains, the future state  $S^{[k+1]}$  depends on the current state  $S^{[k]}$ . Assuming that each of the transition probabilities between states is discrete and stationary – that

the transition between system states is fixed with respect to time – a Markov Chain can be visually represented as in Figure 5. In Figure 5, a 5-state system is shown for states  $\{ S_1, S_2 \dots S_5 \}$  with Markov probabilities relating all states.

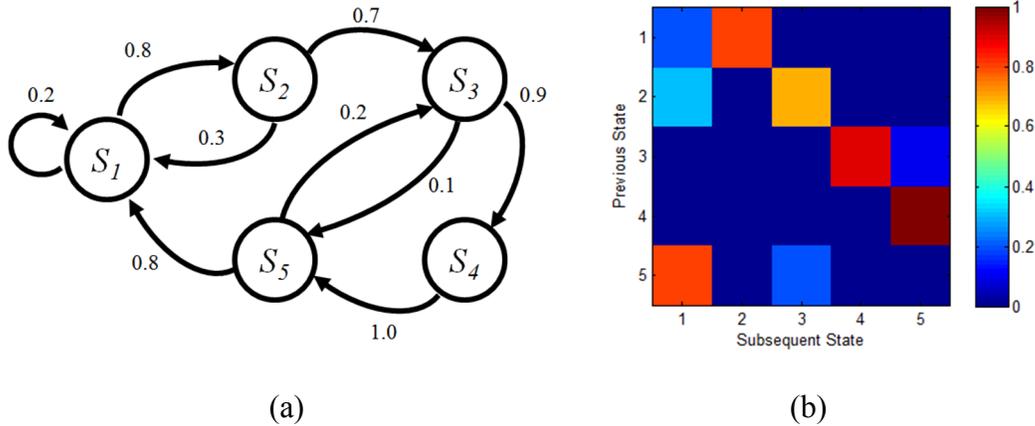


Figure 5. Markov chain with 5 discrete states (a), and corresponding transition matrix (b). Also shown in Figure 5 (b) is a transition matrix (TM). Transition matrices are a common tool for modeling and analyzing stochastic data, and represent the transition probabilities of a Markov chain in matrix format.

Markov chain and transition matrix analysis is often applied to temporal stochastic data for the purpose of pattern recognition and anomaly detection (Patcha & Park, 2007). Obtaining transition matrices from stochastic data is a two-step process: all conditional probabilities are measured, and then all rows are normalized to proper distributions (Winston, 1994). Let  $T$  denote a zero-valued  $N$  by  $N$  matrix, where  $N$  is the number of discrete states that the system may occupy. Assume that the Markov process is stationary, ergodic, and first-order; that is, the probability of arriving at state  $n$  at time  $k$  is purely determined by the system's state  $m$  at time  $k-1$ . Each element of this transition matrix  $T_{mn}$  indicates the probability of the system transitioning to state  $n$ , given that the system was previously at state  $m$ . For a one-dimensional array  $X$ ,

$C(X_{\{S\}})$  represents the count of instances of the sequence of states  $\{S\}$ . Each element of a non-normalized transition matrix  $T'$  is calculated using Equation (21):

$$T'_{mn} = \frac{C(X_{mn})}{C(X_m)} \quad (21)$$

The initial probability distribution  $Q$  shown in Equation (22) is a row vector containing the probabilities  $Q_m$  that the system is in state  $m$  at any given time  $t$ .

$$Q_m = \frac{C(X_m)}{C(X)} \quad (22)$$

Once the transition matrix  $T$  has been populated, it must be row-normalized. That is, for each row  $m$ , the sum of the row's transition probabilities must always sum to 1.

$$\sum_{n=1}^N T_{mn} \equiv 1 \quad (23)$$

This ensures that each row of the transition matrix represents a proper distribution with Cumulative Distribution Function (CDF) having a maximum value of 1. In other words, given that some state  $m$  is observed, there *must* be a state that follows it.

Finally, it is possible to generate stochastic data from a transition matrix using Markov Chain Monte Carlo (MCMC) methods. In MCMC, an initial state is chosen for the system. A draw  $r$  from a Uniform distribution yields a random number on the interval  $r \in [0, 1]$ . Based on the system's current state  $m$ , row  $m$  of the TM  $P$  is converted to a CDF. Depending on  $r$ 's placement within the row-specific CDF, the system transitions to a subsequent state. Given the above example, assume the system starts in state  $S_1$ . Based on row 1 of the TM in Figure 5, the CDF is shown below. A random draw  $r=0.5$  dictates that the system transitions to state  $S_2$  (highlighted in red in Figure 6, rounding down to the nearest subsequent state value). Another random number is drawn, and the CDF for row 2 is used to govern the transition to subsequent

states, and so on. Note that because the transition probabilities for states  $S_3$ ,  $S_4$ , and  $S_5$  are zero, the system can never transition to these states from  $S_1$ .

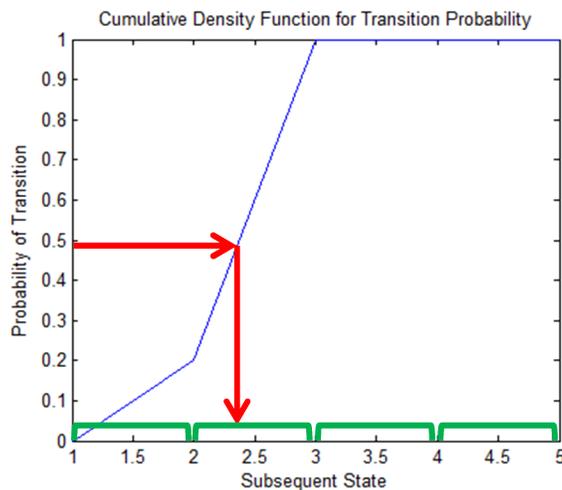


Figure 6. Cumulative Density Function for determining MCMC transition probability.

This process results in data that has identical transitional probabilities to the TM that generated it, but may not match other aspects of a parent dataset. For example, if a TM  $T^{train}$  was obtained based on frequency-dependent data  $y_{freq}$  (e.g., a sine wave), new data generated by performing MCMC on  $T^{train}$  would not have identical frequency domain characteristics to the parent data  $y_{freq}$ . In other words, deterministic reconstruction of data is not necessarily possible through Markov simulation.

There are several special cases of transition matrices that are not wholly applicable to stochastic modeling, because they do not represent ergodic or stationary data. One such example is an “absorbing” state, in which the probability of self-transition is 1. This type of transition matrix occurs when a system reaches a state and can never depart from it. Mechanically, this can happen when a process reaches a fixed steady-state and never resumes motion, such as in local minima or mechanical failure (e.g., a valve sticks and the system remains in that state indefinitely). A simpler analogy of this type of transition matrix is a predator-prey scenario: a

predator attempts to move closer towards a prey, which attempts to flee the predator. The system's state represents the distance between predator and prey; state  $S_1$  represents a distance of 0 (capture), states  $S_2$  through  $S_{N-1}$  represent discrete distances at which the predator continues pursuit, and state  $S_N$  represents the distance at which the predator abandons pursuit (escape). This stochastic system begins in some state between  $S_2$  and  $S_{N-1}$ , but if the system ever reaches  $S_1$  or  $S_N$ , that state will remain forever. A transition matrix representing this type of system is shown in Figure 7: the elements at  $(1,1)$  and  $(N,N)$  are valued at 1, meaning that if the system ever reaches either of those states, it may never leave. The analysis method presented in this dissertation is not designed for stochastic systems with absorbing TM elements, because the hierarchical method relies on the repeated observation of ergodic and stationary time-series data.

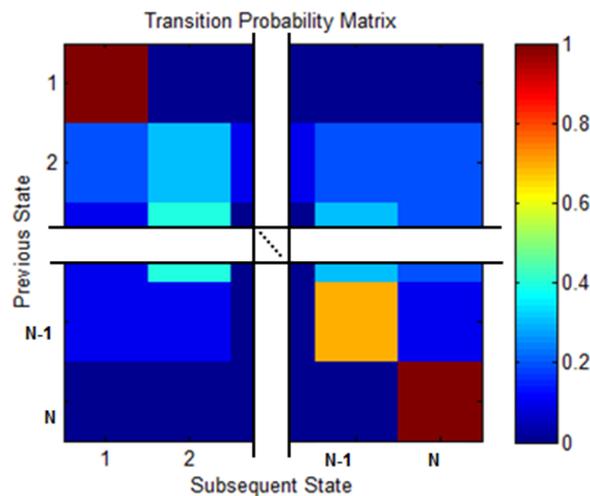


Figure 7. Transition Matrix for predator-prey scenario with two absorbing states

### 1.2.7 Sequential Updating

Sequential updating is a technique used in Bayesian Statistics to recursively estimate an unknown posterior distribution over time using small subsets of incoming measurements. The process may start with a non-informative prior (e.g., a uniform distribution), and update the posterior as new data is added into the model. Bayesian sequential updating is commonly

applied to Code Division Multiple Access (CDMA) signal communication (or “Turbo Coding”) (Berrou, Glavieux, & Thitimajshima, 1993). This approach can be used for signal extraction in noisy data (Wang X. , 2001), the detection of signal or user density changes in CDMA data (Phan & Wang, 2002), and knowledge discovery in accumulative environments (Spiegelhalter & Lauritzen, 1990). The application of sequential updating is not limited to signal processing, and can even be extended to system simulation (Froidevaux, 2004) and modeling of agent-based environments (Gipps & Marksjo, 1985).

The sequential updating scheme in this chapter uses the parameters obtained from iteration  $I^{(k)}$  in the next iteration  $I^{(k+1)}$ . An example graphic is shown in Figure 8.

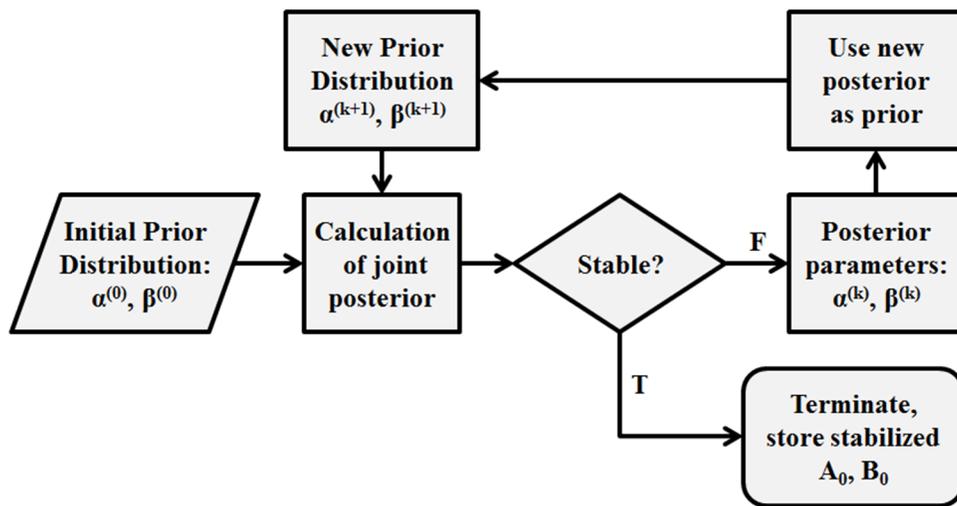


Figure 8. Flowchart of Sequential Updating & Parameter Refinement

This generalized algorithm achieves rapid convergence through Maximum A-Posteriori (MAP) estimates (Friedman & Goldszmidt, 1997). Using proper distributions as opposed to approximations obtained by Monte Carlo methods such as in (Wang, Chen, & Liu, 2002), this stability can be quantified such that the first order statistics become constant. Discretizing this progression yields a simple metric to assess stability. Tracking the area between distributions  $D$ ,

the expected value  $E$ , and the variance  $V$ , shown in Equation (24), results in an eventual termination of the sequential updating loop at user-specified tolerances  $tol_{area}$ ,  $tol_{eval}$ , and  $tol_{var}$ .

$$\begin{aligned}\frac{d^2}{dk^2}(D[\phi]) &< tol_{area} \\ \frac{d^2}{dk^2}(E[\phi]) &< tol_{eval} \\ \frac{d^2}{dk^2}(V[\phi]) &< tol_{var}\end{aligned}\tag{24}$$

Using sequential updating, it is possible to encounter large-number problems in computing, necessitating the use of log-likelihood calculation. However, as the per-iteration complexity and memory requirements are lower than that of parallel updating, sequential updating is the generally preferred method (Kfir & Kanter, 2003).

### 1.2.8 Autoregressive Moving Average Models

Autoregressive Moving Average (ARMA) models are an industry standard for working with stochastic causal time-series data. ARMA describes a stochastic process in terms of two polynomials: one that determines future values in the time-series based on prior values (Autoregressive), and one that incorporates a known variance into the time-series data based on prior variances (Moving Average). In ARMA, the “order” is the level of complexity that stems from the length of time, in discrete number of prior observations, used to predict or determine future system values (Benjamin, Rigby, & Stasinopoulos, 2003).

The Autoregressive (AR) part of the ARMA model incorporates a constant bias  $c$  and a one-dimensional coefficient matrix  $\Phi$  of order  $p$ . Together, these variables form  $y_{ar}$ , the autoregressive component of a data point  $y$ :

$$y_{ar} = c + \sum_{i=1}^p [\Phi_i y(t-i)]\tag{25}$$

The Moving Average (MA) part of the ARMA model incorporates a statistically random shock input  $\varepsilon_t$ , and a one-dimensional coefficient matrix  $\Psi$  of order  $q$ . Together, these variables form  $y_{ma}$ , the moving average component of a data point  $y$ . In some MA models, the random error term  $\varepsilon_t$  has its own coefficient  $\Psi_0$  to account for the immediate effect of noise in the system.

$$y_{ma} = \varepsilon(t) + \sum_{i=1}^q [\Psi_i \varepsilon(t-i)] \quad (26)$$

By combining these terms, the Autoregressive Moving Average model formula is combined to be:

$$y = c + \varepsilon(t) + \sum_{i=1}^p [\Phi_i y(t-i)] + \sum_{i=1}^q [\Psi_i \varepsilon(t-i)] \quad (27)$$

In order to incorporate this noise, the random variable  $\varepsilon$  is assumed to be an i.i.d. (independent and identically distributed) sample from a Gaussian distribution, such that:

$$\varepsilon \sim N(0, \sigma^2) \quad (28)$$

M. Benjamin has extended this approach into non-Gaussian frameworks as Generalized Autoregressive Moving Average Models (GARMA), though both processes are still generalized linear models with no exponential components in either the AR or MA algorithms. M. Benjamin's work also cites the computational complexity of MCMC as being a drawback to rapid system estimation, which will be discussed further in Chapter 3.

Generating stochastic time-series data from an ARMA is a relatively simple process. For this algorithm, the coefficients to the ARMA system are:

$c$ , a constant DC offset

$\Phi$ , the matrix of coefficients for autoregressive calculation

$\varepsilon$ , the coefficient for the shock input as  $\varepsilon \sim N(0, \sigma^2)$

$\Psi$ , the matrix of coefficients for moving average calculation

$T$ , the number of time-series points to generate

$y_0$ , a starting value for the system

An ARMA model higher than first-order (that is,  $p$  or  $q$  are greater than 1) must contain an initial burn-in period during which time the AR or MA portions do not fully account for prior data, as prior data has not yet been generated. This burn-in period only accounts for time-series points having length  $p$  or  $q$ , whichever is greater. After sufficient time has passed, the system may stabilize, though it is possible to specify coefficients which cause instability and eventually approach infinity (Box, Jenkins, & Reinsel, 1994). This divergence occurs when either of the AR or MA matrices have a magnitude larger than 1:

$$\lim_{y \rightarrow \infty} (ARMA) = \pm\infty \text{ if } |\Phi| \geq 1 \text{ or } |\Psi| \geq 1 \quad (29)$$

Following the burn-in period (when sufficient data exists to index  $y_{t-p}$  and  $\varepsilon_{t-q}$ ), the algorithm for data generation is as follows:

```
for t=max(p,q), t<T, t++ {  
    AR = c + P*transpose(flip(Y(t-p:t-1)))  
    E(t) = e*randnorm(0,1)  
    MA = E(t) + Q*transpose(flip(E(t-q:t-1)))  
    Y(t) = AR + MA  
}
```

In this algorithm, the function *transpose* returns the transpose of a linear array, and *flip* reverses the order of a linear array. The function *randnorm* returns an i.i.d. value from a Gaussian distribution with mean 0 and standard deviation 1. This method utilizes matrix math operations

in the calculation of the AR and MA portions of the algorithm, instead of separately calculating the individual values of  $\Phi_i * y_{t-i}$  and  $\Psi_i * \varepsilon_{t-i}$  for each loop iteration.

### 1.2.9 Neural Networks

Neural networks, often referred to as Artificial Neural Networks (ANNs) in the fields of machine learning and artificial intelligence, mimic the way neurons interpret data. A physical neuron receives information, often from multiple sources, through dendrites. These dendrites pass into the nucleus, which determines whether or not the input should cause the neuron to “fire”. This firing happens when the action potential of the axon is overcome, and a signal travels from the nucleus to other neural cells (Snell, 2006). An artificial neuron works in much the same way. Inputs  $y$  pass into an artificial nucleus  $N$  through weights  $w$ . If the weighted inputs meet the activation requirement in Equation (30), the artificial neuron fires, meaning that it returns a digital 1 (for “true”), or fails to fire and returns a digital 0 (for “false”).

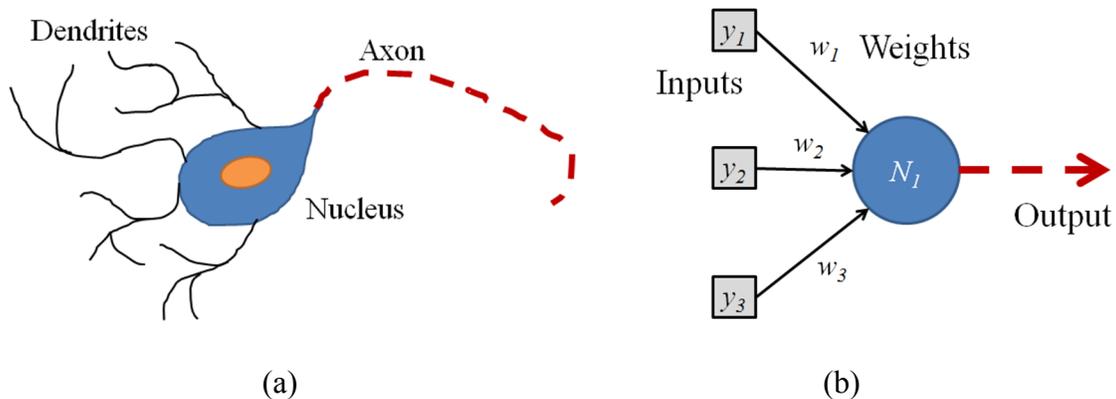


Figure 9. Basic diagram of a physical neuron (a) and an analogous artificial neuron (b)

These systems can be trained to recognize patterns and detect anomalies in a variety of data sources. In an ANN, data are combined into “neurons” which weight each incoming dimension of the data. These artificial neurons output a binary state, much like biological neurons, dependent on their activation threshold. The outputs of one layer of neurons can be fed

into another, and another, until a final decision is made by the network. This is called a feed-forward network, shown in Figure 10.

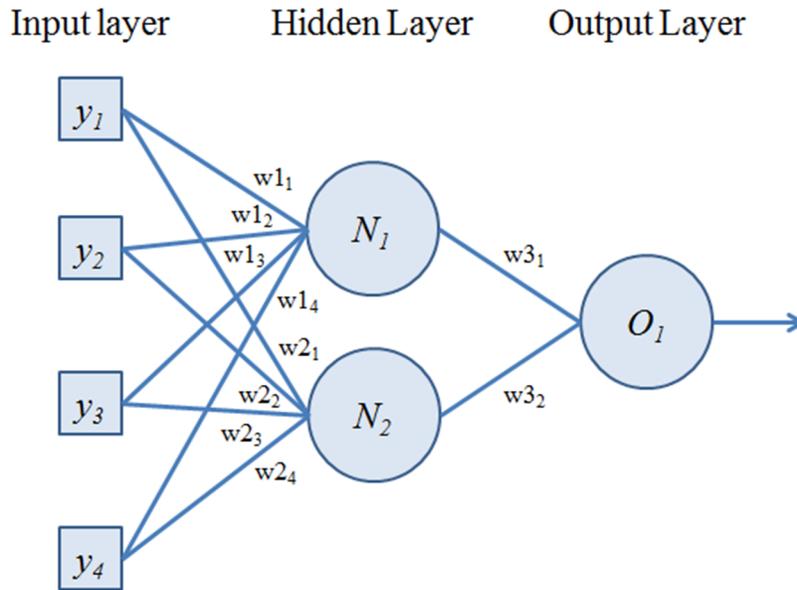


Figure 10. Example Feed-Forward Artificial Neural Network

The input layer consists of data  $Y$  having elements  $y_1, y_2 \dots y_n$ . This input array is passed into each neuron in the hidden layer  $N_1, N_2 \dots N_m$  after being weighted by weighting vectors  $w_{11}, w_{12} \dots w_{m_n}$ . Finally, the neurons are combined through additional weighting vectors  $w_{o1}, w_{o2} \dots w_{om}$  into an output layer  $O$  that represents the “decision” of the overall network.

Each neuron’s activation function depends on its inputs  $y_1, y_2 \dots y_n$ , weighting vectors  $w_1, w_2 \dots w_n$ , and some bias weight  $w_0$  (Russell & Norvig, 2003). These combine to form the neuron’s activation function  $f$  shown in Equation (30):

$$N = f\left(w_0 + \sum_{i=1}^n y_i w_i\right) \quad (30)$$

Training an ANN may be done through supervised, unsupervised, or reinforced learning. The most straightforward of these is the supervised learning case, where a user provides sample pairings between known inputs and desired outputs. For all ANN learning processes, a cost

function must be defined between the network output and target values. This cost function can be specified by the user, and defines the difference between the network output and target values (Duda, Hart, & Stork, 2001). With some initial weighting scheme for the vectors  $\mathbf{w}_1, \mathbf{w}_2 \dots \mathbf{w}_m$ , the training of the network involves minimizing the cost function (often a mean-squared error or similar least-squares function). For each pair of input and desired output vectors, the cost function is minimized and the set of  $\mathbf{w}$  vectors is updated to reflect a new best-match. This results in a neural network that can “recognize” an input similar to all inputs presented to the network during the training phase.

As shown in (Duda, Hart, & Stork, 2001), a sum-of-squares difference cost function  $C$  can be incorporated as:

$$C(\mathbf{w}) = \frac{\|\mathbf{t} - \mathbf{o}\|^2}{2} \quad (31)$$

By finding the derivative of this cost function, the learning process takes a measured step based on some learning coefficient  $\eta$  to reduce the cost with a change in  $\mathbf{w}$ . This finally yields the updating scheme:

$$\mathbf{w}(k + 1) = \mathbf{w}(k) - \eta \frac{dC}{d\mathbf{w}} \quad (32)$$

One example of this type of learning process is the application of ANNs to character recognition for printed text (Gonzalez & Woods, 2008). For all ten decimal digits, there can be a 5x5 image with binary (0,1) pixels that uniquely represents each digit. These digits (0 through 9) are shown below in Figure 11:



Figure 11. Digits 0 through 9 as 5x5 image arrays

A simple feed-forward neural network would incorporate 25 input array elements, 10 artificial neurons within the hidden layer trained on each digit, and one neuron final output layer to return a decision on whether or not a digit has been identified. Alternatively, the output layer may contain ten neurons and return information about which digits had been identified within the hidden layer, and return this information with a known degree of certainty.

This core concept drives the use of artificial neural networks, but also shows one of their key weaknesses: the need for *a-priori* information about the types of anomalies targeted for detection. Using the above example, a neural network may return a false-positive match for the digit 8, when the true input is actually the capital letter “A” (an anomaly in the context of number identification), as the two input arrays differ only by one bit:



Figure 12. An anomaly letter “A” relative to known pattern “8”

While ANNs are robust tools for pattern recognition, forecasting, and analysis in areas such as financial markets (Azoff, 1994) (Baba, 2006) they can fall short of other methods for

novel anomaly detection and change-point detection (Ho, Xie, & Goh, 2002) (Ribiero, 2007).

Prior learning and user input remain key hurdles in using Neural Networks.

### 1.2.10 Hierarchical Approaches

In hierarchical statistical models, data are assumed to be driven by parameters, which are in turn driven by other parameters, at different levels in a hierarchy. In Bayesian statistics, this is often modeled through the use of multi-layer posterior analysis. For example, the Gaussian distribution is also a conjugate prior for the mean of a Gaussian distribution with fixed variance. For data  $y$  drawn from a Gaussian distribution with parameters  $\mu$  and  $\sigma$ :

$$y \sim N(\mu, \sigma^2) \quad (33)$$

the variance parameter  $\sigma^2$  is assumed to be constant. The mean parameter  $\mu$  is assumed to be driven by a Gaussian distribution with parameters  $\mu'$  and  $\sigma'$ :

$$\mu \sim N(\mu', \sigma'^2) \quad (34)$$

With similar assumptions about the base distribution for  $y$ , the parameters  $\mu'$  or  $\sigma'$  can be assigned their own hierarchical models. In this way, the data is driven by distributions with parameters that are driven by their own distributions, and so on until the top level of the hierarchy reveals the pertinent information about the system (Gill, 2009). Figure 13 shows the first 3 levels of one such Bayesian hierarchy.

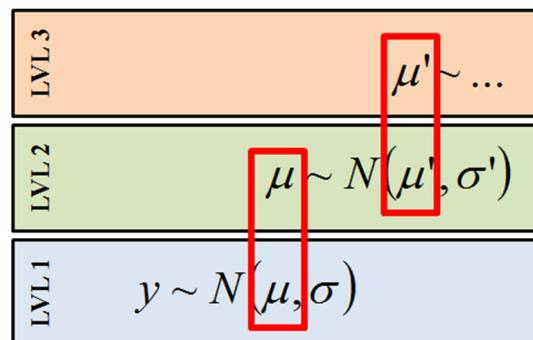


Figure 13. An example Bayesian hierarchical model for mean parameters with fixed variance

Other hierarchical models involve images at varying resolutions (an example of which can be found in Section 2.1.1.3), as well as hierarchies of importance for data containing multiple layers of information such as multispectral imagery. For the purposes of this dissertation, a hierarchy is defined as any data structure with causally-linked layers of information. That is, upper levels of the hierarchy *dictate* the system's behavior at lower levels, and lower levels can only be used to *infer* information about higher levels.

## **2 PROBLEM DEFINITIONS AND LITERATURE REVIEW**

This work demonstrates the usefulness of a hierarchical paradigm for detecting anomalies in a variety of data types. Because of the differences between these data types, it is necessary to first discuss previous approaches tailored to each type. The literature review in this chapter deals with time-series data, parameter-driven data, and spatiotemporal data. Broad techniques and specific methods are both given consideration in this chapter.

### **2.1 Anomaly Detection in Time-Series Data**

Times series data are common in mechanical systems, and the detection of time-series anomalies is critical for mechanical system maintenance. A variety of analysis methods exist for processing this type of data, such as array clustering, thresholding, wavelet analysis, probabilistic models, and neural networks (Ott & Longnecker, 2001). Pattern recognition in time-series data is an especially relevant topic of research, because anomaly detection is an extension of pattern recognition. Algorithms such as decision trees, rule-based classification, nearest neighbor analysis, and Bayesian classifiers are some of the techniques used in the data mining industry (Sankar & Mitra, 2004). This section deals with several related approaches to those described in the later chapters of this dissertation, both to illustrate the differences between approaches and compare advantages & disadvantages between methods in a variety of data types.

#### **2.1.1 Previous approaches using Transition Matrices**

This section details several previous approaches to analysis methods involving Transition Matrices, which are formally defined in section 1.2.6. These transition matrices are not to be confused with the State Transition Matrix used in control theory in conjunction with the state vector (for state-space modeling). These STM matrices are deterministic, not probabilistic, and are used to obtain “solutions” based on the structure of systems (Brogan, 1991).

### 2.1.1.1 Raw Probability Calculation Methods

Some previous approaches to anomaly detection using transition matrices have focused on the calculation of experimental probabilities of subsets of time-series data. N. Ye proposes a method that calculates the probability of occurrence of a series of time-series observations ( $Y_1, Y_2, \dots, Y_t$ ) based on a previously-observed transition matrix  $T$  (Ye, 2000). For any state  $m$ , elements of  $Q$  (the initial probability distribution that determines the probability that the system is in any given state) are calculated with Equation (35):

$$q_m = \frac{\#[m]}{\#[Y]} \quad (35)$$

Using a window of length  $L$  of time-series data, the probability  $P(\bullet)$  of a series of events  $Y_{1,2,\dots,L}$  is at any given time  $t$  is calculated as:

$$P(Y_{t-L+1}, Y_{t-L+2}, \dots, Y_t) = Q_{Y_{t-L+1}} \prod_{i=t-L+2}^t T_{Y_{i-1}, Y_i} \quad (36)$$

The higher the probability, the more likely that the series of observations  $Y_{1,2,\dots,L}$  fit within the normalcy of previously observed data. Figure 14 shows an example 5-state transition matrix  $T$  and the transition probabilities  $i=(1,2, \dots, 8)$  for a time-series sequence  $Y_{1,2,\dots,L} = \{1, 2, 4, 1, 3, 5, 3, 4, 2\}$ . In Figure 14, the transition  $i=1$  corresponds to the sequence  $\{1, 2\}$ , and is highlighted at pixel  $(1, 2)$ . For this element, the probability of arriving at state 2 given a previous state of 1 is approximately 0.35, given by the color of the pixel.

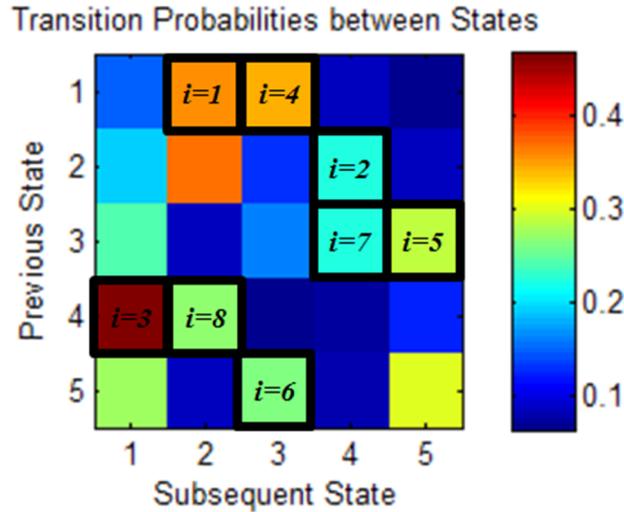


Figure 14. Example transition matrix probability calculation scheme

In the work by (Ye, 2000), this calculation generated normal probabilities between  $1 \times 10^{-57}$  and  $1 \times 10^{-21}$ ; anomalous data resulted in transition probabilities between 0 and  $1 \times 10^{-56}$ . Some risks of using such a method include round-off error in the calculation of probabilities, as well as the chance of false-positive alerts from normal data falsely triggering an anomaly detection threshold. The final determination of normalcy is not based on statistical hypothesis, but instead on a user-defined threshold that must be determined for each potential dataset. This is discussed further in Section 3.5.3; the method described here does not perform a probability calculation of the “normalcy” of time-series data itself, but instead finds the probability that the transition matrix generated from some time-series data fits within “normal” expected parameters.

### 2.1.1.2 Probability Vector Comparison Methods

A more recent approach to using transition matrices for anomaly detection expands the dimensionality of the transition matrix itself from 2 (as presented here and in previous methods) to a D-Markov space. From (Khatkhate, Gupta, Ray, & Patankar, 2008):

**Definition 1:** *A symbolic stationary process is called D-Markov if the probability of the next symbol depends only on the previous D symbols*

In other words, if a system may occupy  $N = 2$  discrete states, a Markov process of order  $D=2$  would have  $D+1$  dimensions and would occupy a 2 by 2 by 2 non-reduced transition matrix. An example of such a process is shown in Figure 15. In this figure, a 3-dimensional transition matrix is shown with previous states on the Y and X axes, and subsequent states on the longitudinal Z-axis. The highlighted matrix element at (1,2,1) is the probability of arriving at state {1} given that the system was previously in state {1}, then {2} (or more simply, state {1 2}).

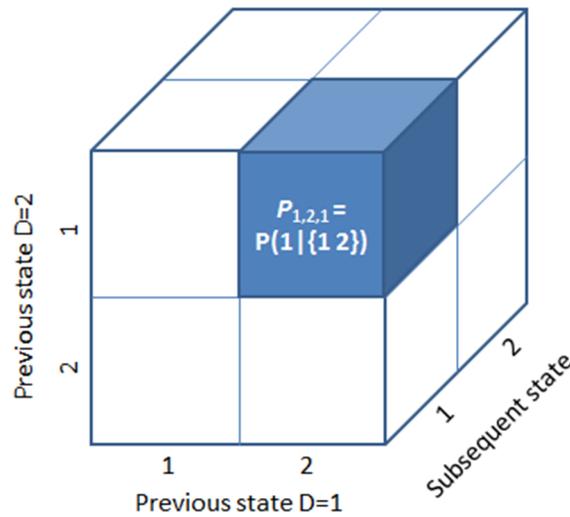


Figure 15. D-Markov transition matrix where  $D=2, N=2$

In such a D-Markov process, the number of elements in the transition matrix grows exponentially with  $D$  as there become far more possible permutations of previous states for every subsequent state. In the example above, previous states may be {1 1}, {1 2}, {2 1}, and {2 2}. If  $D$  were increased to 3, then there would be 8 possible previous sequences of states: {1 1 1}, {1 1 2}, {1 2 1}, {2 1 1}, {1 2 2}, {2 1 2}, {2 2 1}, and {2 2 2}. It follows that for increases in  $D$ , there must be exponentially more time-series observations to fully occupy such a transition matrix; otherwise, the TM will be sparse and methods that rely on adequate sampling may fail to successfully detect anomalies.

To avoid TMs with many high-order dimensions, many methods obtain the irreducible transition matrix (defined as  $II$ ) which defines a “state” as the sequence of states. In other words, if the above example is used ( $D=2, N=2$ ), there are 4 possible previous states, and 4 possible subsequent states (looking forward 2 “system” states instead of 1).

Both (Khatkhate, Gupta, Ray, & Patankar, 2008) and (Ray, 2004) implement probability vectors  $\mathbf{p}_0$  (obtained from observing normal time-series data) and  $\mathbf{p}$  (obtained from candidate, or “test” time-series data). These probability vectors are the 1-dimensional reduction, or “unwrapping”, of the transition matrix  $T$ . Using a Euclidian distance measure defined as:

$$M_k = \left[ \sum_{j=1}^n (\mathbf{p}_k(j) - \mathbf{p}_0(j))^2 \right]^{\frac{1}{2}} \quad (37)$$

these methods compute the normalcy of candidate data, based on the distance-to-normal, relative to known normal training data. From (Ray, 2004), the forward problem (detection of anomalous data) is solved by:

*(F1) Selection of an appropriate set of input stimuli.*

*(F2) Signal–noise separation, time interval selection, phase-space construction.*

*(F3) Choice of a phase space partitioning to generate symbol alphabet...*

*(F4) State Machine [and] determining the connection matrix.*

*(F5) Selection of an appropriate metric for the anomaly measure  $M$ .*

*(F6) Formulation [of a] relation between the computed anomaly measure and known physical anomaly under which the time-series data were collected...*

In step (F5) the anomaly metric  $M$  must be determined by user input after the initial learning period has taken place, and (F6) requires additional formulation of anomaly detection and physical presentation. The method presented in this dissertation significantly differs from

previous works by instead parameterizing the likelihood of transition matrix element values, and hierarchically determining whether or not a candidate transition matrix was likely to have been generated by “normal” data. Making the process of anomaly detection statistically rigorous, the novel method presented in Chapter 3 only requires confidence intervals from the user, which can be specified even before any training takes place. This is due to the statistical nature of the final anomaly detection process.

### 2.1.1.3 Hierarchical Transition Matrix Methods

Hierarchical models have previously been applied to transition matrices, but not in the Bayesian sense outlined in Section 1.2.10. Other hierarchical approaches have been applied to the State Transition Matrix (STM) in control theory, a subject not covered by the scope of this dissertation (Kong, Katahira, Watanabe, Katayama, Hisazumi, & Fukuda, 2011). Instead, (Patil & Taillie, 2001) prescribes a hierarchical approach to transition matrix analysis that treats a raster map as an image, and its elements as conglomerate pixels which are in turn driven from other conglomerated pixels, until a final single pixel defines the entire image. This hierarchy scheme is shown Figure 16:

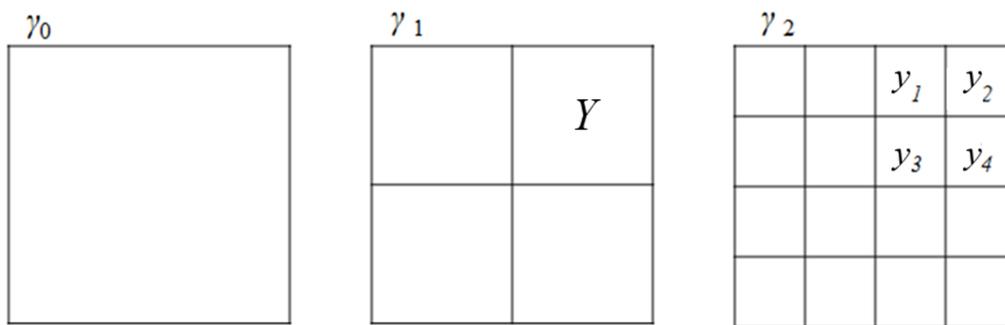


Figure 16. Nested hierarchy of pixels.

In this approach, each raster map  $\gamma_n$  is generated from a transition matrix, which is defined in the referenced work as a matrix variable  $G$ . These transition matrices fit into the

hierarchy at raster generation levels such that the transition matrix  $G^{[n,n+1]}$  dictates the generation scheme between  $\gamma_n$  and  $\gamma_{n+1}$ .

For the method in (Patil & Taillie, 2001), assume that there are  $K$  discrete pixel values which may occupy a pixel of  $\gamma_n$ . This requires  $K$  by  $K$  transition matrices  $G^{[n,n+1]}$ . The variability of the transition matrices causes the final map  $\gamma_n$  to exhibit different spatial patterns at different scaling levels. G. Patil characterizes detection metrics in terms of the eigenvalues of the transition matrix  $G$ . The eigen-decomposition of  $G^{[n,n+1]}$  provides information about the raster maps. However, this is not directly linked to any process of anomaly detection or pattern recognition; the method presented in Chapter 3 is developed specifically for the purpose of anomaly detection in time-series datasets.

### **2.1.2 Previous approaches using Neural Networks**

While not implemented as part of the methods developed in this dissertation, Neural Networks are a widely-used approach for data analysis. Given their popularity, it is important to acknowledge several of the ways that ANNs have been applied to anomaly detection and predictive modeling. Damage detection specifically has been a topic of significant research; this section details two relevant approaches using ANNs as anomaly detection metrics. Giurgiutiu and Kropas-Hughes provide an in-depth comparative summary of Neural Networks in electro-mechanical impedance spectra (Giurgiutiu & Kropas-Hughes, 2002). Their study focuses on damage that manifests through vibrational anomalies, which are classified through a variety of Neural Network types.

Defects in EM impedance plates were analyzed at various frequency bands: 10-40 kHz, 10-150 kHz, and 300-450 kHz. This high-frequency analysis allowed classification and localization of damaged units, but required significant data to perform accurately. Detection of

anomalies was dependent on multiple frequency features: if fewer than four frequency features (defined in the text as “major peaks”) were present, misclassification became a significant error. When eleven or more features were used for classification, in conjunction with probabilistic neural networks, detection was accurate and robust. The Giurgiutiu study illustrates one of the key advantages of Neural Networks in general: the ability to process seemingly arbitrary data into meaningful groups and classifications. However, this technique also requires a significant amount of data and (in the referenced article) feature extraction of known anomaly types for robust detection.

Yang Yuan et al., develop a hierarchical Neural Network approach to anomaly detection in computer datacenters (Yuan, Lee, Pompili, & Liao, 2011). In Yuan’s work, the authors clarify the disadvantages of threshold-based approaches that rely on user specification of “acceptable” operating ranges for the system. This approach is successful in detecting both small- and large-scale anomalies through the hierarchical nature of the neural network framework, similar to the architecture shown in Figure 13. In Yuan’s work,  $2 \times 10^4$  data were used for each training interval, and require 150 seconds of data for accurate detection. Modeling approaches also rely on prior knowledge of the system; in the case of thermal datacenter analysis, this requires expertise in thermodynamics and fluid flow mechanics. The approaches in this dissertation attempt to reduce this data requirement to yield more rapid detection of anomalies and system damage.

## **2.2 Anomaly Detection in Parameter-Driven Data**

Most industrial and mechanical systems exhibit measurable signals (temperature, vibration, distance / aspect size, etc.) which act as life-cycle indicators and may serve as precursors to system failure. The detection of anomalies in these types of systems can reduce the

cost of faults, reduce system downtime, and improve the overall quality of manufacturing results (Hu, Subbu, Bonissone, Qiu, & Iyer, 2008). Condition-based maintenance (CBM) is becoming a subject of interest in the field of industrial systems monitoring. In some cases it is practical to monitor the manufacturing equipment itself, as opposed to the resulting parts. This can be accomplished by measuring the acoustic emission or vibration signature of a piece of machinery, and monitoring this signal for changes (Lee, Hwang, Valente, Oliveria, & Dornfeld, 2006). These approaches perform well when there is ample data, when the Signal to Noise Ratio (SNR) is high, and when there is prior knowledge of the faults to be detected. When signal monitoring is infeasible, quality assurance systems typically rely on the measurement of the produced parts themselves (herein, “units”). When anomaly detection is performed on units, it is desirable to use as few as possible. In many cases, testing is expensive or even requires the destruction of units (Allen, 2010). By using simulation models, fewer physical units are needed for testing. Additionally, methods that require few sample units for testing reduce the overall cost of system monitoring.

### **2.2.1 Approach Background and Survey**

Monitoring precision in tolerance-driven systems is an important aspect of modern manufacturing processes. Previous approaches use accurate but computationally intensive models, such as Artificial Neural Networks (ANN’s) (Monostori & Prohaszka, 1993) (Hou, Liu, & Lin, 2003), pattern recognition, fuzzy logic, and decision trees (Du, Elbestawi, & Wu, 1995). The Cumulative Sum (CUSUM) technique is a commonly-applied method for monitoring the variance of components, especially in small-parts manufacturing and wafer-thin technology (Yashchin, 1994), and for monitoring structural changes (Chu, Stinchcombe, & White, 1996). (Yashchin, 1994) explores the use of variance monitoring for IC quality control using

components such as the distance of a chip from the center of a wafer, locations of wafers within a lot, and differences between raw materials and operators. Principal Component Analysis (PCA) is also widely used for monitoring statistical processes (Qin, 2003) (Li, Yue, Valle-Cervantes, & Qin, 2000), but becomes computationally expensive with increases in dimensionality such as with high resolution images.

Previous metrics for statistical significance would classify small sample sizes, on the order of 20 units, as inadequate for traditional hypothesis testing. For example: given a large population size and a 2% margin of error rate with a 99% confidence interval for testing, a best-case sample size requirement for the data is more than 160 (Hamburg, 1985). Recent approaches using Cumulative Sum (CUSUM) and forward-looking Moving Sum-Of-Squares (MOSUM) have achieved reliable monitoring using sample sizes as small as  $n=60$  (Hsu, 2007).

### **2.2.2 CUSUM algorithm background**

Control charts have been well established as an appropriate method for detecting when stochastic processes are behaving normally (Barnard, 1959). The CUSUM (short for CUMulative SUM control chart) is an analysis technique designed for monitoring change detection. Originally developed by E. S. Page in 1954, CUSUM is an algorithm that determines whether a process is in control or out of control depending on sample data's comparison to prior observations (Page, 1954). A simple and straightforward explanation for the CUSUM's application to change-point and anomaly detection is found in (Brodsky & Darkhovsky, 2000):

*Being a nonparametric methodology, a CUSUM detector detects change points, i.e., the rising and falling edges of the anomalous period, which denotes the duration that the anomaly lasts, without prior knowledge on the distribution of the traffic flow. When the mean of a subset of samples becomes higher than a*

*threshold, a change point is registered, and the level of change is cumulated into an accumulator. The accumulator begins to discharge when the other change point is detected at end of the anomaly period.*

The CUSUM algorithm tracks the cumulative sums of deviations of sample values  $X_i$  from a target value  $T$  over time. This summation occurs as:

$$S_i = \sum_{i=1}^n (X_i - T) \quad (38)$$

where  $n$  is the number of samples tested. The upper and lower limits of the CUSUM-analyzed process are governed by a reference value  $k$  shown in Equations (43) and (40):

$$S_{i,hi} = \max[0, S_{hi,i-1} + X_i - (T - k)] \quad (39)$$

$$S_{i,lo} = \max[0, S_{lo,i-1} - X_i + (T - k)] \quad (40)$$

The above values are compared to some threshold  $H$ , which serves as a decision point to determine if the process is out of control. This value can be user-specified, or derived from prior learning from the averaged run length (ARL) between each false alarm in known normal data (Page, 1963)

### **2.2.3 Limitations of Point Change Detection**

There have been numerous examples of point change detection algorithms applied to industries such as financial markets and network security. One of the more common applications the detection of anomalous volumes of network traffic to identify Denial of Service (DoS) attacks. (Wang, Zhang, & Shin, 2004) analyze network traffic volume (number of packets sent over the network) to identify when traffic levels exceed the expected limits. Unlike some other network traffic volume analyzers, this method does not require absolute time references; instead of treating each day as a time vector, this method operates continuously without any absolute

timestamp. Methods such as CUSUM and MOSUM do not rely on a timestamp to reference a signal relative to the learning history, they instead incorporate all previously-observed data into the subsequent residual distance-from-norm.

Point change detection methods are limited by the need for sufficient data. This sufficiency requirement is explored generally in related literature, but there is no unifying equation to formalize how much data is truly “necessary” for statistical inference. Different hypotheses require different sample sizes, and the variety of test metrics prevents standardization of sufficiency (Dell, Holleran, & Ramakrishnan, 2002). In recent point-change analyses, 3000 samples were found to be sufficient for detecting congestion in network flows (Rubenstein, Kurose, & Towsley, June, 2000). A related study achieved robust detection using wavelet-based analysis with only 1000 samples (Kim, Kim, Shin, & Lam, August, 2004). This dissertation proposes methods to incorporate prior learning with statistical rigor, significantly reducing sample size requirements for accurate detection. Because there is no rigorous definition for sufficient sample size, this dissertation compares results using ROC curves and processing times at various sample sizes to demonstrate a quantitative advantage of the developed methods.

### **2.3 Anomaly Detection in Spatiotemporal Data**

There are number of established anomaly detection approaches commonly applied to the processing and analysis of large-scale datasets. Neural networks are able to automatically learn and predict motifs in low-noise data (Chiu, Lonardi, & Keough, 2003), and discrete convolution integrals have been used in combination with neural network learning to automatically find previously-defined events and behavior in video information (Ji, Xu, Yang, & Yu, 2010), and in spatiotemporal sensor environments (Sanso & Schmidt, 2004). Dimensionality reduction is a typical first step in addressing a high-dimensionality dataset; this ranges from reduction of

information depth (Kumar, Lolla, Keough, Lonardi, & Ratanamahatana, 2005) (Lin, Wei, Lonardi, & Keough, 2007), to the reduction of multi-dimensional data into its principal components (Shlens, 2005).

Principal component analysis and eigenspace reduction are both popular approaches to processing high-dimensionality data, sometimes combined with graph theory to improve detection of known events (Ide & Kashima, 2004). The approach developed by Ide and Kashima is applied exclusively to computer systems (virtually spatial networks). In the approach for network analysis, this convolution approach is used to extract information about known event signatures from the data. By observing known normal training data, this approach uses supervised learning to identify “events” in the data, and models those events probabilistically in a hierarchical framework.

Much work has been done in the area of anomaly detection using the Poisson distribution to model time series of network traffic flow (Chen D. , 2008), monitor the occurrences of clustered interactions between humans (Panangadan, 2004), and investigate anomalies driven by changes in weather, such as El-Niño (Sanzo & Schmidt, 2004). Both (Chen D. , 2008) and (Panangadan, 2004) apply the Poisson model and flag anomalous events after some comparison to a threshold value, whereas (Devore, 1995) uses a Poisson rate parameter to characterize the probability of trends in the anomaly rate for yearly weather. These detection metrics can be validated against other existing methods of anomaly detection in ground-truth weather data (Rajagopalan e. a., 1997).

Although much work has been done in the field of small-area intelligence using binary sensors (Nakashima & al., 2009) (Singla & al., 2010), there is little existing work on sensor processing in open environments where many different agents are present (Wren & al, 2006).

Two principal hurdles in processing this data are sensor noise (false alarms) and relating sensor alarms to individual agents moving through the environment. Additionally, these alarms must be able to describe overall activity taking place in the observed environment, and be able to relate these activities to normal or abnormal events. To accomplish this, GEOINT data can be integrated into a statistical framework to relate sensor alarm data to actual agents moving along trajectories within the environment (Singla & al., 2010).

### **2.3.1 Network Traffic Analysis**

This dissertation refers to sensor networks monitoring a physical environment, but previous work in spatiotemporal network anomaly detection has been performed in the field of network traffic analysis (e.g., IP telephony). As before, statistical methods are used to perform anomaly detection in time-series data that is spatially linked. Previous methods such as (Roesch, 1999), (Paxson, 1999), and (Yegneswaran, Griffin, Barford, & Jha, 2005) tend to classify “intrusions” as anomalies, but require prior knowledge of the intrusion type in order to perform detection. (Paschalidis & Smaragdakis, 2009) use a statistical method in two parts: a “model-free” approach to categorize an i.i.d. sequence of aggregated system activity, and a “model-based” approach using a Markov Modulated Process similar to those found in Sections 1.2.5 and 1.2.6.

### **2.3.2 Discrete Convolution Approaches**

Kernel filtering is an image processing technique that uses discrete convolution. A smaller kernel (typically only a few pixels in in size) is convolved across a parent image. The kernel is often normalized to have a total value of zero, so the convolution does not add a uniform bias to the result. By convolving different kernels across a parent image, different

results yield information (or cause different effects) such as edge detection, blurring, or sharpening (Davies, 2005).

Many approaches are applied to the field of image processing, such as probabilistic interpretation of images (Sunhil & Desai, 2001), and multispectral processing for data with multiple layers of information (Lucas & Kanade, 1981) (Ricchetti, 2000). By treating spatiotemporal data as an image (with one axis reserved for spatial data, and one reserved for time), image processing techniques can be applied to extract information and perform anomaly detection in these types of datasets.

## 2.4 Anomaly Detection in Rotating Machinery

Previous industry standards have focused on identifying anomalous behavior in rotating machinery, one of the key focuses of this dissertation. Bearings typically consist of four main components; the inner raceway, outer raceway, ball, and cage, as shown in Figure 17. The raceways are often lubricated to decrease friction against the rolling elements, and the cage guides the rolling elements during operation by maintaining equal spacing.

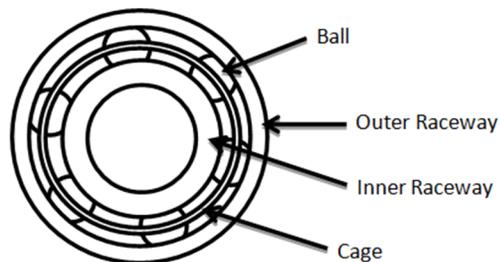


Figure 17. Example ball bearing assembly, cross-section

Discrete faults on any part of the bearing lead to some form of vibration change in the system, so monitoring system vibration is one method of assessing the type, location, and severity of faults. However, these faults can also produce other measurable phenomena: shock impulses, temperature increases, and acoustic changes, which can also be used to detect defects

(Kumar & Tandon, 2003) (Draney, 2007). Other works have implemented image analysis of disassembled bearings to diagnose faults, which is time-consuming and requires disassembly and reassembly of the system (Ng, 2006).

Previous anomaly detection methods such as Bayesian filtering, artificial neural networks (ANN) and statistical methods such as Principal Component Analysis (PCA) and (Extended) Kernel Regression Analysis (KRA) have been compared against the symbolic Nearest Neighbors matching approach, which has been shown to exhibit greater capability of detection with low false alarm rates (Rao & Ray, 2009). A chief disadvantage of using Nearest Neighbor searches is their computational expense. Some approaches attempt to reduce the computational burden of storing large time-series datasets. (Burgess, 2002) proposes a method that reduces time-series signals that are relatively periodic into “two-dimensional time”. This approach relies on the periodicity of computer signals and reparameterizes time in topological slices of period  $P$ . The formal parameterization is shown in Equation (28):

$$t = nP + \tau \quad (41)$$

The approach in Chapter 5 relies on a similar paradigm for handling periodic time-series data. For sensor network traffic analysis, the daily traffic in the network is assumed to be predictable and periodic over a 24 hour period, and again over the course of weeks. In Chapter 3, the Hierarchical Transition Matrices Method (HTMM) is introduced to improve computational efficiency and anomaly detection reliability, and is compared to the Nearest Neighbors Euclidean norm method. While rotating machinery signals are periodic, the frequency of operation may be too high to yield adequate sampling for robust anomaly detection.

Current industry standards such as ISO 10816 do not incorporate the stochastic nature of vibration data for machine fault detection. This measurement standard separates machines into

classes by size and support foundation. While this specification is restricted to machines with a nominal power above 15 kW and speeds between 120 and 15000 rpm, the ISO table shown in Figure 18 recommends condition monitoring independent of rotational speed (International Organization for Standardization, 2009).

ISO 10816-3		Medium-sized machines		Large machines	
Advisor		Group 2		Group 1	
Velocity		Rated Power			
in/sec eq. Peak	mm/sec RMS	15 kW – 300 kW		300 kW – up	
0.61	11.0	<b>DAMAGE OCCURS</b>			
0.39	7.1				
0.25	4.5	<b>RESTRICTED OPERATION</b>		<b>UNRESTRICTED OPERATION</b>	
0.19	3.5	<b>UNRESTRICTED OPERATION</b>			
0.16	2.8	<b>NEWLY COMMISSIONED MACHINERY</b>			
0.13	2.3				
0.08	1.4	<b>NEWLY COMMISSIONED MACHINERY</b>			
0.04	0.7				
0.00	0.0	<b>NEWLY COMMISSIONED MACHINERY</b>			
Foundation		Rigid	Flexible	Rigid	Flexible

Figure 18. ISO Velocity Chart from <http://www.stiweb.com/TechNote117.html>

The above figure is an interpretation of the ISO standard for vibration analysis, used in industry as a guideline by condition monitoring analysts (STI Vibration Monitoring, Inc., 2009). Physical faults may develop before a machine’s vibration exceeds the above tolerance levels; this

dissertation develops a method for analyzing stochastic data and applies it to fault detection in rotating machinery vibration.

Finally, there are other methods that use Bayesian methods to analyze damage in structures for the purpose of preventative maintenance and failure prevention. (Sohn, Law, & Kincho, 1998) formulate a method using a system stiffness matrix  $\mathbf{K}$  as an assembly of sub-structure stiffness matrices. The approach analyzes the Bayesian hypothesis test  $H_j$  for a damage fault at node  $j$  of the stiffness matrix  $\mathbf{K}$  based on a modal dataset  $\psi$ . (Zhang, Sconyers, Byington, Patrick, Orchard, & Vachtsevanos, 2008) focuses on CBM to reduce the time required to diagnose faults in rolling element bearings using an adaptive filtering method. In Zhang et al.'s approach, spectral peaks above background noise are isolated to highlight periodic disturbances. The side-bands of these frequency spectra (after adaptive noise reduction) are used to identify the presence of anomalous signals and, therefore, damaged parts. The feature used for anomaly detection is the sum of all frequency components within a weighted frequency band – this assumes that anomalous behavior presents with a known signal and known frequency characteristics. A key advantage of methods that rely on frequency analysis is their ability to operate in noisy systems – known frequency characteristics allow predictable noise to be filtered out.

### **3 ANOMALY DETECTION IN TIME-SERIES DATA**

This chapter details a method for anomaly detection in time-series state dependent data. In this case, it is assumed that the data can be adequately described using a finite number of discrete states using dimensionality reduction. By analyzing the time series transitions between these states, a probabilistic approach for system behavior is developed. This approach incorporates a hierarchical framework to model the transition probabilities themselves as random variables, and parameterizes the “normalcy” of the transition probability matrices. In this chapter, various synthetic data types are analyzed as well as data from experimental rolling element bearing systems. This experimental study demonstrates the application of the developed methods to real-world systems. This chapter primarily deals with systems where data are adequately sampled to populate the transition matrices, but also explores cases where the data are insufficient to populate the matrix, or the matrix itself is sparse.

#### **3.1 Introduction**

Rolling contact bearings are common in most modern rotating machinery, and are critical to the reliability of machine operation and lifespan. Bearing performance can be negatively affected by surface imperfections which cause anomalous signals to emerge as a fault progresses. These faults can ultimately lead to catastrophic system failure (Lacey, 2007). Early detection of anomalies can prevent such failure, reducing machine down time, repair cost, and the risk of physical injury to workers (Bin, 2008) (Lacey, 2008) (Sadettin, 2005). Monitoring vibration signals is a standard method used to infer information about bearing health state, and is a common part of many maintenance regimens (Lacey, 2008). Most fault detection/prediction methods developed to date are highly system specific. In this chapter, an attempt is made to

develop bearing-related anomaly detection approaches that are applicable across a wide spectrum of systems that incorporate bearings as critical components.

Anomaly detection is more difficult as the Signal to Noise Ratio (SNR) becomes very low. Fault detection/prediction becomes further complicated in high-dimensionality data, and in data which exhibit an inherently high degree of variability (Varun, 2009) (Asok, 2006) (Ericsson, 2003) (Eftekharnjad, Carrasco, Charnley, & Mba, 2011). In this chapter, a novel approach to anomaly detection in noisy signals is presented to isolate anomalous events in noisy, sparse, and experimental rolling element vibration data. The chapter is structured as follows:

1. The data are preprocessed with a combined wavelet transform and symbolic dimensionality reduction approach to reduce the effect of noise.
2. A novel prediction/detection approach is developed using hierarchically parameterized Transition Matrices to identify anomalies in both synthetic and experimental data.
3. This approach is compared to alternate methods and is shown to outperform the symbolic Nearest Neighbors search algorithm in both accuracy and speed.
4. Findings are summarized and future applications are proposed.

## **3.2 Background**

### **3.2.1 Discrete Wavelet Transform (DWT)**

A common step in vibration data preprocessing is to transform signal data into the wavelet domain (Mallat, 2008). Performing a first level Discrete Wavelet Transform (DWT) on raw data before symbolic approximation reduces the presence of noise typically associated with high dimensionality systems (Khatkhate, Gupta, Ray, & Patankar, 2008). Approximate and detailed wavelet coefficients are found at user-specified time shifts  $k$  to model the signal by passing the data through a high pass and low pass filter simultaneously. This is used instead of

the Short Time Fourier Transform (STFT) to overcome time-frequency resolution problems (Tzanetakis, 2001). The DWT can be mathematically represented by Equation (42):

$$W(j, k) = \sum_{j=1}^n \sum_{k=1}^n [f(k) \cdot 2^{-j/2} \psi(2^{-j} n - k)] \quad (42)$$

where  $\psi$  is the mother wavelet function. Decomposition of the discrete signal  $f[n]$  through the two filters can be expressed by Equations (43) and (44):

$$y_{high}[k] = \sum_n (f[n] \cdot g[2k - n]) \quad (43)$$

$$y_{low}[k] = \sum_n (f[n] \cdot h[2k - n]) \quad (44)$$

where  $y_{high}$  and  $y_{low}$  are the outputs of the high-pass and low-pass filters  $g$  and  $h$  respectively, and  $n$  is the discrete time step. Having reduced noise, this wavelet-transformed data is now suitable for dimensionality reduction described in the next section. In this work, the Daubechies family of wavelets was selected for use because of their continuous nature and overlapping windows, making them appropriate for processing frequency-dependent data such as rolling element signals.

### 3.2.2 Symbolic Aggregate Approximation (SAX)

This section briefly describes the fundamental concepts behind Symbolic Aggregate Approximation (SAX) (Keogh, Lin, & Fu, 2005). While recently published literature routinely implements SAX for dimensionality reduction in time series data related to the medical, surveillance, and industrial monitoring fields (Chiu, Lonardi, & Keough, 2003) (Sant Anna & Wickstrom, 2011) (Kasetty, 2008), the details of the reduction algorithm can be found in full in (Lin, 2003). SAX converts a time series sequence into a string of text, by partitioning windowed segments of data into equiprobable divisions of a Gaussian distribution using the relative distance from the signal mean. An assigned symbol from an alphabet of size  $N$  is then used to

label that segment of data. Since SAX assumes a normal distribution, the Z score of a data point is determined by Equation (45).

$$Z = \frac{y_i - \bar{y}}{\sigma} \quad (45)$$

The data are assigned to equiprobable divisions spaced by index values  $\gamma_{1,2\dots k}$  as in Equation (46).

$$\int_{\gamma_m}^{\gamma_{m+1}} f(y) \equiv \int_{\gamma_n}^{\gamma_{n+1}} f(y) = \int_{y=\gamma_m}^{y=\gamma_{m+1}} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(y-\bar{y})^2} \quad (46)$$

Figure 19 shows an example symbolic approximation of vibration data in the wavelet domain for a bearing under zero load for one revolution. For visual purposes, the SAX window size is 0.01 sec and the signal is separated into an alphabet size of  $N=6$ . The resulting string for this sequence would be “DBECD”.

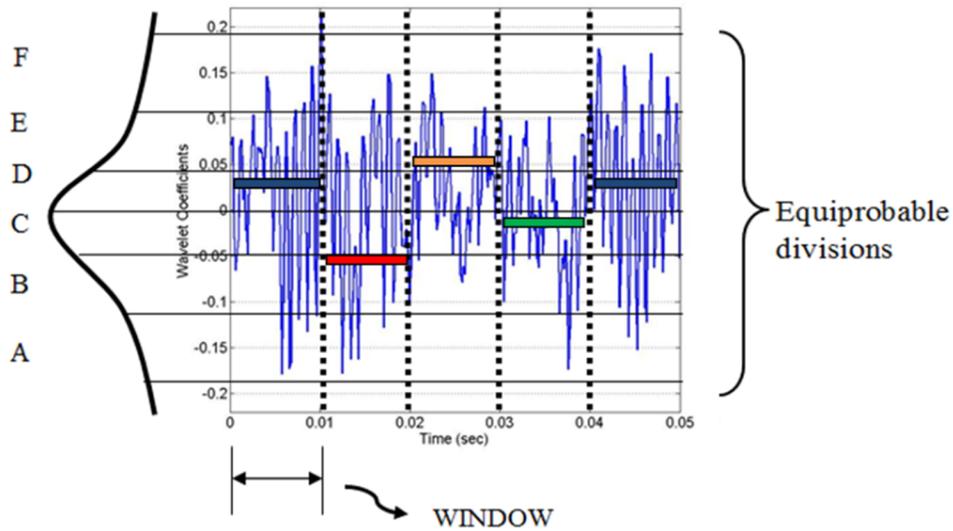


Figure 19. Representation of symbolic bin assignment in SAX

The main advantages that a symbolic approach such as SAX offers to processing data from rolling element bearings are:

- Compatibility with low resolution signals (Ray, 2004)

- Robustness against high frequency noise (Ray, 2004).
- Real-time processing on commercially available and inexpensive platforms (Gupda, Ray, & Keller, 2007)
- Improved computational efficiency in high dimensionality data, such as mechanical signals (Asok, 2006), (Lin, 2003).

Recent literature has found SAX to be comparable to or better than other dimensionality reduction methods such as DWT or Discrete Fourier Transform (DFT) (Buhl & Kennel, 2005) (Keogh E. , May, 2001) (Yi & Faloustos, 2000) (Chin, Ray, & Rajagopalan, 2004). This reduced-dimensionality data occupies a discrete set of possible system states, leading to a state-dependent analysis method detailed below.

### 3.2.3 Hidden Markov Models (HMMs) and Transition Matrices (TMs)

A state-dependent system with  $N$  discrete states can be represented by a Markov Chain if the likelihood of observing subsequent states is based on the likelihood of observation of previous states (Duda, Hart, & Stork, 2001). These transitions between system states form a discrete set of transition probabilities  $T$  of order  $N^2$ . An example transition probability between two states  $A$  and  $B$  is computed using Bayes' Theorem, shown in Equation (47).

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad (47)$$

The generation of this Markov Chain is predicated on observation of the data, or sufficient *a priori* knowledge of the data's conditional dependencies. An example Markov Chain is shown below in Figure 20. In this simplified example, the system only occupies three possible states.

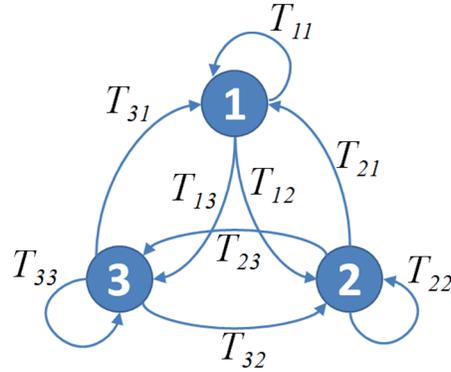


Figure 20. Markov Chain with transition probabilities for a three-state system

With sufficient data, it is possible to generate a complete stochastic Markov Chain, such that all transition probabilities are nonzero (i.e., all transitions have been observed to occur at least once in the dataset). These transition probabilities can then be organized into an  $N$  by  $N$  matrix, often referred to as a Transition Matrix (TM) or Stochastic Matrix. The generation of the TM is fairly straightforward, accomplished by tallying the total number of each unique state transition in the data. This matrix should then be normalized to a Right (or Left) matrix such that the sum of each row (or column, depending on orientation) sums to 1. In other words, the total probability that *some* state occurs after any given state  $m$  must always be 1. This normalization coefficient is derived by Equation (48):

$$T_{m,n} = \frac{\hat{T}_{m,n}}{\sum_n \hat{T}_{m,n}} \quad (48)$$

such that for any row  $m$ ,  $\sum_n T_{mn} \equiv 1$ . This approach results in the  $N$  by  $N$  Transition Matrix, shown below in Figure 21.

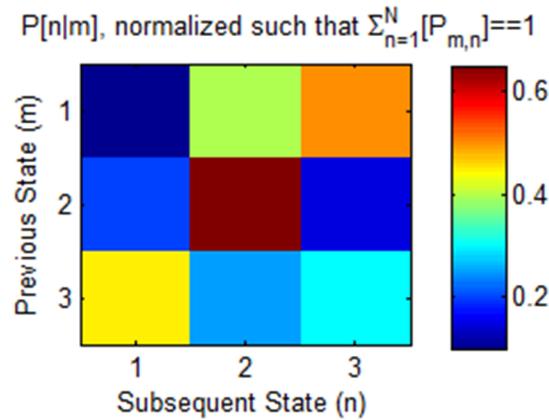


Figure 21. Example Transition Matrix for a 3-state System

Previous work has attempted to classify these TM's using the distance from nearest neighbor as an anomalousness metric (Khatkhate, Gupta, Ray, & Patankar, 2008). Instead, this chapter demonstrates the advantages of using a parameterized hierarchical model for Transition Matrices for robust, rapid, and accurate anomaly detection in bearing vibration signals.

### 3.2.4 Markov Chain Monte Carlo (MCMC) data generation

Often, transition probabilities are used to generate data for Bayesian Analysis in methods such as Markov Chain Monte Carlo (MCMC) (Gill, 2009). In this way, synthetic data can be generated to match the stochastic data, so that the conditional probabilities between states are nearly identical in each dataset's global TM. This is done by sampling the observed data's TM in a weighted random walk, drawing samples from the state-dependent histogram and generating sequential data points. Pseudo-code for such a sampling process is given in Table 4.

Table 4. Pseudocode for Markov Chain Monte Carlo data generation algorithm

```

T = (Transition Matrix),          N = (Number of States)
m = (System State to Start)  R = (Length of Generated Data)
for r = 1 to R
    breakPoints(1,1) = 0
    breakPoints(2,1) = T(m,1)
    for k=2 to N
        breakPoints(1,k) = breakPoints(2,k-1)
        breakPoints(2,k) = breakPoints(1,k) + T(m,k)
    randomDraw = RANDNUM  $\epsilon$  (0,1)
    for j = 1 to N
        if breakPoints(1,j) < randomDraw <= breakPoints(2,j)
            break
    nextState = j
    m = j

```

In this example, the matrix *breakPoints* is simply a calculation of the discrete cumulative probability function for a prior state *m*, and the variable *nextState* is returned to the calling function. This new state becomes the basis for the subsequent transition, and so on, until the dataset of size *R* has been generated. The created dataset can now act as a control set, enabling ground-truth analysis of the performance of various detection methods.

### 3.3 Overview of Analytical Approach

#### 3.3.1 Hierarchical Parameterization

The theory driving hierarchical parameterization has been described in further detail in Section 1.2.10. For this approach, the hierarchical parameterization of an anomaly detection method for time-series data is as follows:

Level 1: Time series data reduced into discrete states

Level 2: Markov probability matrices, which determine state changes with time

Level 3: Distributions of TM elements, an element's difference from expected value

Level 4: Distribution of difference metrics themselves for Anomaly Detection

A visual representation of this hierarchical model is given below in Figure 22.

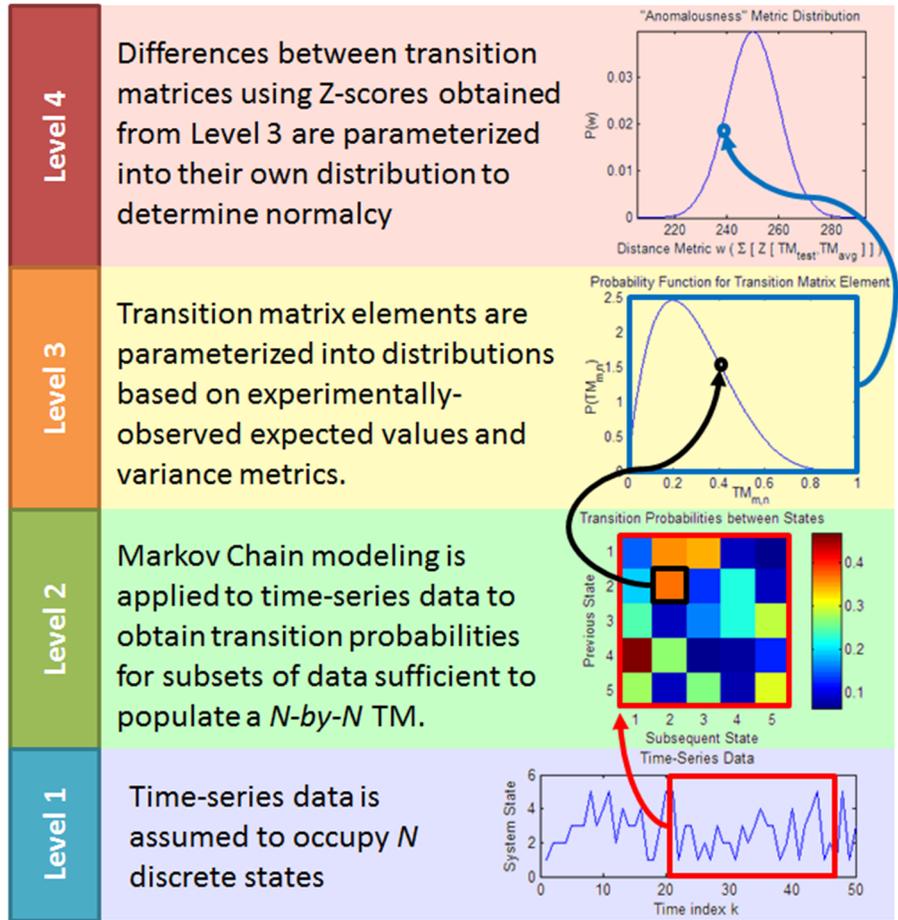


Figure 22. Hierarchical Parameterization Structure for Anomaly Detection

The fourth level of this hierarchy is a one-dimensional distribution to which confidence intervals are applied to determine the presence of anomalies. The detection of anomalies proceeds in the following way:

1. A subset of data generates a transition matrix.
2. This transition matrix is distinct relative to previously observed TMs.
3. By measuring this difference using the TM element distributions, the TM is Z-scored.
4. This Z-score is compared to all previously observed Z-scores.

If the Z-score for the candidate TM is sufficiently different from the learned database of Z-scores for TMs obtained from “normal” data, then the candidate TM (and therefore the data

that generated it) is classified as “anomalous”. A formal derivation of this process is given in greater detail in the following sections.

Previous methods have used hierarchical modeling for Transition Matrices by scaling the definition of the TM as a whole: comparing the entire matrix at a time, then by halves, then quadrants, and so on until the maximum resolution possible (Patil & Taillie, 2001). In contrast, the method presented in this chapter involves the parameterization of the TM elements themselves, introducing a hierarchical model for each transition probability in the Markov Chain. By windowing the data into smaller subsets (which need not be sufficiently larger than  $N^2$ ), sparsely populated transition matrices  $T^{(k)}$  are calculated, shown below in Figure 23.

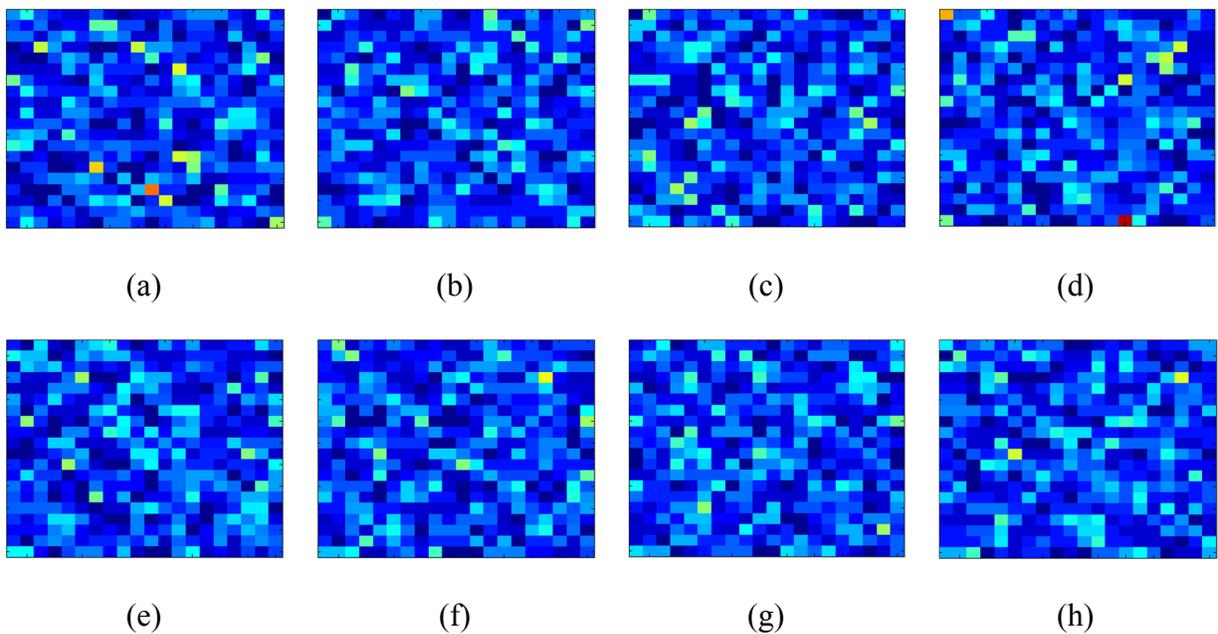


Figure 23 (a)-(h). Several examples of sparse Transition Matrices from normal data

The plots (a) through (h) of Figure 23 are example Transition Matrices calculated from small windows ( $O \approx N^2$ ) of time series data. The color intensity represents the probability of transitioning from a prior state (y-axes) to a successive state (x-axes), as each row is normalized to a total probability of 1.

### 3.3.2 Block diagram of approach

This section outlines the data flow for the work performed on both the synthetic and experimental sources. The HTMM approach developed in this chapter is a Machine Learning algorithm, with separate processes for training and testing. Figure 24 shows the training portion of this algorithm, and Figure 25 shows the testing portion. The full derivation of this analysis method is given in detail in Section 3.6.

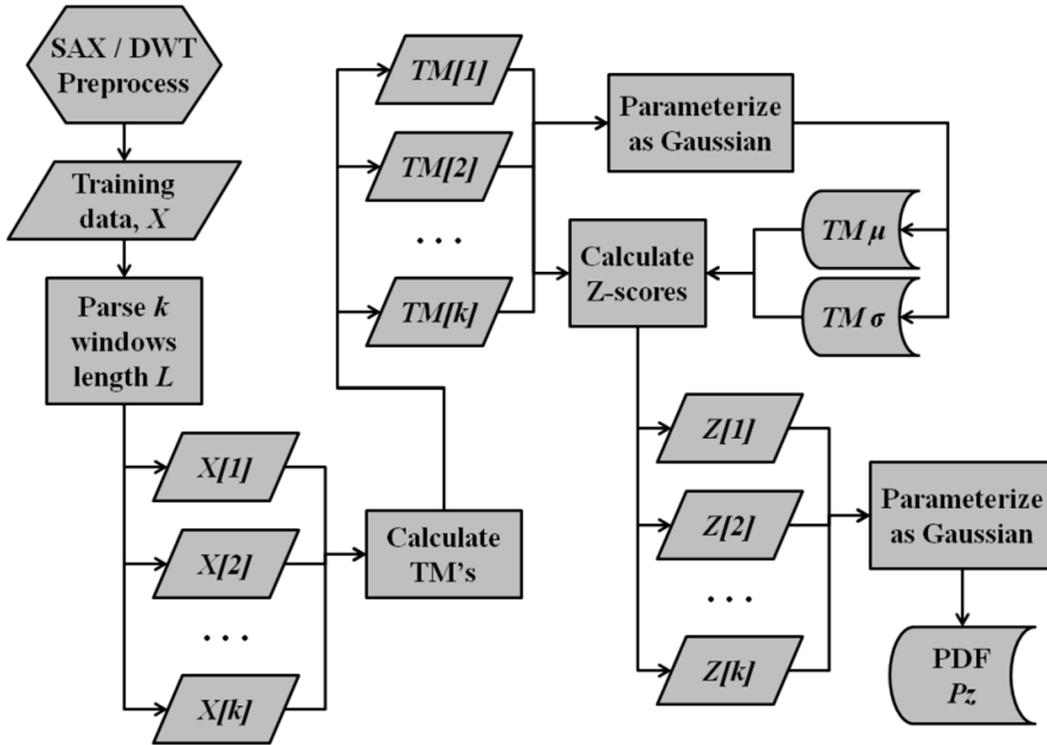


Figure 24. Training Algorithm Flowchart

Training method:

1. Perform *SAX / DWT* preprocessing on raw data
2. Load time-series reduced-dimensionality training data  $X$
3. Window training data into  $k$  sections  $X[k]$  of length  $L$
4. Obtain the Transition Matrix  $P[k]$  for each  $X[k]$
5. Parameterize  $P[1:k]$  into a Gaussian distribution with parameters  $TM \mu$  and  $TM \sigma$

6. Obtain total Z-scores  $Z[k]$  for each  $P[k]$  relative to  $TM \mu$  and  $TM \sigma$
7. Parameterize  $Z[k]$  into a Gaussian distribution  $P_z$

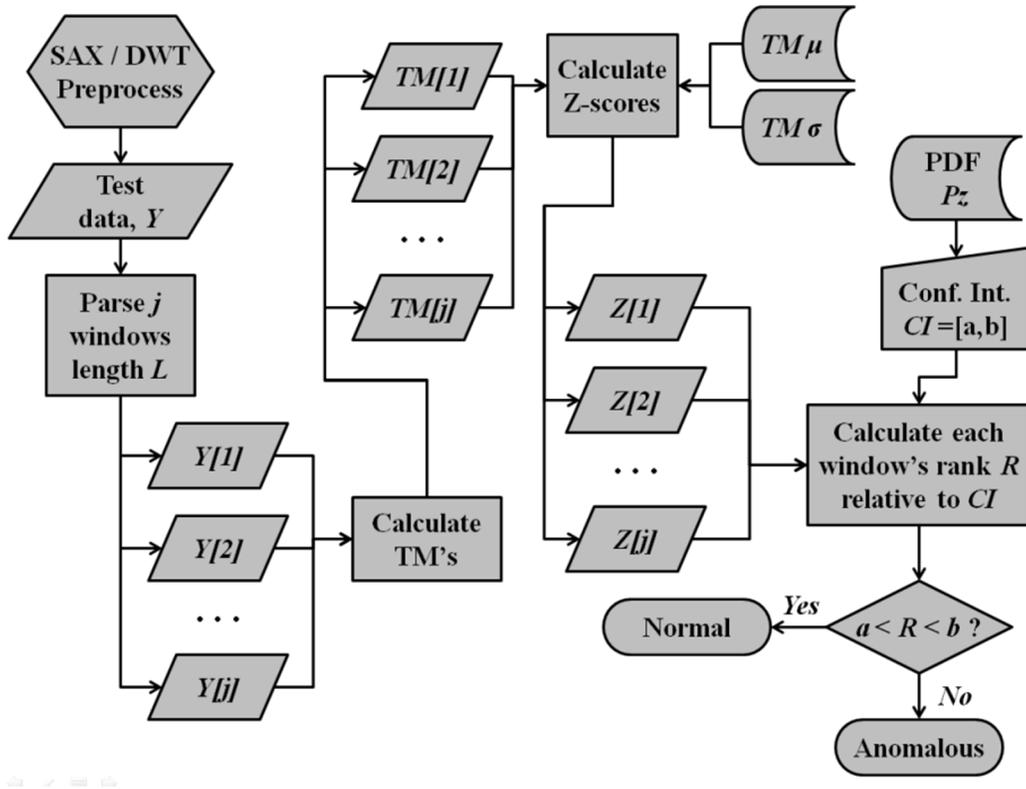


Figure 25. Testing Algorithm Flowchart

Testing method:

1. Perform *SAX / DWT* preprocessing on raw data
2. Load time-series reduced-dimensionality test data  $Y$
3. Window training data into  $j$  sections  $Y[j]$  of length  $L$
4. Obtain the Transition Matrix  $P[j]$  for each  $Y[j]$
5. Obtain total Z-scores  $Z[j]$  for each  $P[j]$  relative to  $TM \mu$  and  $TM \sigma$
6. Specify a Confidence Interval  $CI$  with bounds  $[a, b]$
7. Rank the test Z-scores  $Z[j]$  relative to the  $CI$  obtained from  $P_z$
8. If the Z-score falls outside the  $CI$ , flag the window  $Y[j]$  as anomalous

### 3.4 Data Types Obtained and Generated

#### 3.4.1 Data Generated by ARMA

A test dataset was generated using an autoregressive moving average model (ARMA) to validate the HTMM algorithm's application to stochastic time-series data. This dataset was generated using an ARMA model with parameters shown in Table 5.

Table 5. Parameters for ARMA test model

Normal Data Model		Anomalous Data Model	
c	0	c	0
$\Phi$	[0.100, 0.100]	$\Phi$	[-0.100, 0.100]
$\varepsilon$	1.000	$\varepsilon$	1.000
$\Psi$	[0.100, 0.100]	$\Psi$	[-0.100, 0.100]

These parameters cause a difference in the ARMA model in the first term of both AR and MA polynomials. A test dataset of  $10^6$  data points was generated using both the normal and anomalous models, and tested using HTMM. This time-series data was converted to 20 discrete states using SAX (Keogh, Lin, & Fu, 2005). A sample of this test data is shown in Figure 26.

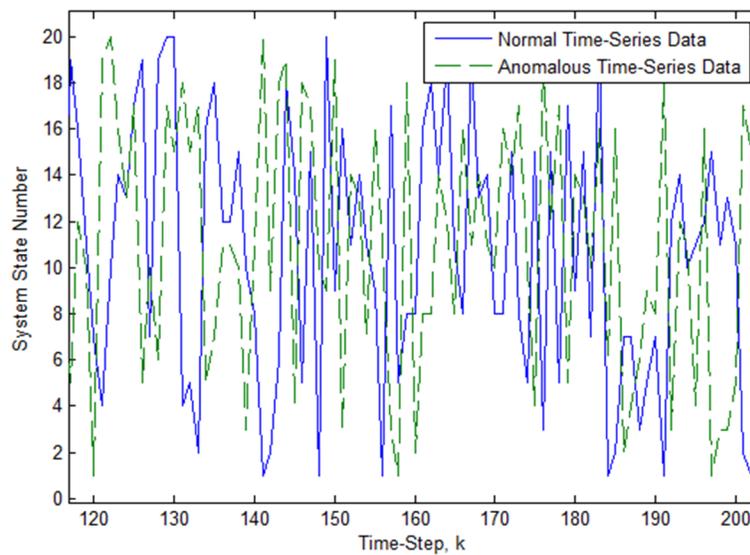


Figure 26. Time-series data generated by ARMA, normal and anomalous models

Also shown below in Figure 27, the power spectral density (PSD) estimates of both time-series datasets are compared. It is clear from this figure that the normal and anomalous data differ in the frequency domain, but this distinction requires significant data for testing.

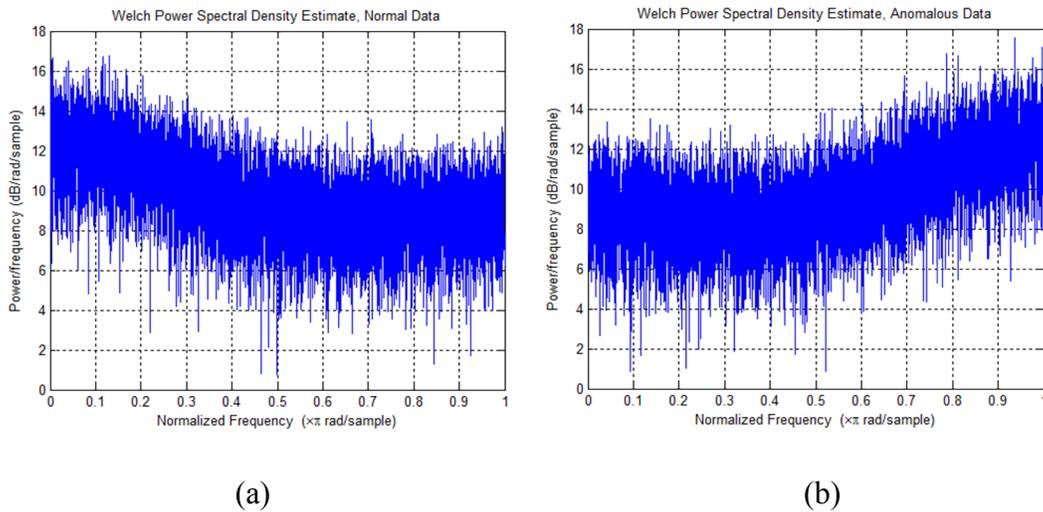


Figure 27. Power spectral density of normal (a) vs. anomalous (b) data

### 3.4.2 Synthetic Data from MCMC

Three datasets were generated using Markov Chain Monte Carlo (MCMC) with three distinct pairs of normal and anomalous transition matrices. The first of these transition matrices was diagonal and intentionally sparse, shown in Figure 28. Another TM set was generated randomly, using the Matlab function `randn`, restricted to positive values and row-normalized, shown in Figure 29. The third TM set was generated as a bistochastic matrix; all rows and columns must sum to 1. This final TM shown in Figure 30 represents data where each state has an equal probability of occurrence, making frequency domain conversion and traditional statistics invalid for analysis.

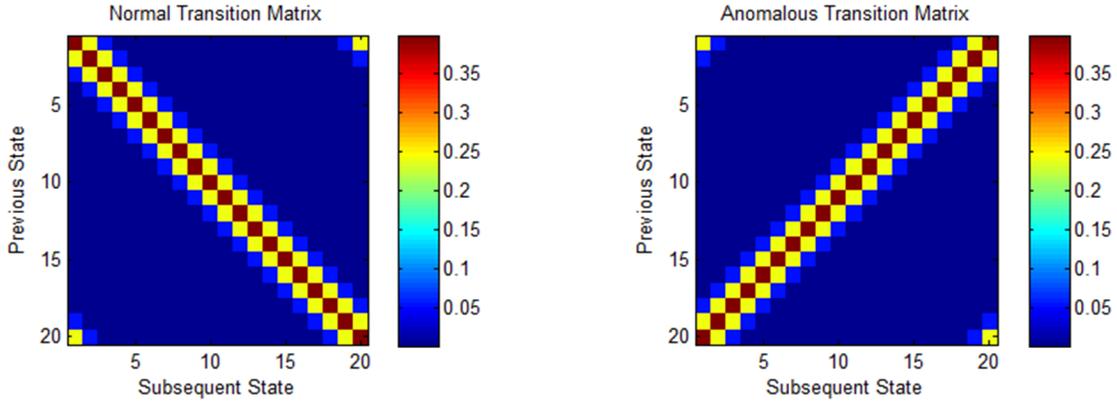


Figure 28. Normal and anomalous transition matrices, sparse diagonal case.

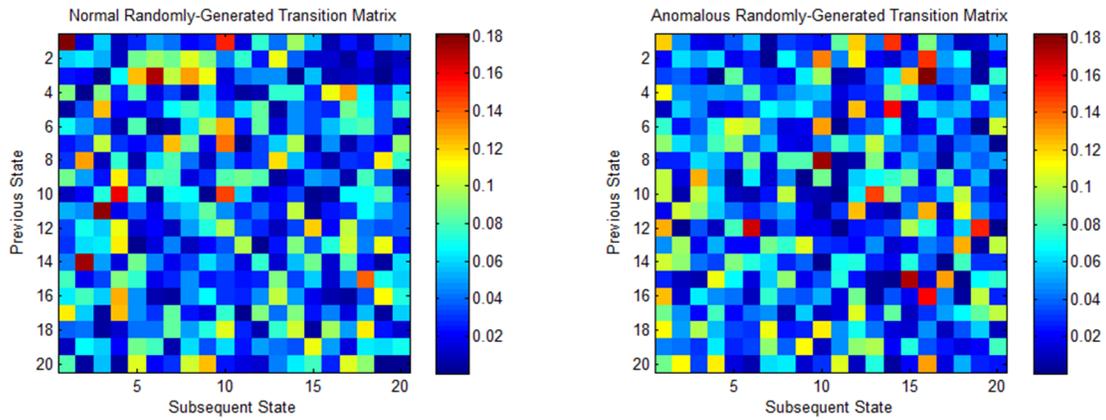


Figure 29. Normal and anomalous transition matrices, purely random case.

Traditional analysis methods such as frequency analysis or simple statistical histogram analysis are not well suited to extract information from high-noise data. An even less favorable case would be if the data were not corrupted by, but indistinguishable from, white noise (using frequency analysis and first-order statistics). Such truly random state-dependent data should fit the criterion in Equation (49).

$$P(a) = P(b) = P(c) \dots = P(n) \quad (49)$$

However, this equiprobable quality is also observed in nonrandom deterministic data, in cases where the total probability of each state is equal, but the probability of state transitions is

not. A bistochastic (or *doubly stochastic*) matrix is one such case, and will be used to demonstrate that the methods developed in this chapter are robust for low SNR data.

Algorithm 1 of (Cappellini, 2009) details an iterative method for generating a nonrandom bistochastic matrix from any nonnegative real valued Transition Matrix,  $T$ . By iteratively normalizing each row and column to unity, this process will converge on unity row and column sums across the entire matrix. This algorithm, reproduced with minor changes below, is valid for any stochastic input matrix  $T$  with nonnegative elements and is expected to converge quickly (Cappellini, 2009). Given the input  $N$  by  $N$  matrix,  $T$ :

1. Normalize each row vector of  $T$  by dividing it by the sum of its elements
2. Normalize each column vector of  $T$  by dividing it by the sum of its elements
3. Stop if the matrix  $T$  is bistochastic up to a user-specified precision

The metric chosen for determining the precision of this bistochastic matrix is the Summation of Absolute Differences (SAD) for each row and column, shown below in Equation (50):

$$\varepsilon = \sum_{n=1}^N ARG\_ABS\langle T_{m,n} - 1 \rangle + \sum_{m=1}^N ARG\_ABS\langle T_{m,n} - 1 \rangle \quad (50)$$

The error term,  $\varepsilon$ , is specified by the user to apply a desired degree of accuracy. For reliability, this threshold was chosen to be  $\varepsilon < 0.01 * N^2$ , so that the mean bistochastic error for any given matrix element would be less than one percent. Additionally, when generating this matrix, it was desirable to choose a nonrandom structure for the state transition probabilities. Separate and unique bistochastic Transition Matrices were used to generate both normal and anomalous data. The structure of normal data is governed by Equation (51).

$$T_{m,n} = \frac{\sin(m) + \sin(n)}{2} + 0.75 + RAND \in [0,0.1] \quad (51)$$

Anomalous data were generated from a similar but distinct schema, governed by Equation (52).

$$T_{m,n} = \frac{\cos(m) + \cos(n)}{2} + 0.75 + RAND \in [0,0.1] \quad (52)$$

The Transition Matrices for these schemas are shown below in Figure 30

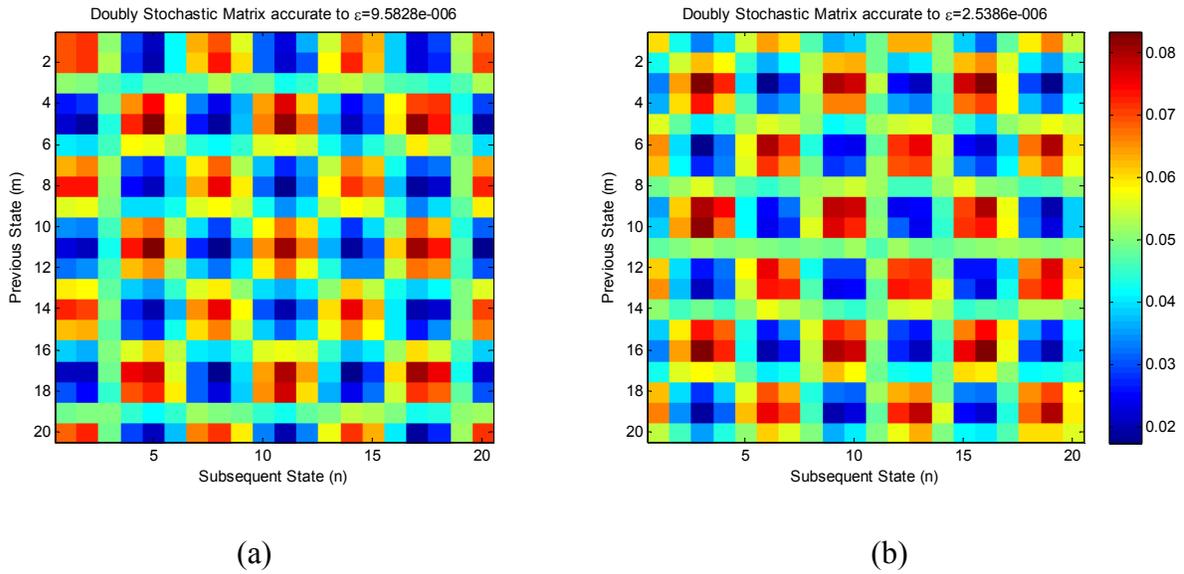
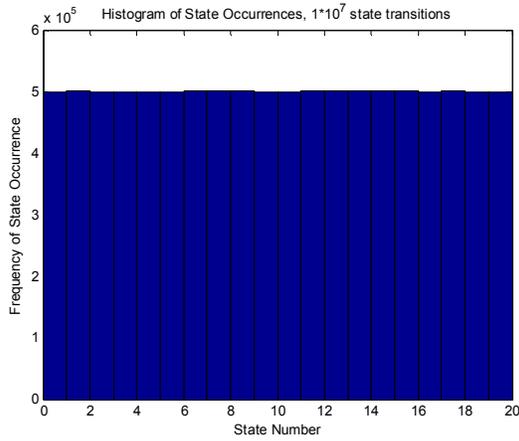
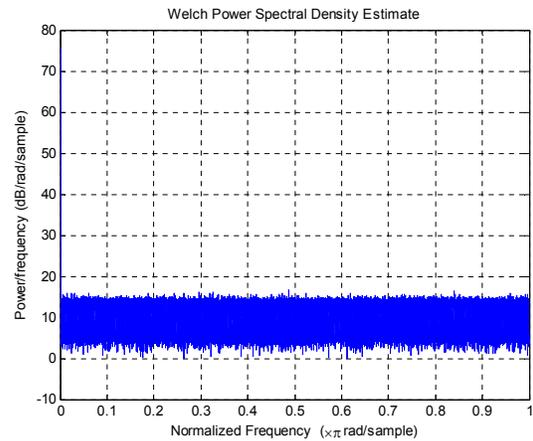


Figure 30. Bistochastic Matrix of Normal Data (a) and Anomalous Data (b)

Normal time-series data were generated from the bistochastic TM in Figure 30 (a) using the method outlined in Section 1.2.6, and first-order statistics and frequency domain characteristics were calculated. The mean and standard deviation of this data were found to be 10.500 and 5.769, respectively. A histogram of the normal training data and power spectral density are shown below in Figure 31.



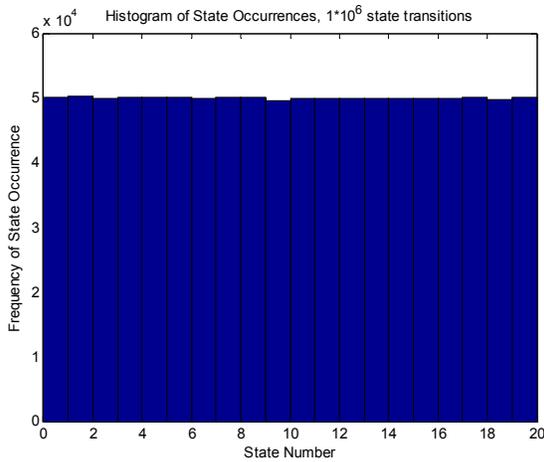
(a)



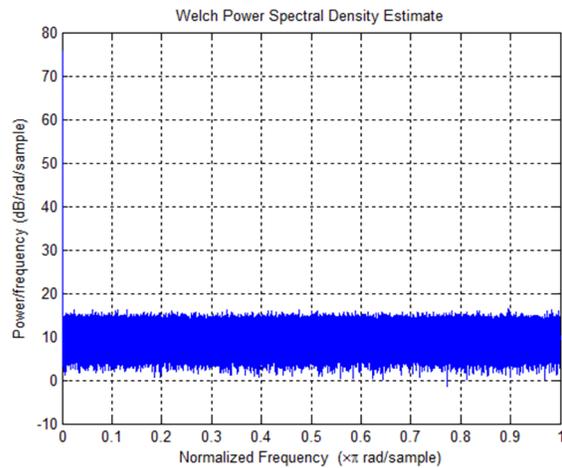
(b)

Figure 31. Histogram (a) and Frequency Analysis (b) of Normal Bistochastic Data

Similar graphs were generated from anomalous test data, which were in turn generated from Figure 30 (b) above. To demonstrate that the test data are functionally indistinguishable from the training data using previous metrics, these graphs are shown below in Figure 32.



(a)



(b)

Figure 32. Histogram (a) and Frequency Analysis (b) of Anomalous Bistochastic Data

It is clear that certain data types are not well-suited for processing with methods such as frequency domain analysis, basic statistical analysis, and Euclidian Nearest Neighbor / Markov distance analysis. Organizing the Transition Matrices in a parameterized hierarchical model

provides an avenue for detection/prediction in normal and anomalous data that are statistically indistinguishable using previous approaches.

### **3.4.3 Experimental Data**

Svenska Kullagerfabriken (SKF) ball bearing test data were used from experiments conducted by Case Western Reserve University on a 2 hp (1.49 kW) Reliance brand electric motor. Test conditions of the motor (load and angular velocity) were monitored in conjunction with the bearing accelerometer data. Single point bearing faults having diameters of 0.007” and 0.021” (0.1778 mm and 0.5334 mm) were manually developed on the outer raceway using an Electro-Discharge Machining (EDM) process. Accelerometer data was recorded under loads of 0, 1, 2, and 3 hp (0, 0.745, 1.49, 2.23 kW). Both normal and anomalous data were sampled from the opposite side of the seeded fault (at 12:00) on the outer raceway, at 12000 samples/sec (12 kHz). Data files and further descriptions of the setup can be found at:

<http://cseggroups.case.edu/bearingdatacenter/pages/welcome-case-western-reserve-university-bearing-data-center-website>

## **3.5 Comparison and other processing techniques**

### **3.5.1 Discrete Wavelet Transform & Dimensionality Reduction**

A Discrete Wavelet Transform (DWT) was performed using Matlab’s DB1 function on both the experimental and synthetic data in order to reduce the impact of high frequency noise (Asok, 2006). To approximate the resulting wavelet-domain signal into a symbolic sequence, SAX is applied at a window length of 1 point. For a performance comparison, a window of 10 data points was also applied. An alphabet size of  $N=20$  was selected for all processing. This alphabet length is significantly larger than that of previous work (Khatkhate, Ray, Keller, Gupta, & Chin, 2006), (Xin, 2011). The larger alphabet size incorporates added computational

complexity; it is demonstrated in section 3.8 that the increased alphabet length exponentially increases processing time using other anomaly detection approaches.

### 3.5.2 Comparison to Nearest Neighbor Analysis

The Nearest Neighbors (NN) approach to anomaly detection is a common method for identifying portions of data which do not heuristically match a training set. This is often done using a simple Euclidian distance comparison between two vectors of equal length. While this approach is valid for continuous data, it is computationally easier to perform on reduced dimensionality data (such as the data presented in this chapter) having a discrete number of system states. The Euclidian distance  $E$  between two vectors  $U$  and  $V$  is given by Equation (53).

$$E = \sum_{k=1}^K \sqrt{(U_k - V_k)^2} \quad (53)$$

This approach yields a result equal to the Sum of Absolute Differences (SAD) method. In the Nearest Neighbors approach, a system is trained by obtaining these distance measures  $E$  for each subset of training data against the rest of the training set. The resulting distance metrics can then be parameterized into an appropriate distribution, and used for the detection of anomalous data. A portion of data which is farthest from its nearest neighbor is considered more “anomalous”. However, as the size of the training set increases, this approach becomes less reliable; the probability of a similar or identical window of data existing within the training set grows as the training set itself grows. This is discussed further in section 3.8.2. In this work, the Nearest Neighbors method is applied to the Transition Matrices generated from the windowed data. Each training TM is compared to all other training TMs, and the NN distance is calculated. This creates a set of known “normal” distances from each training TM’s nearest neighbor. A test TM that is farthest from its nearest neighbor in the training set is considered more anomalous.

By parameterizing these results into a Gaussian distribution, confidence intervals are applied to compare normal and anomalous distributions, and performance analysis becomes possible.

### 3.5.3 Comparison to Raw Probability Calculation

For the method described in Section 2.1.1.1, to perform in sparse data, the training transition matrices must not be sparse. From (Ye, 2000):

*While using formula (4) to infer the probability of support to the sequence of states, the probabilities of zero would dominate the final probability result from formula (4) and make it zero, regardless of the number of nonzero elements in the computation using formula (4). In this study we assigned the small probability value of 0.00001 (or 1E-5 in the scientific expression) to the initial state and state transitions which did not appear in the training data, while using formula (4) to infer the probability of support to a sequence of states*

This modification to the transition matrices causes all TMs to be non-sparse, and assumes that  $1 \times 10^{-5}$  is an appropriate value for the probability in question. The method proposed in this chapter makes no such assumption – instead, the transition matrix elements themselves are parameterized. In cases where state transitions are never observed in the training data, HTMM's parameterization becomes a  $N(0,0)$  distribution; other methods do not parameterize these cases with statistical rigor. In the case of the diagonal TM detailed in Section 3.4.2, the prevalence of zero-probabilities was inherent in the data. In cases where training TMs were sparse and test windows were small ( $L < N$ ), the method in (Ye, 2000) had significantly higher false-positive rates than HTMM; for cases where TMs were non-sparse and well-populated, both methods obtained reliable detection rates. This would indicate a broader range of applications for HTMM, as it does not rely on assumptions about state transition probabilities.

### 3.6 Hierarchical Transition Matrix Model (HTMM)

As detailed in Section 3.2.3, a deterministic state-dependent system can be described using a Markov Model. By using smaller subsets of data (on the order of  $N^2$  samples), the same method of generating Transition Matrices shown in Section 3.2.4 will result in an incomplete set of transition probabilities. Even if the number of samples is insufficient to fully populate a TM, it is still possible to approach a true normal parameterized value for the transition probabilities. This occurs through the central limit theorem, which states that as the number of samples  $n$  approaches infinity, a set of random variables converges to a normal distribution  $N(\mu, \sigma^2)$  (Billingsley, 1995).

#### 3.6.1 Formulaic Derivation

In this method, time-series data occupy a discrete number of states  $N$ . The key assumptions are that the data must be:

- Ergodic (over time, the system's behavior is predictable)
- Causal (previous states affect the probability of observing future states)
- Stationary (the expected value and variance of the data are unchanging over time)

The novel approach described here treats each element of a transition matrix  $T$  as a random variable. For each random variable, z-scores for the  $K$  training transition matrix elements  $T_{m,n}$  are calculated using expected value and variance:

$$E[T_{m,n}] = \frac{1}{K} \sum_{k=1}^K T_{m,n}^{(k)} \quad (54)$$

$$V[T_{m,n}] = \frac{1}{K} \left( \sum_{k=1}^K T_{m,n}^{(k)2} - \frac{1}{K} \left( \sum_{k=1}^K T_{m,n}^{(k)} \right)^2 \right) \quad (55)$$

and the standard absolute Z-score calculation is calculated as:

$$Z_{T_{m,n}}^{(k)} = \frac{|T_{m,n}^{(k)} - E[T_{m,n}]|}{\sqrt{V[T_{m,n}]}} \quad (56)$$

For each transition matrix  $T^{(k)}$ , the above Z-scores are summed over the entire transition matrix into a total value to describe the TM's "normalcy". This measure of normalcy is given in Equation(57), and further hierarchically parameterized into a Gaussian distribution with parameters  $\mu_z$  and  $\sigma_z$  as in Equation (58).

$$Z^{(k)} = \sum_{m=1}^N \sum_{n=1}^N Z_{T_{m,n}}^{(k)} \quad (57)$$

$$Z^{(k)} \sim N(\mu_z, \sigma_z) \quad (58)$$

The appropriateness of the Gaussian model is discussed further in Section 3.6.2, and the Z-score distribution's (as a function of  $\mu_z$  and  $\sigma_z$ ) use in final anomaly detection is detailed in Section 3.6.3.

### 3.6.2 Appropriateness of Gaussian Model

Because the Gaussian distribution is used as the final step for testing anomalousness, it is necessary to demonstrate that this distribution is appropriate. Each transition matrix is compared to the mean TM, so the Z-scores from this comparison must be well represented by the Gaussian distribution. Through the central limit theorem and the law of large numbers, the expected value of all TM images converges to the true TM of the dataset. The Lindeberg-Levy Central Limit Theorem dictates that if a sequence of i.i.d. random variables follows some expected value and variance, as the number of samples grows larger, the sample mean random variable converges to a Gaussian distribution (Billingsley, 1995).

$$\{Y_1, Y_2, \dots\} \Rightarrow E[Y] = \mu, \text{Var}[Y] = \sigma^2 \quad (59)$$

$$\text{as } n \rightarrow \infty, \left( \left( \frac{1}{n} \sum_{i=1}^n Y_i \right) - \mu \right) \sim N(0, \sigma^2) \quad (60)$$

Because Z-scores are summed over the entire TM, each TM's overall Z-score statistic is well-represented by a Gaussian distribution through the above theorems. The absolute Z-score of a TM element must always be positive, which is often modeled by continuous positive distributions such as the Gamma distribution. However, the following simulation and goodness-of-fit tests show that the Gaussian distribution is an appropriate model for the TM Z-scores.

Stochastic data was generated from a 10x10 random transition matrix, then parsed into windows of 200 samples and used to populate training transition matrices. All TM Z-scores were calculated for a dataset containing  $10^3$ ,  $10^4$ , and  $10^5$  samples. Each set of training Z-scores was converted to a Cumulative Distribution Function (CDF), and compared to a CDF generated using the expected value and variance parameters of a true Gaussian distribution. The results are shown in Figure 33:

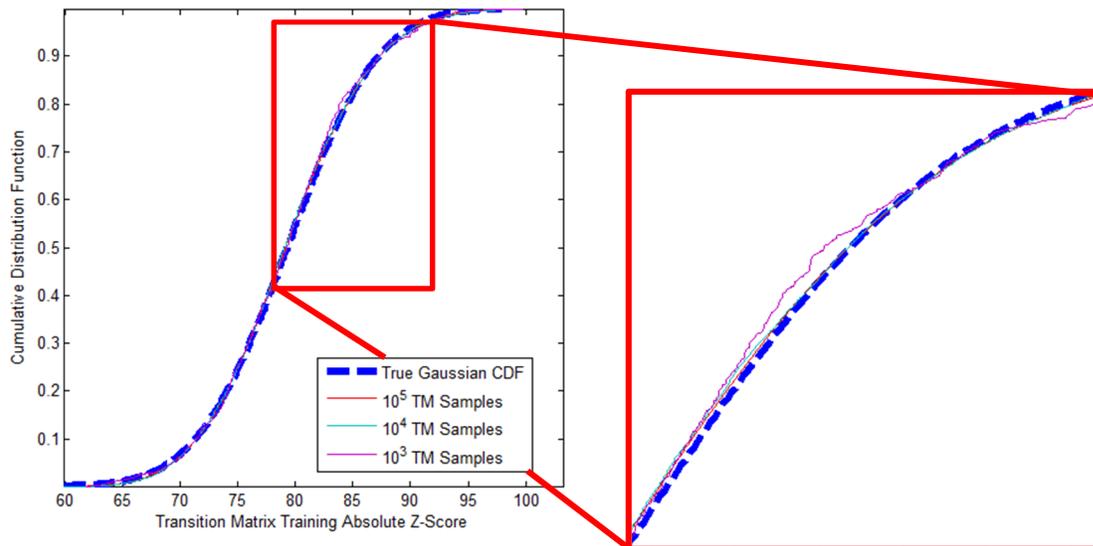


Figure 33. Cumulative Distribution Functions for TM Z-scores vs. Gaussian Model

The analysis is performed through the use of the Kolmogorov–Smirnov (K-S) test and the Cramér–von Mises (CVM) criterion. The Kolmogorov–Smirnov test either confirms or rejects the hypothesis that some data  $Y$  were generated from a continuous distribution  $F$ . For hypothesis testing, the K-S statistic is:

$$D_n = \sup_x |F_n(y) - F(y)| \quad (61)$$

where  $F_n$  is the empirical distribution for  $Y$ ,  $F$  is the continuous or expected distribution, and “sup” indicates the supremum (the least upper bound of the differences). If the calculated K-S statistic  $D_n$  meets some critical value  $D_{crit}$ , the null hypothesis is rejected (the data are assumed to follow the expected distribution). For empirical distributions with  $n > 35$  and a significance of  $\alpha = 0.05$ , the critical value for  $D$  is calculated as:

$$D_{crit} = \frac{1.358}{\sqrt{n}} \quad (62)$$

The largest difference between the experimental and true Gaussian distributions occurred in the  $n = 10^3$  training set, yielding a  $D_n$  value of 0.0337. For this set, a corresponding  $D_{crit}$  value is 0.0429. The null hypothesis is shown in Equation (63):

$$H_0 : D_n > D_{crit} \quad (63)$$

These results provide sufficient evidence to reject the null hypothesis  $H_0$ , and validate the assumption that the TM Z-score distribution is well-approximated by a continuous Gaussian model.

To further confirm the validity of the assumed Gaussian distribution, the Cramér–von Mises test statistic was calculated for the  $10^3$ ,  $10^4$ , and  $10^5$ -point datasets. This test statistic is calculated in Equation (64):

$$W = \int_{-\infty}^{\infty} [F_n(y) - F(y)]^2 dF(y) \quad (64)$$

Using critical  $W$  values from (Thode, 2002), test statistics for the CVM criterion confirm or reject the null hypothesis:

$$H_0 : W_n > W_{crit} \quad (65)$$

For all three training TM sets, the CVM statistic rejected the null hypothesis at statistical significances between  $\alpha=0.05$  and  $\alpha=0.01$ . This result confirms between 95% and 99% significance that the Gaussian distribution is an appropriate model for the Z-score anomaly detection metric.

### 3.6.3 Application of Confidence Intervals & Anomaly Detection

For the final stage of anomaly detection, the topmost level of the hierarchy, confidence intervals are used to classify candidate data as “normal” or “anomalous”. Using the Gaussian distribution, a user specified confidence interval (CI) is used to determine boundary coefficients  $a$  and  $b$  such that:

$$CI = \int_a^b \left( \frac{1}{\sqrt{2\pi\sigma_z^2}} e^{-\frac{1}{2\sigma_z^2}(y-\mu_z)^2} \right) dy \quad (66)$$

Due to the symmetric nature of the Gaussian distribution, the confidence interval boundaries  $a$  and  $b$  are further constrained by:

$$\mu - a = b - \mu \quad (67)$$

When a transition matrix is generated from test time-series data, it is compared against the expected value for a “normal” transition matrix based on previous training observations.

$$Z_{T_{test}} = \sum_{n=1}^N \sum_{m=1}^N \frac{|T_{test} - E[T_{train}]|}{\sqrt{V[T_{train}]}} \quad (68)$$

This TM Z-score is compared to all previously obtained TM Z-scores through the distribution in Equation (58) to generate one final metric for anomalousness. If this  $Z_{T[test]}$  value falls outside the boundaries for  $a$  and  $b$  obtained from the training data, the TM is declared anomalous:  $IF((Z_{T_{test}} \leq b) \text{ AND } (Z_{T_{test}} \geq a)), THEN(normal), ELSE(anomalous)$  An example histogram of these Z-scores is shown below in Figure 34.

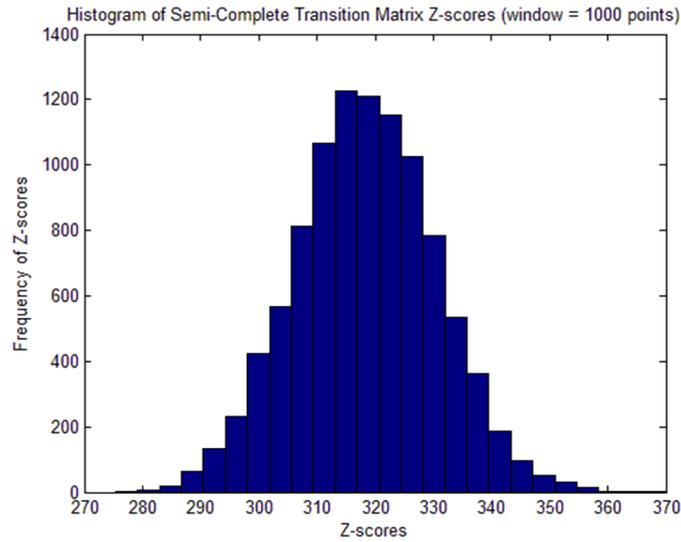


Figure 34. Histogram of Normal Data Transition Matrix Z-scores

When a histogram of TM Z-scores (with anomalies) is plotted against these confidence intervals, the anomalous data is immediately apparent. This visual example of automated anomaly detection is shown below in Figure 35.

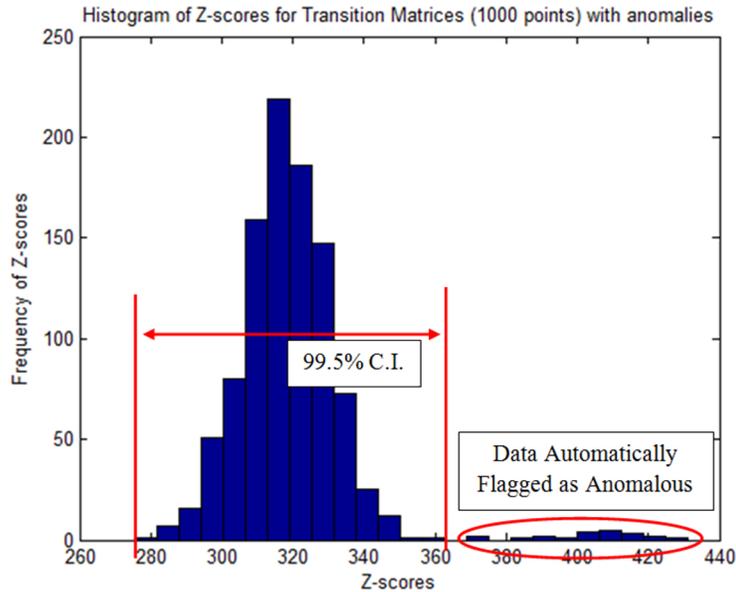


Figure 35. Example Z-Score distribution of example HTMM analysis

To further examine this detection measure’s performance, a variety of anomaly sizes and detection thresholds were applied. These results enabled the generation of Receiver Operating Characteristic (ROC) curves to rate ground-truth performance, described in detail in Section 3.7.

### 3.7 Results

#### 3.7.1 Detection results in ARMA-generated data

Performing robust anomaly detection typically requires large sample sizes (e.g., frequency domain modeling). Using HTMM, performance characteristics were generated for detection windows ranging from length  $L = N^2/6$  to  $L = 2N^2$ . For a discretization level  $N=20$ , the smallest of these window lengths is  $L = 80$  data points, falling well below typical sample sizes for accurate detection in other methods. The true positive and false positive rates (TPR and FPR) were combined into receiver operating characteristic (ROC) curves, shown in Figure 36.

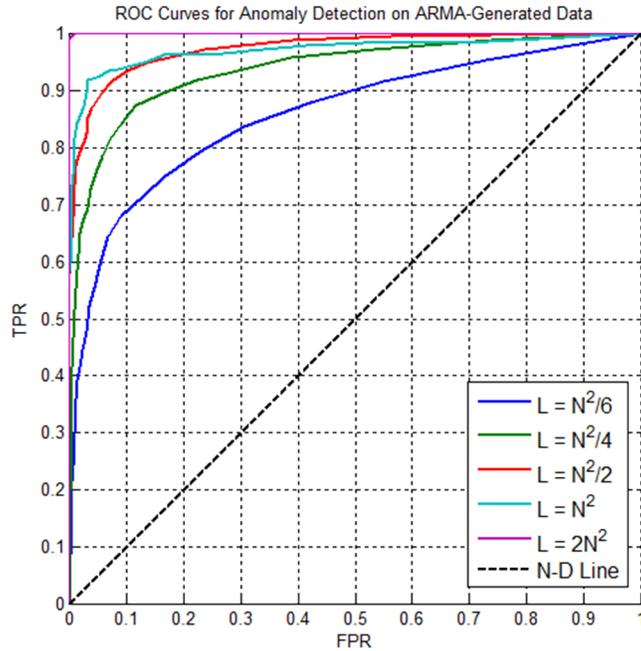


Figure 36. ROC Curves for HTMM applied to ARMA-generated data

Performance characteristics for this method were obtained for all test window sizes.

Testing was performed using MATLAB 2011b (64-bit), with Windows 7 (64-bit) running on an Intel Core2 Duo Processor (2.00 GHz) with 4 GB of RAM. The best-case TPR and FPR for each test window size was obtained, as well as the per-test processing time, shown in Table 6.

Table 6. Performance Characteristics for HTMM applied to ARMA-generated data.

	TPR (best-case)	FPR (best-case)	Test time (seconds per test window)
$L = N^2 / 6$	0.681	0.089	0.0064
$L = N^2 / 4$	0.842	0.089	0.0067
$L = N^2 / 2$	0.898	0.058	0.0075
$L = N^2$	0.921	0.032	0.0077
$L = N^2 * 2$	1.000	0.008	0.0090

These results demonstrate accurate anomaly detection (>92%) with very low (3%) false alarm rates for all window lengths above the  $N^2$  threshold (required to populate candidate TMs). For even sparsely populated TMs sampled from  $L \leq N^2/2$ , HTMM resulted in robust anomaly detection with high TPR:FPR ratios, and processing times suitable for real-time analysis of time-series stochastic data.

### **3.7.2 Detection results in MCMC-generated data**

To test this new HTMM analysis method, the ground-truth normal data were processed to find the Z-score values of each parameterized TM. This was accomplished with a variety of non-overlapping anomaly sizes with a fixed window of observation. That is, the tuning parameter for the performance metric was the length of anomalous data required for detection. The test set for this matrix occupied  $1 \times 10^7$  data points, with anomalous data of varying lengths inserted at known times. These anomalies were then adjusted to occupy shorter and shorter time-spans in the dataset, to test the limit of what HTMM can successfully detect.

For this study, a 10,000,000 point dataset was corrupted with 500 distinct spans of anomalous data in lengths of 1000, 900, 800, 700, 600, 500, 450, 400, 375, 350, 325 and 300 points. The True- and False-Positive ratios were obtained by adjusting the bounds of the Confidence Interval used in the detection algorithm. The ground-truth performance characteristics of the anomaly detection method were collected and analyzed, and compiled into ROC curves shown below in Figure 37.

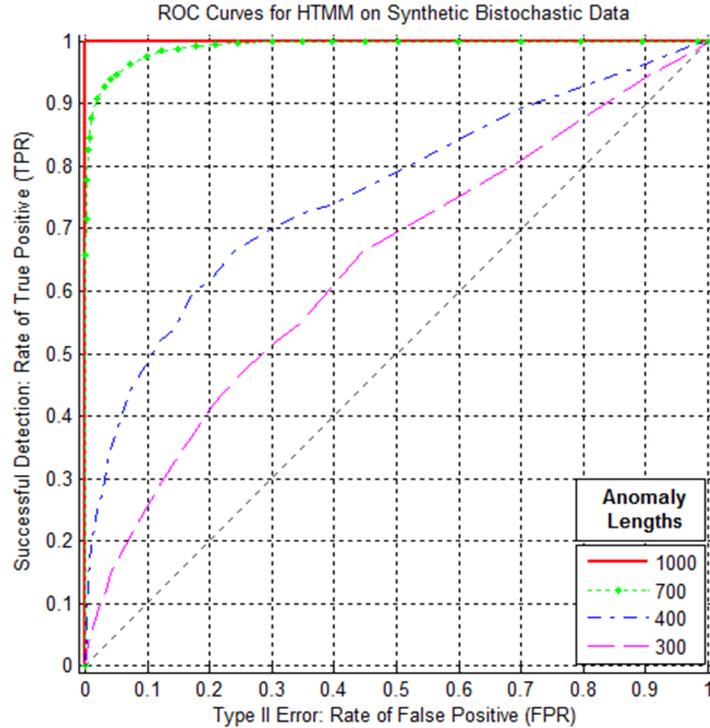
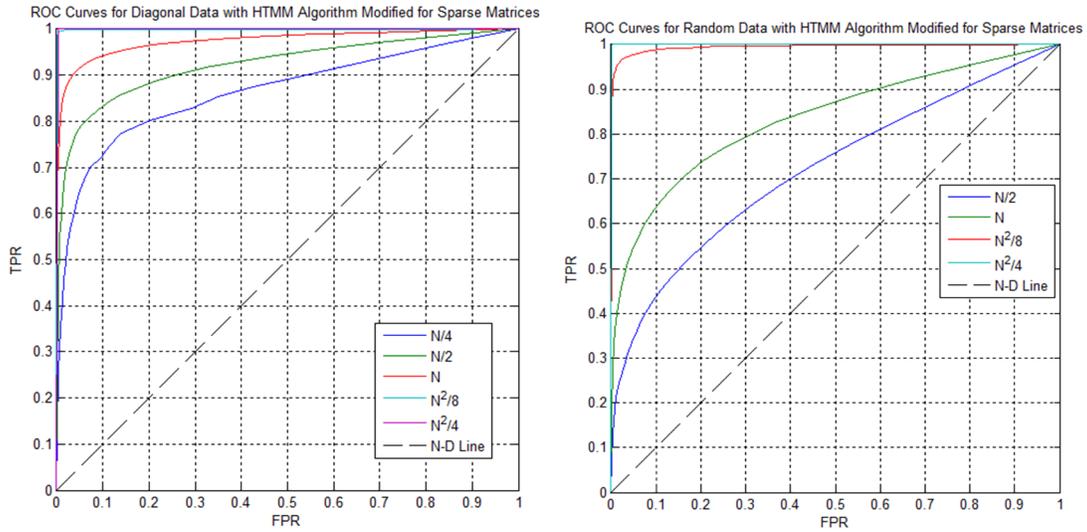


Figure 37. ROC Curves tested against 500 anomalous spans for various anomaly lengths.

From these results, it is clear that any anomalous span of data greater than  $O > 2N^2$  provides highly accurate detection rates – in this case, as high as 99.8% true-positive, and 0.15% false-positive for a 99.9% confidence interval. This method still outperforms a random selector even as the anomaly size is insufficient to populate a TM. Therefore, this method shows promise for sparse and noisy data.

Each TM used for data generation was 20x20, resulting in 20 discrete system states for all test sets, and an  $N^2$  size of 400 (the minimum number of samples for non-sparse test TM population). For each of the above data cases, ROC curves were generated as above in Section 3.4.1. For all cases, HTMM outperformed a random selector even for sample sizes as small as  $L=5$  (diagonal case). Calculation times ranged from between 0.0003 s/window ( $L=5$ ) and 0.001 s/window ( $L=400$ ). Figure 38 below shows the ROC results for each of these three test cases.

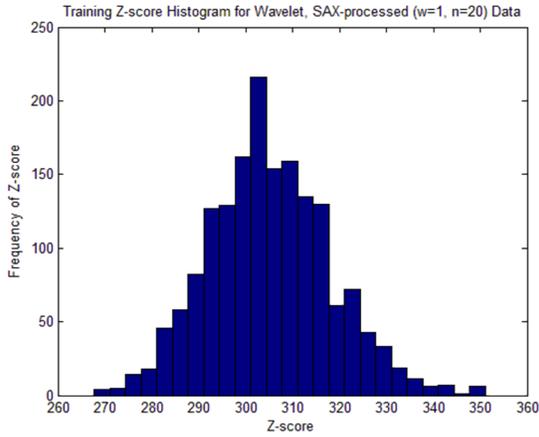


(a) (b)  
Figure 38. ROC curves for diagonal (a), random (b) TMs

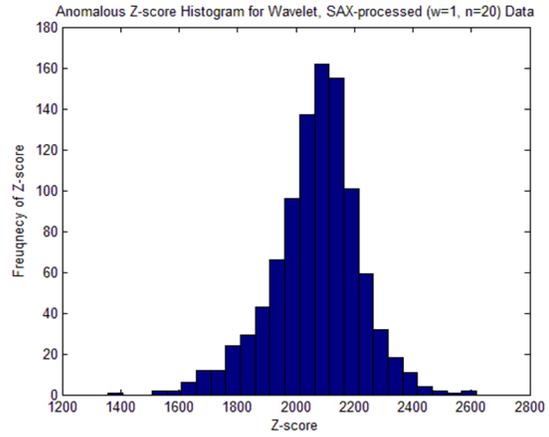
### 3.7.3 Experimental Bearing Fault Data using HTMM

Using the same HTMM approach outlined in Section 3.6, the experimental bearing fault data were analyzed for anomalies. For this test, data were processed at two separate SAX window lengths to demonstrate HTMM robustness against frequency dependence. All data were initially processed using a DB1 discrete wavelet transform window, and a reduced-dimensionality alphabet size of  $N=20$ .

Initially, the data were processed using a SAX window of 1 (equivalent to a non-windowed approach) – HTMM successfully flagged all known anomalies within the dataset, and yielded a 100% true positive, 0% false-positive performance rate at a 99.96% confidence interval.



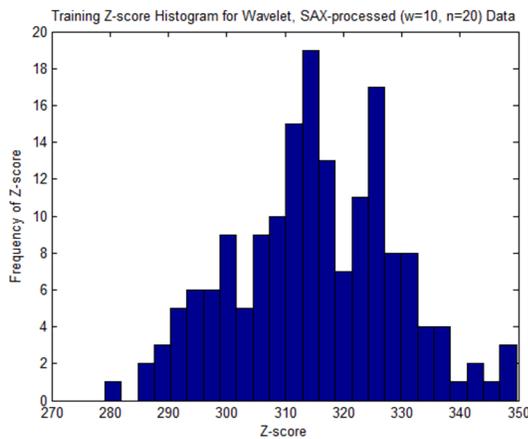
(a)



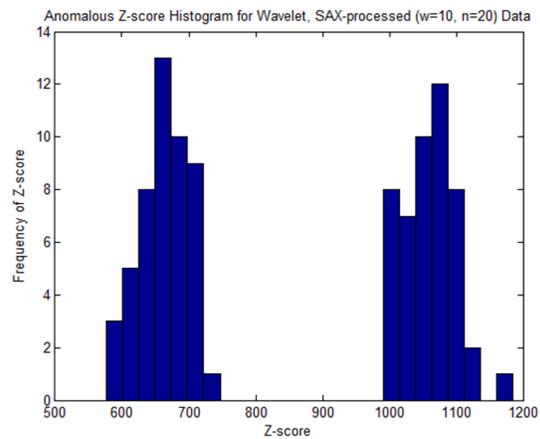
(b)

Figure 39. Z-Scores for DB1 / SAX window 1 experimental normal (a) and anomalous (b) data

To demonstrate the validity of HTMM against low-resolution data, the same data were aggregated using SAX at a window length of 10, an order of magnitude lower than the previous test. Again, HTMM yielded highly accurate results and successfully flagged all known anomalies, with a 100% true positive, 0% false positive performance rate at a 99.9% confidence interval.



(a)



(b)

Figure 40. Z-Scores for DB1 / SAX window 10 experimental normal (a) and anomalous (b) data

At the 12 kHz sampling rate used to obtain the experimental data, a 1000-point anomaly would occupy less than 0.1 seconds. As the average processing time to detect such an event is  $2 \times 10^{-4}$  seconds, this method shows great promise for applications in real-time anomaly detection.

### 3.8 Performance Comparison

#### 3.8.1 Demonstration of Computational Efficiency

The hierarchical approach described above is advantageous for its computational efficiency and rapid detection of anomalous patterns in time-series data. The computational cost of eigen-image decomposition (as outlined in Section 1.2.2) is significant, often on the order of  $O:R^3$  where  $R$  is the number of pixels in the image to be processed. When combined with the computational cost of generating transition matrices from data, the overall cost of these types of methods can grow exponentially.

For HTMM, the largest computational load comes from the generation of TM's themselves, and the calculation of each TM's Z-score. For a test section of data  $X_{test}$  containing  $L$  data points, the generation of the  $N$  by  $N$  transition matrix  $P$  costs  $O_{p[test]} : L + (2N)^2$  where  $L$  is the cost of populating the transition matrix, and  $(2N)^2$  is the cost of row-normalizing the TM to be row-stochastic. The computational cost of comparing this TM to the training results as shown above in Equation (57) is on the order of  $4(N)^2$ .

Searching a Nearest-Neighbors database for a best match for a TM containing  $N^2$  elements has a running time of  $O : N^2 R$  where  $R$  is the number of recorded training transition matrices (using the Naïve Nearest Neighbors algorithm) (Weber, Schek, & Blott, 1998). In contrast, analyzing a candidate TM's placement as a function of the Z-score difference metric requires only two operations: one to calculate the anomalousness metric of  $T$  as in Equation (68), and one comparator to determine if the Z-score is within the CI bounds  $a$  and  $b$  as in Equation

(67). This results in very rapid and robust anomaly detection, as the calculations can be performed in real-time.

### 3.8.2 Results in Experimental Data using Nearest Neighbors

Using the Nearest Neighbors approach to Transition Matrices, detection rates of 91% true positive and 1.4% false positive were obtained at a 99.0% confidence interval. Histograms of these Euclidian Distance metrics are shown below in Figure 41.

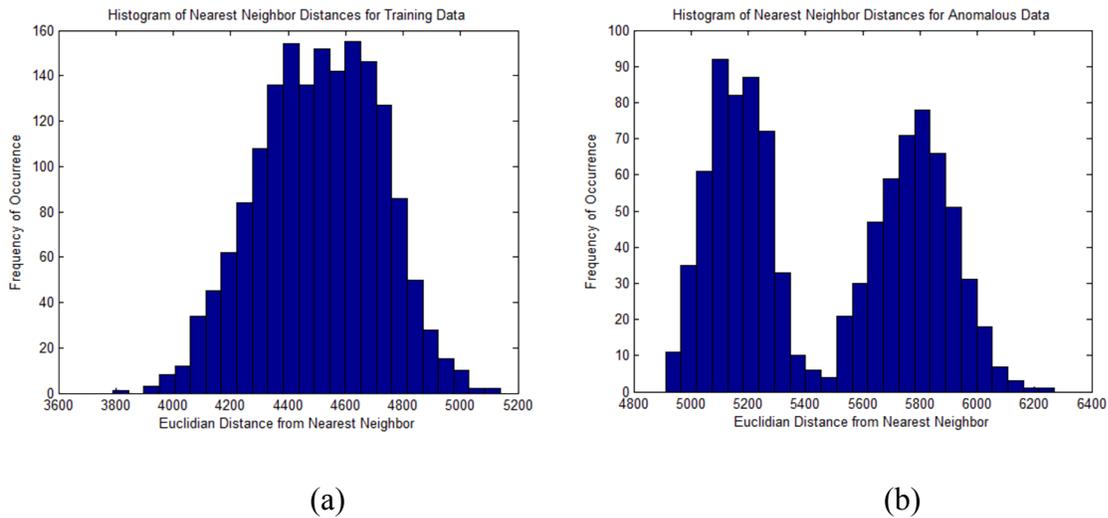


Figure 41. Nearest Neighbor detection for experimental normal (a) and anomalous (b) data

For the experimental dataset, this process took 104 seconds of computing time to complete, which is longer than the length of the data (81.4 seconds). Therefore, this method would be impractical for use in applications requiring real-time detection. Additionally, this approach breaks down when presented with large amounts of data. The Law of Large Numbers dictates that the probability of a “good” match to a neighbor candidate (even within a completely anomalous dataset) becomes greater as the set size approaches infinity. When presented with large amounts of data, this type of anomaly detection method becomes computationally intensive and suffers significant issues with accuracy (Chen S. L., 2010).

### 3.8.3 Results in Synthetic Data using Nearest Neighbors

The Nearest Neighbors approach was lastly applied to the synthetic bistoochastic data. The best true positive rate obtained using this method for bistoochastic data is 4.1%, which fails to meet good detection criteria. The histogram range for this test is shown below in Figure 42, demonstrating an almost complete overlap between normal and anomalous difference metrics.

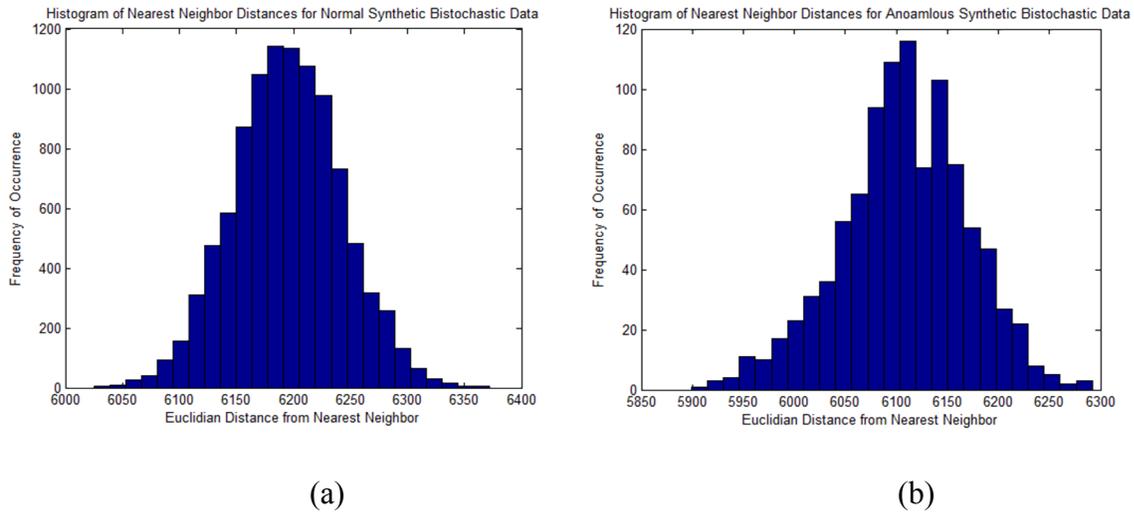


Figure 42. Nearest Neighbor detection for synthetic normal (a) and anomalous (b) data

Table 7 through Table 10 below represent a summary of training data types, anomaly sizes, performance metrics, and computation times (for all test points):

Table 7. Synthetic ground-truth bistoochastic data, using Nearest Neighbors

# Training Points	# Test Points	TPR	FPR	Time to Compute
10,000,000	1,000,000	4.1%	2.4%	192s
1,000,000	100,000	1.1%	2.3%	29s

Table 8. Synthetic ground-truth bistoochastic data, using HTMM

# Training Points	# Test Points	TPR	FPR	Time to Compute
10,000,000	1,000,000	99.8%	0.15%	0.9s
1,000,000	100,000	99.1%	0.5%	0.2s

Table 9. Experimental bearing fault data, using Nearest Neighbors

SAX Window Length	TPR	FPR	Time to Compute
1 point (unwindowed)	91%	1.4%	104s
10 points	80%	1.0%	83s

Table 10. Experimental bearing fault data, using HTMM

SAX Window Length	TPR	FPR	Time to Compute
1 point (unwindowed)	100%	0%	0.4s
10 points	100%	0%	0.2s

### 3.9 Discussion

#### 3.9.1 Conclusions

In a ground-truth simulated dataset with 500 anomalies, in which the normal data and anomalous data were indistinguishable using frequency analysis and first order statistics, the HTMM approach produced highly accurate results at 99.8% TPR: 0.15% FPR at a 99.9% confidence interval. This method was also successful in outperforming a random selector even as the length of anomalies dropped below the number of state transitions in the data, demonstrating robustness against sparse data, as well as data with lower sampling rates.

The HTMM method developed in this chapter also outperforms comparable methods by significantly reducing the required processing time for detection. In a  $1 \times 10^7$  point synthetic bistochastic dataset, the previously mentioned 0.998 TPR : 0.0015 FPR ratio was obtained in under 1.9 seconds, which yields a processing time of approximately  $2 \times 10^{-4}$  seconds per 1000-point window of data. In contrast, the Nearest Neighbors method took nearly 239 seconds to reach a performance ratio of 0.041 TPR : 0.024 FPR in synthetic data. This chapter shows that

the HTMM method is a significant step towards the goal of real-time rapid anomaly detection, even in seemingly random data. HTMM yields high true-positive detection rates (>99%) and low false-positive detection rates (<1%) using  $N^2/8$  and  $N^2/4$  points/window for sparse and purely random transitional probability cases respectively, indicating extensibility into a variety of system types.

Finally, when applied to experimental data from real bearings, the HTMM method successfully identified more than 99.9% of the anomalous data available with less than 0.1% error. This approach outperforms the Nearest Neighbors method both in accuracy (0.91 TPR : 0.14 FPR) and speed (1250% faster).

The results clearly demonstrate that this method produces highly accurate results in less computation time than similar approaches. These results also show the advantages of the HTMM method as applied to: noisy data, data which are unsuitable for frequency-domain analysis, and sparse experimentally-obtained data.

### **3.9.2 Future Work**

There are several possible avenues for future work based on this approach to anomaly detection. First, this work has application in anomaly detection for stochastic processes other than bearing fault data, and should be applied to other state-dependent systems such as streaming video sources and other multi-dimensional data types. This approach should be applied to other synthetic data sets and simulation models to further test its capacity for anomaly detection in sparse or noisy data. Finally, this method stands to be implemented as a real-time anomaly detection metric in a physical system for active monitoring.

## 4 ANOMALY DETECTION IN PARAMETER-DRIVEN DATA

Precision monitoring systems are increasingly commonplace due to exacting requirements for tolerances in small-parts machining, Microelectromechanical Systems (MEMS) equipment manufacturing, and other mechanical processes. Variance monitoring is a common practice in fields such as Integrated Circuit (IC) machining and statistical quality control, where the variance parameter (e.g., distance of a chip from the center of the wafer) may indicate manufacturing defect. These types of analysis methods typically rely on monitoring a system parameter of interest. Previous approaches to have involved statistical techniques such as Neural Networks and cumulative sum (CUSUM), which can be computationally expensive and often require substantial data to perform accurately. This chapter develops a novel approach to anomaly detection in both simulated and experimental parameter-driven processes, using a newly introduced method: Bayesian Posteriors Updated Sequentially and Hierarchically (BPUSH). This approach outperforms previous methods, achieving reliable detection results with low computational cost and low false alarm rates ( $\sim 0.1\%$ ). Finally, this chapter shows that sample size requirements for BPUSH fall well below typical sizes for similar hypothesis testing in traditional statistics, achieving True Positive Rates (TPR)  $> 99\%$  using as few as  $n=25$  samples.

### 4.1 Background

The proposed sequential updating method described in this chapter reduces the required number of samples for accurate detection. This reduction results from the agglomerative nature of sequential updating, demonstrated in greater detail in Section 1.2.7. The process of anomaly detection uses small sets of data for training and testing, which are compared to all previous observations. Because there is no minimum requirement for statistical significance, this approach is highly scalable.

This chapter details the derivation and implementation of a novel approach to anomaly detection in quality monitoring of precision manufacturing systems. In BPUSH, anomalies in a parameter-driven system are detected by monitoring measurable system signals of interest, particularly in how they affect the shape of posterior Cumulative Distribution Functions (CDF's). Sequential updating yields stable parameters for a prior distribution of the parameters of interest. This prior distribution acts as the basis for all training posteriors. The training posteriors are fitted to a hierarchical distribution – in this case, using closed-form integrals to calculate the area between posterior curves. This hierarchical distribution ranks the “normalcy” of any given subset of training data relative to the average posterior. In test data, posteriors generated from anomalous data are expected to fall outside of a user-specified confidence interval. The advantages of BPUSH are demonstrated in Section 4.4 of this chapter, and include:

- Rapid detection of anomalies at real-time processing speeds
- Accuracy that outperforms previous approaches such as variance tracking
- Reduction in required test sample sizes to as few as  $n=10$  units

## 4.2 Proposed method

This chapter describes a novel method for detecting anomalies in precision-based manufacturing. Using sequential updating, a stable posterior estimate is obtained for the precision parameter of a manufacturing process. Using this posterior estimate as a fixed prior, Bayesian analysis is performed on small identically-sized subsets of training data known to be “normal”. The resulting CDF's are compared to the mean CDF using an area-based difference metric which is then hierarchically parameterized across all training samples. The final parameterization of this difference metric fits a Gaussian distribution at the top level of the hierarchical model. This difference metric quantifies the normalcy of any given data subset.

Depending on a test set’s distance-to-mean relative to a user-specified confidence interval, that set is automatically classified as either “normal” or “anomalous”. A flowchart illustrating this process in greater detail is shown in Figure 47 in Section 4.2.4.

This chapter assumes a precision-based manufacturing process that produces independent and identically distributed (i.i.d.) units that must meet a specified tolerance. These units are referenced to a known datum such that the mean value for the measured tolerance dimension is constant. Figure 43 below shows a representation of this type of production process.

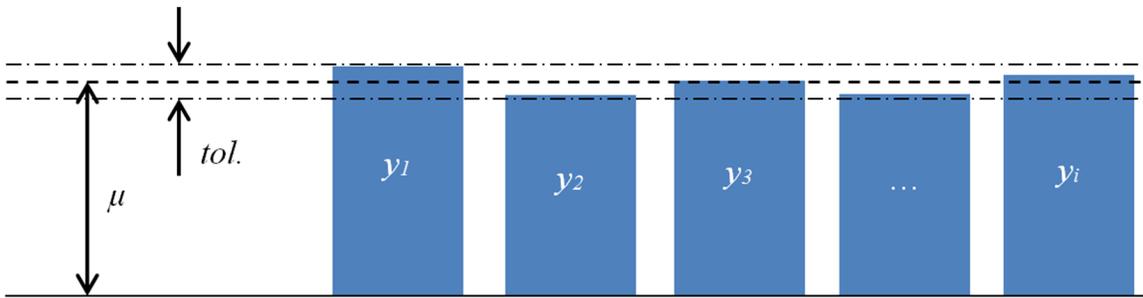


Figure 43. Example production of parts  $y_i$  with mean  $\mu$  and precision  $\phi$ .

In the above example Figure 43, units  $\{1, 2 \dots i\}$  are generated from a synthetic manufacturing process. These parts may contain multiple dimensions, but the signal of interest is measured as  $y$ , with an expected mean value  $\mu$  and specified tolerance  $tol$ . If the distribution of  $y$  is Gaussian, then the tolerance is an estimator of the precision parameter  $\phi$  such that some user-specified percentage of parts (e.g., 99.9%) should fall within “normal” ranges:  $tol \cong 3\phi^{-1}$

#### 4.2.1 Assumptions

The following assumptions allow the derivation of the full joint posterior distribution:

1. Samples  $y_i$  are drawn from a normal distribution:  $y_i | \phi \sim N(\mu, \phi^{-1})$
2. Mean  $\mu$  is stationary,  $n$  is fixed, and samples are ergodic
3. The variance of the parts is significantly smaller than the part dimension / aspect

The variable of interest in this case is the precision,  $\phi$ . Assume that  $\phi$  is driven by a 2-parameter Gamma distribution with shape parameter  $\alpha$  and rate parameter  $\beta$ :  $\phi \sim Gam(\alpha, \beta)$

The impact of using these assumptions is discussed in the next section.

#### 4.2.2 Derivation of Joint Posterior & Sequential Updater

Using Bayes' law, the estimation of the precision parameter based on  $y$  (the overall population, not a sample subset) is given by:

$$P(\phi | y) = P(y | \phi) \frac{P(\phi)}{P(y)} \quad (69)$$

As demonstrated by (Jordan, 2010) and (Murphy, 2007), the Gamma distribution is a conjugate prior for the Normal Distribution, validating Assumption 1 from the previous section. The posterior distribution is derived briefly here in Equations (70) through (72), and given in full in Appendix A. Note that the posterior in this case is derived using a sample set  $\tilde{y}$  containing observations  $y_1, y_2 \dots y_n$ .

$$P(\phi | \tilde{y}) \propto \prod_{i=1}^n [P(y_i | \phi)] P(\phi) \quad (70)$$

$$P(\phi | \tilde{y}) \propto \frac{B^A}{\Gamma(A)} \phi^{A-1} e^{-B\phi} \quad (71)$$

$$P(\phi | \tilde{y}) \sim Gam(A, B) \begin{cases} A = \alpha + \frac{n}{2} \\ B = \beta + \sum_{i=1}^n (y_i - \mu)^2 \end{cases} \quad (72)$$

The Gamma Distribution has the following Cumulative Distribution Function (CDF):

$$CDF = \frac{\gamma(A, B\phi)}{\Gamma(A)} \quad (73)$$

where  $\Gamma$  is the Gamma Function and  $\gamma$  is the Lower Incomplete Gamma Function.

The Full, Lower-Incomplete, and Upper-Incomplete Gamma Functions are defined by:

$$\Gamma(a) = \int_0^{\infty} t^{a-1} e^{-t} dt \quad (74)$$

$$\gamma(a, bx) = \int_0^{bx} t^{a-1} e^{-t} dt \quad (75)$$

$$\Gamma(a, bx) = \int_{bx}^{\infty} t^{a-1} e^{-t} dt \quad (76)$$

which lead to the identity:

$$\gamma(a, bx) + \Gamma(a, bx) = \Gamma(a) \quad (77)$$

This identity plays an important role in the solution of closed form integrals in Section 4.2.3.

### 4.2.3 Derivation of Distance Function between Gamma Posterior Distributions

The Cramer von Mises (CVM) criterion is a widely used goodness-of-fit statistic for comparing two empirical distributions (Thode, 2002). In this application, the difference metric of interest is proportional to the area between two CDF's. For two arbitrary functions  $F_1$  and  $F_2$ , the CVM statistic  $w$  is formally defined as:

$$CVM = \int_{-\infty}^{\infty} [F_2(x) - F_1(x)]^2 dF_2(x) \quad (78)$$

In statistics, the CVM criterion is useful for determining whether a sample set of data were generated from an expected distribution. This integral is also useful as a nonparametric estimator of the area between two functions. However, the Gamma Distribution's CDF from Equation (73) does not have a closed-form integral for the CVM criterion, as there is no closed-form solution to the integrals:

$$\int \gamma(a, b\phi)^2 \quad \text{or} \quad \int \Gamma(a, b\phi)^2 \quad (79)$$

However, if one function is monotonically larger than another function, then the area between two curves becomes a simple subtraction of integrals. Using the sequential updating

formula derived in Equation (71), if two posterior Gamma Distributions are generated from the same prior using the same sample size  $n$ , they share the same  $A$  value per Equation (72).

Because the CDF's of these Gamma Distributions depend on the Incomplete Gamma Function, they differ only in the rate parameter  $B$ . As the integral in Equation (76) is lower-bounded by  $B$ , it follows that any two posterior CDF's sharing the same  $A$  value must never intersect.

$$\Gamma(a, b_1 x) \leq \Gamma(a, b_2 x) \quad \text{where } b_1 > b_2 \quad (80)$$

Using this identity, the area between two Gamma Distribution CDF's, which share the same shape parameter  $A$ , is:

$$w = \int_0^{\infty} \left[ \frac{\gamma(A, B_1 \phi)}{\Gamma(A)} - \frac{\gamma(A, B_2 \phi)}{\Gamma(A)} \right] d\phi \quad (81)$$

Equation (80) holds true as long as the sample size  $n$  is fixed for all posteriors. If this requirement is met, the integral in Equation (81) reduces to Equation (82) using the full derivation given in Appendix B. The resulting difference metric obtained from completing the closed-form integral in Equation (81) is:

$$w = A \left( \frac{1}{B_2} - \frac{1}{B_1} \right) \quad (82)$$

This final value  $w$  is the area difference between two posterior Gamma Distribution CDF's generated from the same prior distribution, using the same sample size  $n$ . By using this simplified calculation instead of numerical integration, the area difference metric  $w$  can be calculated very rapidly, allowing real-time analysis and anomaly detection. This difference metric is then parameterized hierarchically, and used to determine the “normalcy” of a given subset of test data as described in the following section.

#### 4.2.4 Hierarchical Parameterization of Difference Metric

The parameterization of the difference metric  $w$  is accomplished in two steps: first, the training dataset is sampled and compared to the stabilized prior  $A_0$  and  $B_0$  to achieve a distribution of CDFs with parameters  $A, B^{[m]}$ ; then, these CDF's are compared to the mean CDF to obtain difference metrics  $w^{[m]}$ , which are parameterized hierarchically into a separate distribution. Using  $M$  conditionally independent subsets of training data from dataset  $X$ , all posterior CDF's are obtained for each subset  $x^{[m]}$ . This process is shown below in Figure 44.

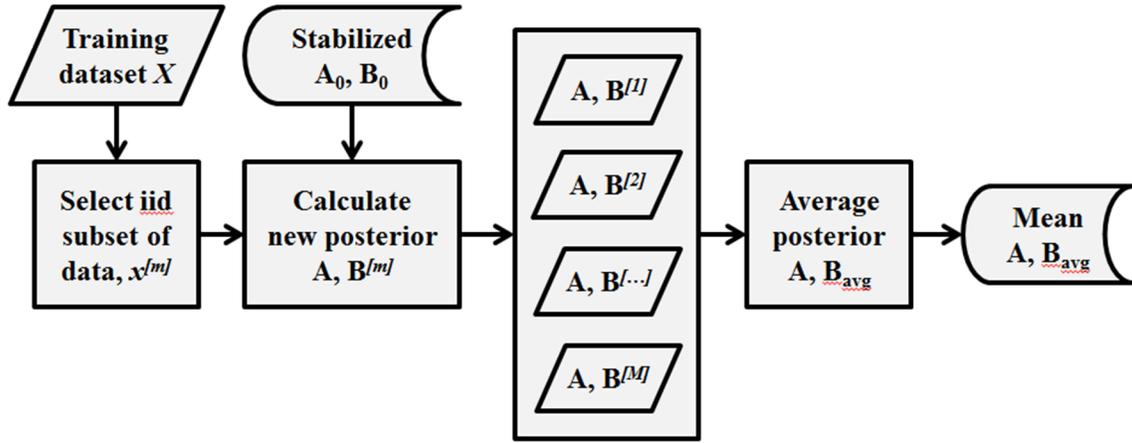


Figure 44. Flowchart for obtaining mean normal posterior CDF

It is important to note that the Euclidian mean of the posterior CDF's is well-approximated by the CDF of the mean of the parameters, shown below in Equation (83).

$$\frac{1}{M} \sum_{m=1}^M GAM(A_m, B_m) \cong GAM\left(\frac{1}{M} \sum_{m=1}^M A_m, \frac{1}{M} \sum_{m=1}^M B_m\right) \quad (83)$$

This comparison is demonstrated below in Figure 45. In Figure 45, the numerical mean of 100 unique CDF's is compared to the CDF of the mean of the same 100 parameters. Error lines between the two distributions are too small to display on the plot; the total area difference between the two curves is on the order of  $1 \times 10^{-5}$ , less than one hundredth of a percentage point.

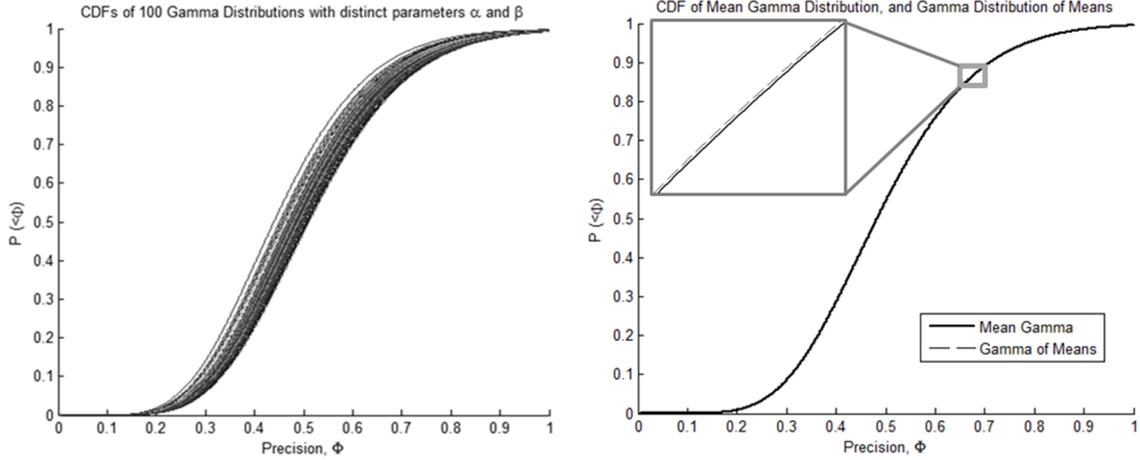


Figure 45. (a) 100 distinct Gamma CDFs, (b) Mean CDF vs. CDF of means.

The identity in Equation (83) ensures that the difference metric calculation performed in Equation (82) is always valid, such that no two distributions ever intersect. This difference metric  $w^{[m]}$  is calculated for each CDF relative to the mean CDF, and then parameterized as a normal distribution with parameters  $\mu_w$  and  $\sigma_w$  as shown in Figure 46. This final distribution defines the anomalousness of candidate subsets of test data.

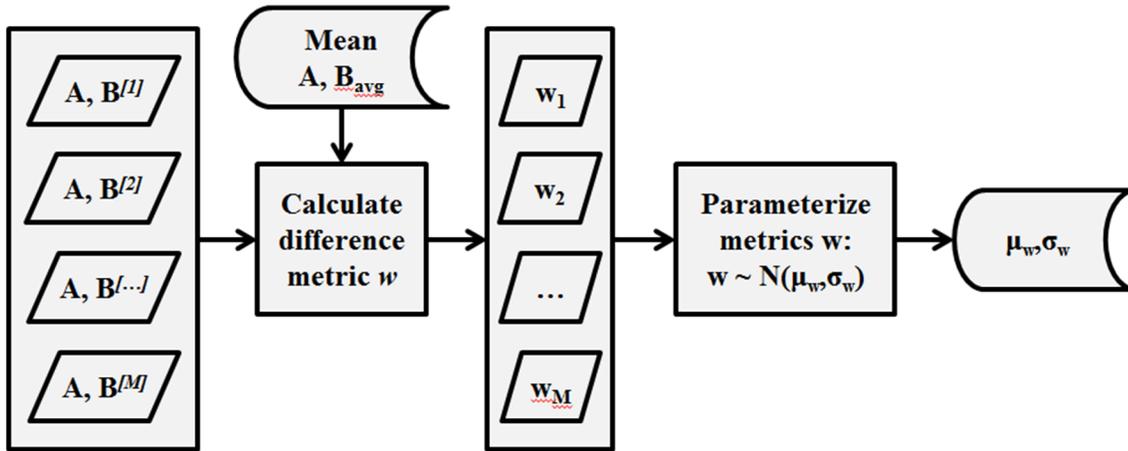


Figure 46. Flowchart for parameterizing difference metric  $w$ .

#### 4.2.5 Anomaly Detection Process

Finally, the anomaly detection process incorporates the results of all previous calculations into a single hierarchical decision tree. For a test dataset  $Y$ , using subsets of identical length  $n$ ,

new posteriors are calculated using stabilized parameters from the training data. The difference metric  $w$  for this test candidate subset is calculated and compared to the training distribution of difference metrics. If this comparison places the candidate set outside of a user-specified confidence interval, the subset is flagged as anomalous. This process is shown below in Figure 47.

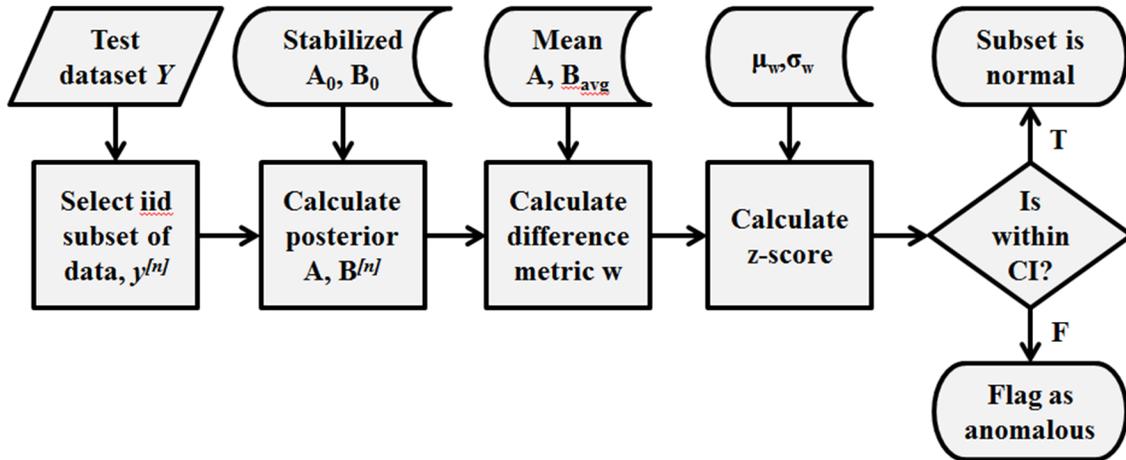


Figure 47. Flowchart for Anomaly Detection Process

### 4.3 Data sets

For this chapter, two separate datasets were used to quantify the performance of BPUSH’s detection of anomalies in precision manufacturing processes. First, a synthetic dataset was generated with ground truth information to assess the performance characteristics of BPUSH: accuracy, speed, robustness against sample size, etc. Second, an experimental dataset was obtained from a wafer etch manufacturing process, and used to assess the performance of BPUSH on real-world data obtained from <http://www.eigenvector.com/data/Etch/>

#### 4.3.1 Simulated Data

A simulated dataset was generated with separate training and test portions. The simulated dataset was generated using a single measurement dimension, modeled after the

process shown in Figure 43. This simulation mimics a process that generates parts with a measurable, fixed dimension and known tolerance of  $\pm 2$  nm. It is assumed that individual parts which fall outside the tolerance are rejected, but that these individual parts may simply be outliers generated from the “normal” process – a normal standard deviation of  $\sigma=0.5$  would result in approximately 0.01% of units being rejected for exceeding tolerances. The anomalous process, which generates parts with a higher standard deviation, may not exceed the tolerance threshold. Applying BPUSH to this process should yield anomaly detection for small groups of units, improving on previous detection methods which rely on thresholds or moving averages.

The training set consisted of 10,000 iid data points with a simulated standard deviation  $\sigma$  of 0.5 nm. The test set consisted of 1,000,000 simulated units with a normal simulated standard deviation of 0.5 nm, identical to the training set. Into this test set were inserted one thousand anomalies at known locations (every 1000 units starting at 901). The anomalous entries spanned a fixed length of  $L$  units each, with a simulated standard deviation of 1.0 nm to generate a significant change from the normal units. This anomalous standard deviation would result in less than 5% of anomalous units being rejected due to exceeding tolerance, an increase from the normal rejection rate, but still well below desirable detection levels. The anomaly length  $L$  was set at 10, 20, 25, and 50 units for each simulation to test BPUSH’s ability to detect anomalies in smaller batches than previous methods.

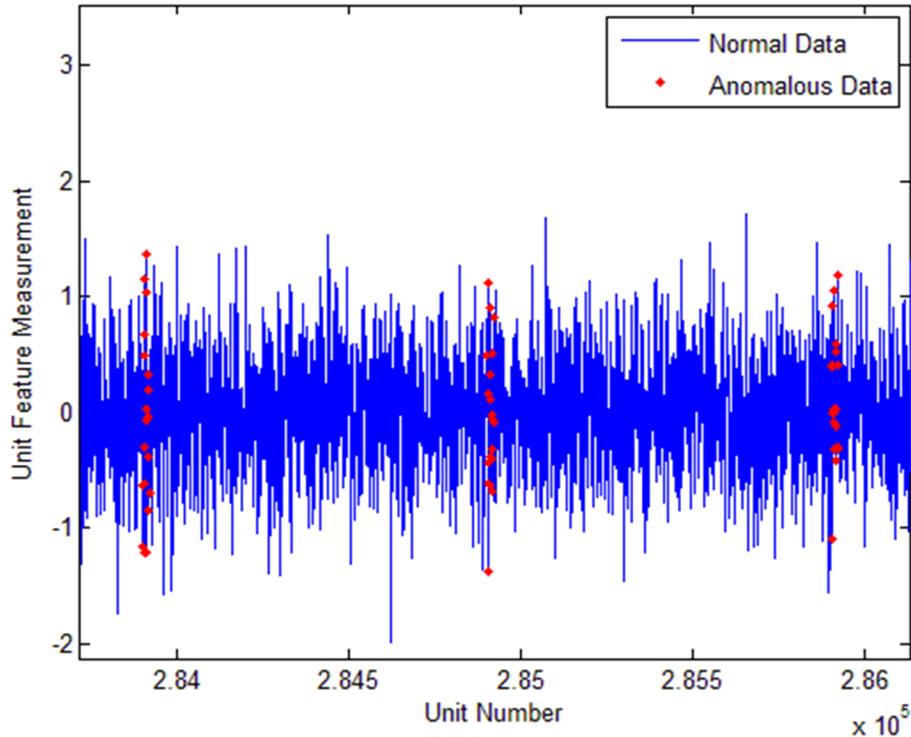


Figure 48. Simulated Precision Manufacturing Data with Anomalies,  $L=20$

Figure 48 shows a series of normal parts (gray) with anomalous measurements highlighted in black. This figure illustrates the difficulty in identifying anomalous parts using traditional Z-score metrics, and even the CUSUM / MOSUM algorithms. The small variance and sample size of these subsets of data hinder reliable classification using traditional methods.

### 4.3.2 Experimental Data

Empirical precision manufacturing process data was also obtained to test the reliability of the BPUSH method on real-world systems. From the data manufacturer:

*“This data set consists of the engineering variables from a LAM 9600 Metal Etcher over the course of etching 129 wafers. The data consists of 108 normal wafers taken during 3 experiments (numbers 29, 31 and 33) and 21 wafers with intentionally induced faults taken during the same experiments. Note that the experiments were run several weeks apart and*

*data from different experiments has a different mean and somewhat different covariance structure.”*

This secondary dataset spanned a total of 129 units measured across 20 different parameters, with approximately 100 time-series measurements per parameter (RF Load, Valve State, etc.), per unit produced. Previous work has attempted to classify anomalous units based on analysis of all parameters simultaneously using dimensionality reduction techniques such as Principal Component Analysis (Wise, Gallagher, Butler, White, & Barna, 1999). The 21 parameters, the first of which is “Time”, are shown below in Table 11:

Table 11. List of Empirically Measured Parameters

1. Time	7. Endpt A	12. RF Phase Err	17. TCP Impedance
2. Step Number	8. He Press	13. RF Pwr	18. TCP Top Pwr
3. BCl3 Flow	9. Pressure	14. RF Impedance	19. TCP Rfl Pwr
4. Cl2 Flow	10. RF Tuner	15. TCP Tuner	20. TCP Load
5. RF Btm Pwr	11. RF Load	16. TCP Phase Err	21. Vat Valve
6. RF Btm Rfl Pwr			

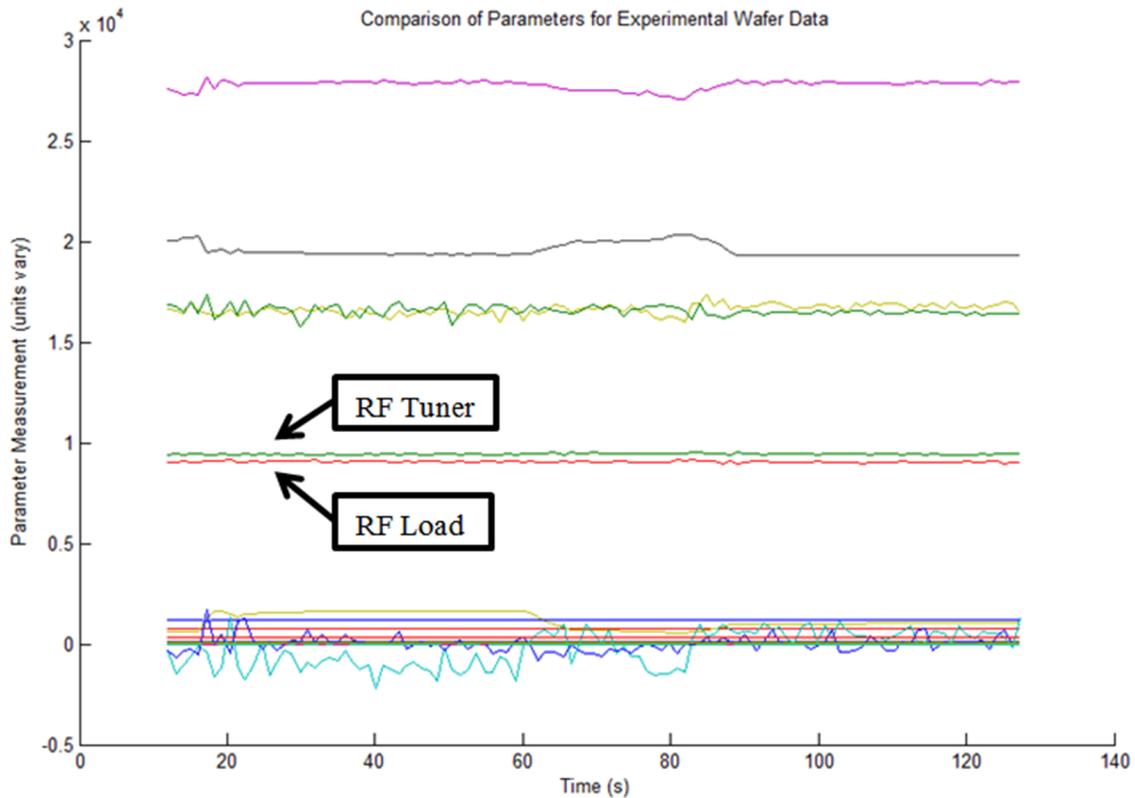


Figure 49. Comparison of Experimental Measurement Parameters

In order to apply BPUSH to the experimental dataset, a single parameter (RF Load) was selected for its ergodic nature, relative stability, and tendency for anomalous units to exhibit lower precision as shown above in Figure 49. Other parameters such as Valve State, Pressure, and TCP variables were not used due to lack of ergodicity, significantly reduced dimensionality, or lack of change between normal vs. anomalous data. To further reduce the effects of hysteresis and data resolution, all datasets were preprocessed time-series using Symbolic Aggregate Approximation (SAX). The SAX algorithm is a common dimensionality reduction technique. In this dataset, SAX was performed at various window-lengths and cardinalities for data processing (Keogh, Lin, & Fu, 2005). In SAX, the amplitude of stationary time-series data are fitted to a Gaussian distribution, separated into equiprobable “bins”. Each of these amplitude

bins has a letter or symbol assigned to it. Time-series data are aggregated into these bins, reducing the overall dimensionality of the data into a finite number of states.

## 4.4 Results

### 4.4.1 Performance Summary, Simulated Data

The simulated test dataset of  $1 \times 10^6$  units was processed comparing BPUSH to other rapid-processing techniques, specifically variance monitoring and MOSUM change detection. Figure 50 shows the results of a parameter tracking algorithm on subsets of 10 units (in gray), with anomalies highlighted using black asterisks. This figure shows one of the pitfalls of variance tracking algorithms: despite relatively rapid processing speed ( $\sim 0.00055$ s per set of 10 units), precision tracking fails to detect a significant number of anomalous samples due to their similarity to normal precision. In the following plots, red asterisks that fall within the bounds of the blue lines (applied as thresholds) are classified as “normal”, resulting in Type I error.

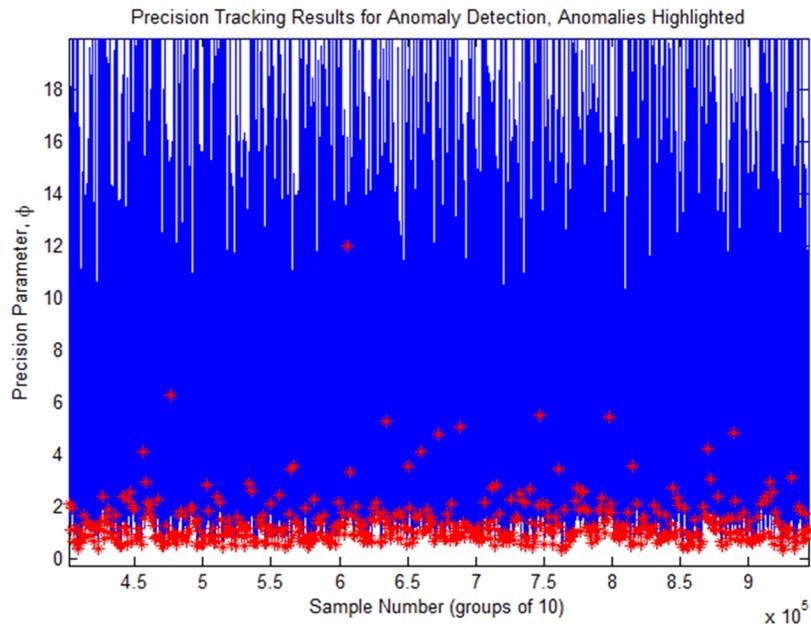


Figure 50. Precision Tracking in Simulated Precision Process Data,  $n=10$

In Figure 51, the Moving Sum of Squares (MOSUM) calculation was applied to the same dataset with sample size  $n=10$ . Residuals for all data are shown in blue, with known anomalous residuals highlighted with red asterisks. From this figure, it is clear that the application of traditional confidence intervals would also exclude a significant portion of the anomalies.

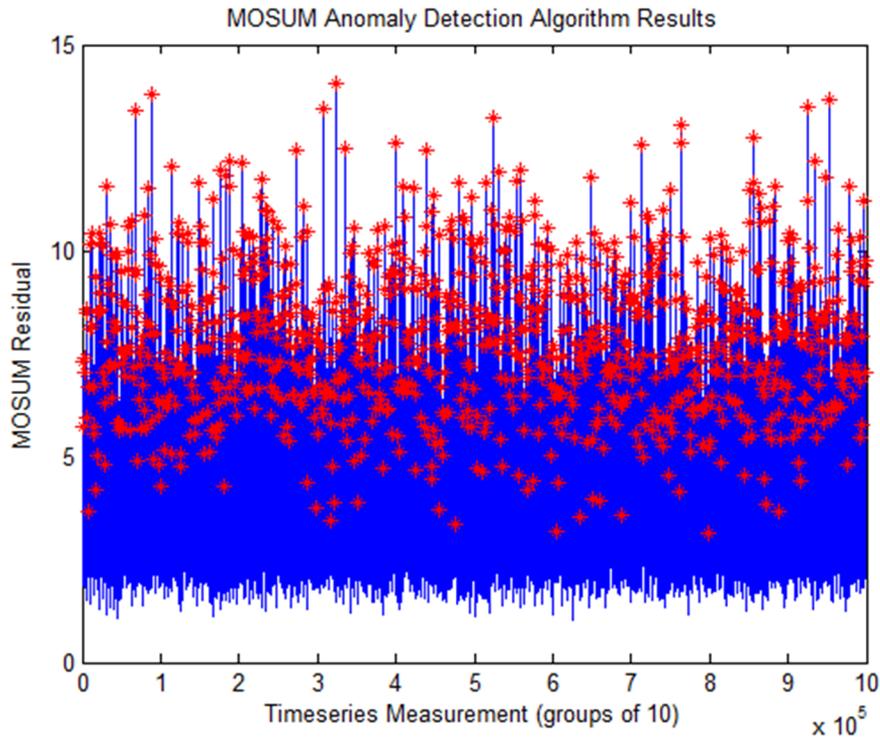


Figure 51. MOSUM Residuals in Simulated Precision Process Data,  $n=10$

All calculations were performed using an Intel Core 2 Duo Processor clocked at 2.00 GHz, with 4.00 GB of DDR2 RAM using Matlab 2011b (64-bit) on Windows 7 (also 64-bit). In all cases, BPUSH was found to exceed the processing speed and detection accuracy of both Precision Tracking and MOSUM. This performance analysis is shown below in Table 12.

Table 12. Performance Characteristics Comparison

System Parameters		BPUSH	MOSUM Test	Precision Tracking
$n=10$	<i>Best TPR / FPR</i>	0.81 / 0.11	0.67 / 0.06	0.72 / 0.10

	<i>Processing time</i>	0.98 s	2.01 s	5.49 s
<i>n=20</i>	<i>Best TPR / FPR</i>	0.98 / 0.03	0.91 / 0.04	0.85 / 0.09
	<i>Processing time</i>	0.82 s	1.06 s	2.66 s
<i>n=50</i>	<i>Best TPR / FPR</i>	1.0 / 0.001	0.998 / 0.03	0.98 / 0.01
	<i>Processing time</i>	0.51 s	0.55 s	1.39 s

These results also demonstrate that the sample sizes required by BPUSH are significantly smaller than what would normally constitute a statistically “valid” sample size. Even modern approaches to determining sample sizes, which rely on conditional probability, require a larger value of  $n$  to reject null hypotheses. Table 1 from (Bartlett II, Kotrlik, & Higgins, 2001) and Table 2 from (Jiroutek, Muller, Kupper, & Stewart, 2003) show that, to generate a sample size with 98% confidence with 3% error, the minimum sample sizes are at least double the requirements of BPUSH to yield 98% TPR / 3% FPR. This result indicates that BPUSH may be applicable to sparse data where few measurements are available or in applications where sampling is expensive.

#### 4.4.2 ROC Curves, Simulated Data

Receiver Operating Characteristic (ROC) curves were generated for the simulated dataset detailed in the previous section. Based on the performance results from Table 12, ROC curves were only generated for the BPUSH method, as this method outperformed previous approaches at all tested sample sizes. Results were obtained by fixing the sample size  $n$  and varying the confidence interval for detection between 0 and 1. Window sizes sampled were  $n = 5, 10, 20, 25,$  and  $50$ . The plot below in Figure 52 show that even for very small sample sizes ( $n = 5$ ), which fail to produce results using other approaches, BPUSH still significantly outperforms a

random selector (ND-Line). Windows larger than  $n = 25$  yielded  $\text{TPR} > 99\%$  detection rates for this dataset with very low ( $\sim 0.1\%$ ) false alarm rates.

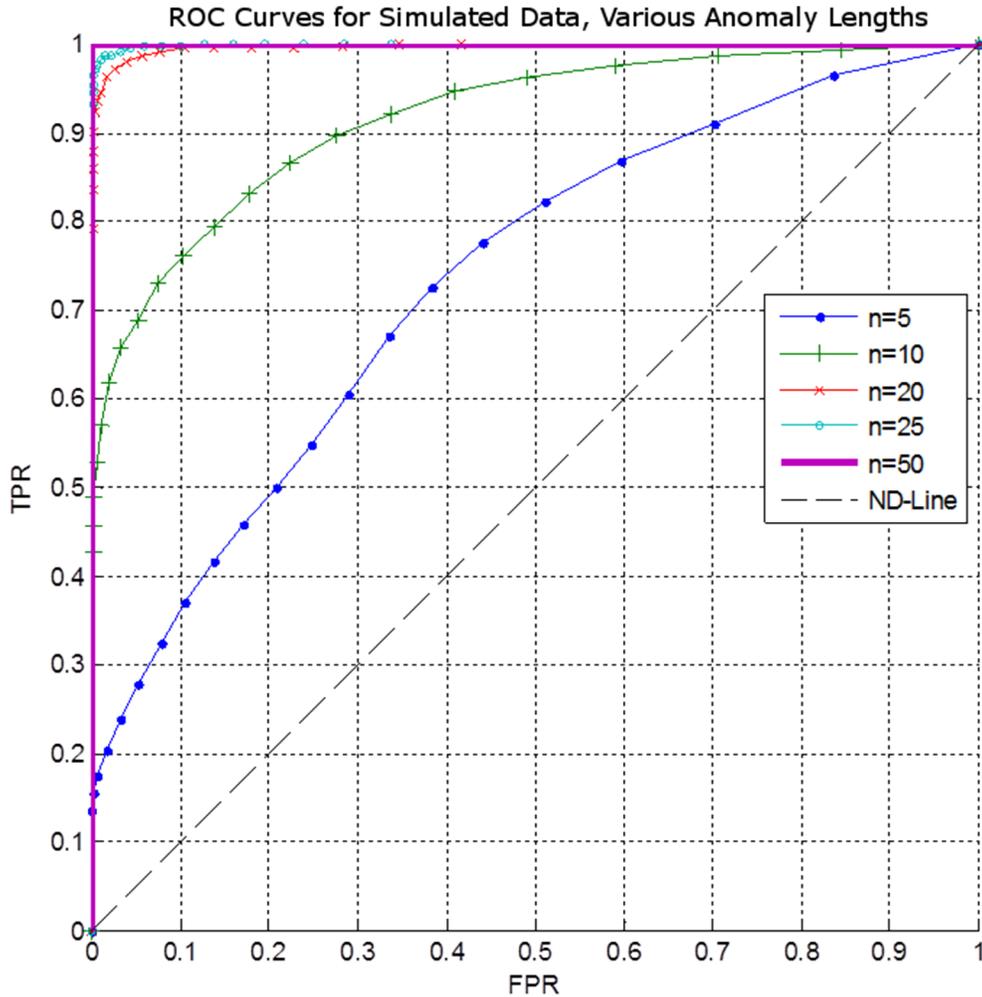


Figure 52. ROC Curves for Simulated Data, Various Anomaly Lengths

#### 4.4.3 Experimental Data

Processing of the experimental data was first done using a SAX cardinality of 20, and an aggregate window size  $n = 20$ . Using this approach, BPUSH achieved 100% detection of anomalous process data, with performance characteristics of  $\text{TPR}: 100\%$ ,  $\text{FPR}: 2.1\%$ . This small sample size is possible due to the agglomerative nature of the BPUSH algorithm, comparing groups of data instead of individual points within a sample set.

A second experimental approach was processed using a SAX cardinality of 10, and an aggregate window size of  $n = 10$ . This reduces the equivalent number of samples, as well as the data resolution, by half. In this data, BPUSH achieved the following performance characteristics: TPR: 91%, FPR: 4.8%.

These results were achieved using only one measurement parameter (RF Load) as a test metric; the incorporation of additional measured system parameters should accelerate the detection process of anomalies into real-time. Data fusion between various datasets is proposed as a topic for future study.

#### **4.5 Conclusions**

The results in this chapter show that the novel method BPUSH is a strong candidate for anomaly detection in precision manufacturing processes, demonstrating robustness against small test sample sizes. By obtaining a closed-form solution for the area between two posterior CDF's, processing speed increases dramatically and allows real-time detection of anomalies. The detection of these anomalies will result in lower manufacturing maintenance costs, rapid implementation of stop-gap methods to prevent unusable parts, and improved overall quality of precision-based processes. The approaches detailed in this chapter should be extended to further experimental data and implemented as real-time detection metrics. The final results of this chapter clearly show the benefits of BPUSH, and its practical applicability to the fields of change-point analysis, precision manufacturing, and quality control in general.

## 5 ANOMALY DETECTION IN SPATIOTEMPORAL DATA

Anomaly detection in large-scale datasets is a relevant problem in the fields of data mining and machine learning, and applications such as unattended ground sensors and persistent surveillance. The specific application in this study involved the monitoring of a geographic area for anomalous activity using sparse information data gathered from a number of trigger-based sensors, including Passive Infrared (PIR) and Beam-Break (BB). The dataset was analyzed for anomalous behavior over a period of 10 months.

To generate a controlled set of data, a simulation was constructed based on pedestrian and vehicular trajectories to mimic the geographical characteristics of the real-world environment. Alert timings were dictated by the movement of vehicles and pedestrians along parameter-driven paths, and a number of anomalous events were inserted into a dataset spanning 260 days. This provided ground-truth and enabled the performance analysis of detection metrics.

Several processing methods were applied to the data: Poisson probability alert count analysis, discrete convolution & image masks for feature extraction, and event-space statistical analysis for anomaly detection. The results demonstrate successful classification of anomalous events such as changes in activity levels and unexpected deviations in agent trajectories. In a ground truth meta-tagged simulation these approaches yield low error with a true-positive rate of 87.5% and a false-positive rate of 0.9% of flagged events.

### 5.1 Introduction

The Real-World Sensor Environment (RWSE) had two principal ingress and egress points, and contained several parking areas and large buildings. Movement throughout the sensor environment was limited to vehicular and walking traffic; 16 sensors were monitored continuously for a period of 10 months and resulted in a total of 644383 agent-generated sensor

alerts. An overhead perspective (not to scale) of this environment is shown in Figure 53: elevated orange sections indicate buildings, gray areas indicate roadways or parking areas, and markers indicate approximate locations and labels of sensors.

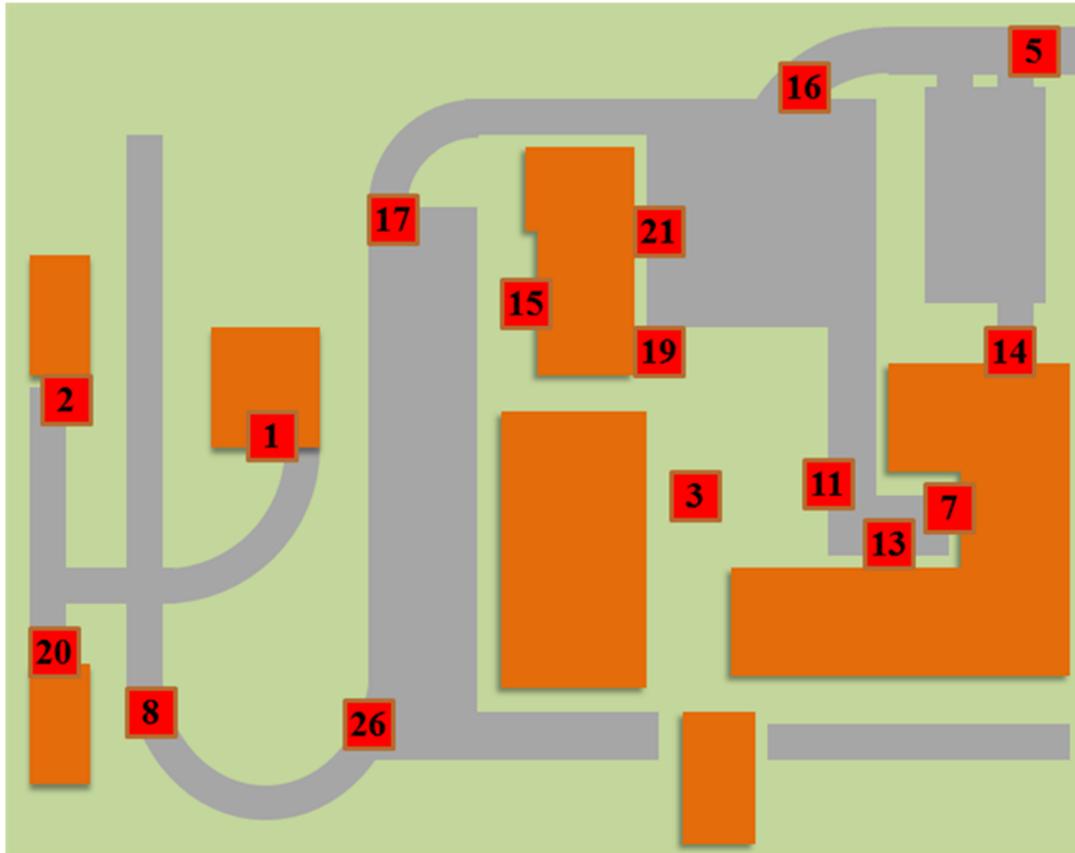


Figure 53. Real World Sensor Environment (RWSE) Sensor Layout

Alerts from these multiple sensor types were combined into a single database of time-stamped trigger events. The resulting logs were parsed into a multi-dimensional data-space, with dimensions for various fixed-length time periods. The data were then analyzed using several approaches in an attempt to identify and classify patterns related to human and vehicular movement through the surveillance area.

To generate a controlled set of data and obtain performance characteristics, a simulation was constructed based on pedestrian and vehicular trajectories. This model provided ground-

truth data and enabled the analysis of meta-tagged events in the data structure, whereas the real sensor network had little to no ground truth. Simulated timing of alerts was dictated by the movement of vehicles and pedestrians along parameter-driven trajectories (one example parameter would be vehicle velocity along a known road). In this model, all parameters were treated as random Gaussian distributions. This approach created a dataset of statistically distributed agent-generated alerts as a time-series signal for each sensor. An “agent” in this case refers to any person, vehicle, or other entity that can trigger sensors and generate alerts. A block diagram of this simulated environment is shown in Figure 54.

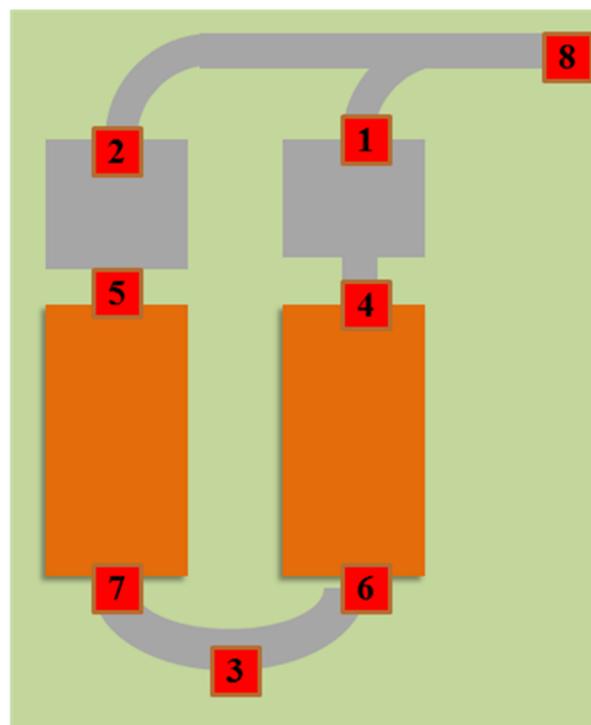


Figure 54. Simulated Sensor Environment Layout

The simulated sensor network data consisted of 260 days of training data, and 260 days of validation data with 5 types of anomalies embedded in the data. These anomalies are:

- Sensor saturation (constant triggering)
- Sensor drop-out (no triggering)

- Population increase (overall environment)
- Population decrease (overall environment)
- Inter-building path change (changes in agent movement)

The validation set contained 12 randomly embedded anomalies.

## 5.2 Approach

For this data, the event-space convolution model was compared to the Poisson probability distribution model. The results show that, while the Poisson model has high detection rates, the false alarm rate (Type II error) of this previous approach is also an issue. The event-space model achieves comparable detection rates with very low (<1%) false alarm rates, demonstrating an improvement over previous work.

Poisson probability distributions are a good candidate for anomaly detection because they model the occurrence of rare events within a given time interval (Devore, 1995). Poisson distributions were fitted to the number of alerts at a given sensor across various timespans (per minute, per hour, etc.). Using training data, these Poisson parameters established a baseline for “normal” behavior. Four numerical scores were calculated as metrics used to detect and classify multiple types of anomalies. The results from this method show that the Poisson probability model is a useful anomaly detection approach, with the added benefit that it does not require any prior knowledge of the anomaly type or form.

Both sensor networks were also analyzed using discrete convolution masks. The convolution approach transforms spatiotemporal data into a low-dimensionality event-space. This event-space was created using an approach based both on image filtering and convolution masking, and analyzed with two separate anomaly-detection metrics: Nearest Neighbor distance, and a probabilistic event likelihood model. This approach treats spatiotemporal data as an image

model, and utilizes image feature extraction methods to identify events in that model. The resulting analysis of this event-space incorporated detection metrics using Nearest Neighbors (for long spans of time) and an event likelihood probabilistic analysis (for short spans of time). The results demonstrate successful detection of anomalous changes in overall activity levels, individual sensor triggering, and unexpected agent trajectories.

### 5.3 Preprocessing

For image manipulation and large timescale data series, Symbolic Aggregate Approximation (SAX) was implemented as a dimensionality reduction approach. In SAX, the data's amplitudes are stratified into equiprobable bins using distance from mean, and then assigned symbolic labels (a, b, etc.) based on bin location. This technique preserves shape trends in the data while significantly reducing the space required to store information (Lin, Wei, Lonardi, & Keough, 2007).

In SAX, the dimensionality reduction algorithm assumes the data are normally distributed as in Equation (84) such that:

$$f(y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(y-\bar{y})^2} \quad (84)$$

Using a normal distribution, the Z-score of a data point is determined by:

$$Z = \frac{y_i - \bar{y}}{\sigma} \quad (85)$$

The data are stratified into equiprobable bins separated at  $y$ -index values  $\gamma_{1,2,\dots,k}$  such that:

$$\int_{\gamma_m}^{\gamma_{m+1}} f(y) \equiv \int_{\gamma_n}^{\gamma_{n+1}} f(y) = \int_{y=\gamma_m}^{y=\gamma_{m+1}} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(y-\bar{y})^2} \quad (86)$$

The result of this manipulation is that each data point is assigned a bin number (1,2,...k) corresponding to its position in the normal distribution. Letters (a, b,... k) are assigned to the

bins, and the data is stored symbolically. This probabilistic approach to dimensionality reduction allows for rapid processing and reduced storage requirements, and creates a scale-invariant data structure for analysis.

#### 5.4 Poisson Distribution Data Model

The Poisson distribution gives the probability,  $P(k|\lambda_{ij})$ , of  $k$  alarms at sensor  $i$ , at time  $j$ , given  $\lambda_{ij}$  expected alarms. The data for each sensor and time (in minutes) is assumed to follow a Poisson distribution. The distribution parameter  $\lambda_{ij}$  was calculated for each sensor at time  $j$  by averaging training data from multiple days, also at time  $j$ . The Poisson probability that an alarm occurs for sensor  $i$ , at a specific time  $j$ , on day  $d$  is given by Equation (87), while Equation (88) shows the time-averaged Poisson probability for some time of day  $j$  over a training period of days  $D$ .

$$P(k_{ijd} | \lambda_{ij}) = \frac{(\lambda_{ij})^{k_{ijd}} e^{-\lambda_{ij}}}{k_{ijd}!} \quad (87)$$

$$P(k_{ij} | \lambda_{ij}) = \frac{\sum_{d=1}^D P(k_{ijd} | \lambda_{ij})}{D} \quad (88)$$

Several numerical scores (which vary from 0 to 1) were developed to detect anomalies from the expected values of the Poisson parameter  $\lambda_{ij}$  by comparison to user-defined thresholds. Equation (89) detects single sensor outages and saturation using the absolute value of the difference between the training mean and the observed Poisson probabilities. After comparing to a threshold value, the remaining alarms are then scanned for consecutive anomalies.  $NS_2$ ,  $NS_3$ , and  $NS_4$  were designed to detect anomalies across multiple sensors, such as increases and decreases in agent population and inter-building movement. Equation (90) is the absolute value of the ratio of change from the average for each sensor. Equation (91) is the average of  $NS_2$  for all sensors. Equation (92) gives the absolute value of the ratio of change from the average for all sensors.

$$NS_1 = \Delta P = |P(k_{ijd} | \lambda_{ij}) - P(k_{ij} | \lambda_{ij})| \quad (89)$$

$$NS_2 = \left| 1 - \frac{\sum_{j=1}^T P(k_{ijd} | \lambda_{ij})}{\sum_{j=1}^T P(k_{ij} | \lambda_{ij})} \right| \quad (90)$$

$$NS_3 = \frac{1}{N} \sum_{i=1}^N NS_2 \quad (91)$$

$$NS_4 = \left| 1 - \frac{\sum_{i=1}^N \sum_{j=1}^T P(k_{ijd} | \lambda_{ij})}{\sum_{i=1}^N \sum_{j=1}^T P(k_{ij} | \lambda_{ij})} \right| \quad (92)$$

These numerical scores quantify the performance of the Poisson model, and are compared against the average number of sensor alarms across user-specified intervals. The following Equations (93)-(94) were substituted into Equations (89)-(92) to obtain these benchmarks. These benchmarks demonstrate an ability to detect anomalies using average sensor alarm frequency without any prior assumed statistical distribution.

$$P(k_{ijd} | \lambda_{ij}) \rightarrow k_{ijd} \quad (93)$$

$$P(k_{ij} | \lambda_{ij}) \rightarrow k_{ij} \quad (94)$$

## 5.5 Poisson Model Analysis and Results

### 5.5.1 Poisson Results: Simulation data.

The Poisson model was evaluated on its ability to detect the types of anomalies enumerated in the simulated environment data (saturation, drop-out, population change, and movement change). Figure 55 (a) shows an example of single sensor saturation detection using  $NS_1$ , Figure 55 (b) shows the detection of increased inter-building movement using  $NS_2$ ,  $NS_3$ , and  $NS_4$ .

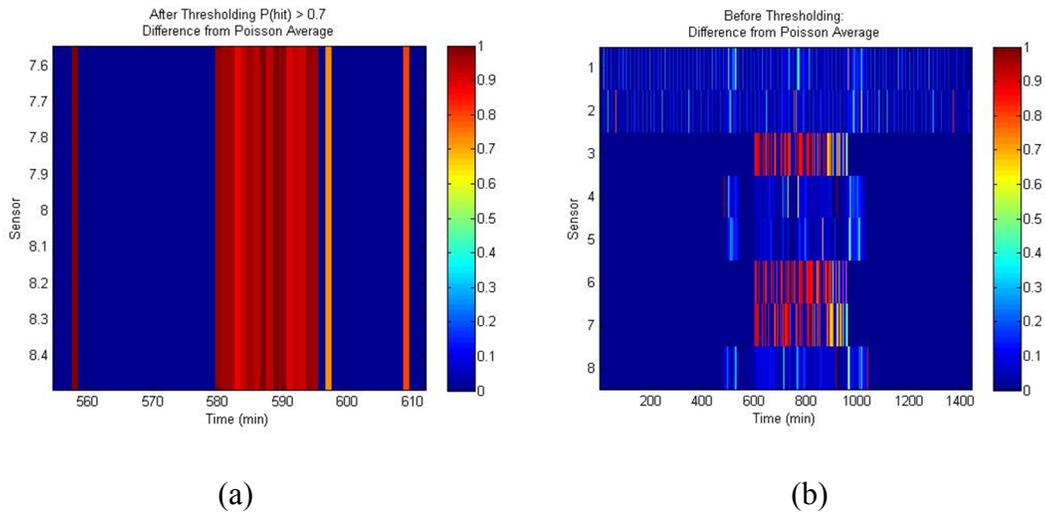


Figure 55. (a) Single Sensor and (b) Multi-Sensor Movement Increase Detected

The thresholded values from Equations (89)-(92) were varied to produce Receiver Operating Characteristic (ROC) curves for the Poisson and summation numerical scores. Three of the 12 anomalies were sensor outages occurring at times when no alarms were regularly present, which resulted in a 25% type II error rate in the Poisson distribution approach. ROC curve results for the Poisson and benchmark numerical scores are shown in Figure 56.

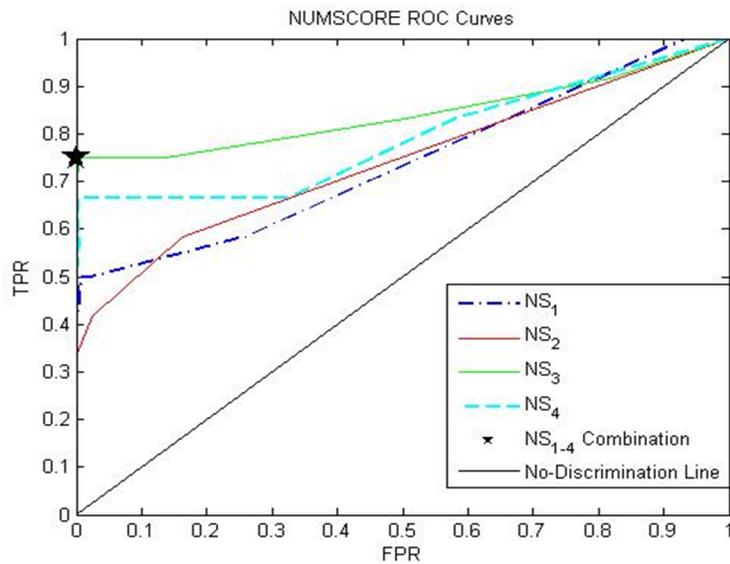


Figure 56. Numerical Score ROC Curves

Figure 56 shows that all Poisson methods ( $NS_1$ - $NS_4$ ) have positive classification characteristic. Simple summations and averages outperform the Poisson method for multiple (simultaneous) sensor anomaly detections.

### 5.5.2 Poisson Results: RWSE Data.

The Poisson distribution and same numerical scores were also calculated using the real-world sensor network data. The training data consisted of 86 weekdays, and the test data consisted of 267 weekdays. Three holidays, Labor Day (Week 36, Day 2), Thanksgiving (Week 48, Day 5), and Christmas (Week 52, Day 6), were assumed to be holidays with low activity.

Table 13 below is a summary of the findings:

Table 13. RWSE Data Results

Anomaly #	Anomaly Type	Description	Comments: $NS$ =numerical score
1	Low Activity	Labor Day	Anomaly detected $NS_4$
2	Low Activity	Thanksgiving	Anomaly detected $NS_4$
3	Low Activity	Christmas	Anomaly detected $NS_4$

Due to the lack of ground truth information for this dataset, holidays were useful as a reference point for general activity levels. The automatic identification of "low activity" anomalies on these holidays shows that the Poisson distribution anomaly detection method detects lower-than-normal activity in sparse noisy data over long periods of time.

## 5.6 Discrete Convolution Event Extraction Model

As demonstrated in (Gonzalez & Woods, 1992), mask convolution is a viable tool for identifying best-match patterns in spatially-linked data. Using a mask  $w(x, y)$  and dataset  $f(x, y)$ , the correlation theorem states that:

$$f(x, y) \otimes w(x, y) \Leftrightarrow F^*(u, v)W(u, v) \quad (95)$$

where  $\otimes$  is the convolution of  $w$  over the domain  $f$ , and  $F^*$  is the complex conjugate of the Fourier Transform of  $f$ . This results in a spatial correlation coefficient, which can be normalized such that it is non-sensitive to scale changes in  $w$  and  $f$ . For purposes of this chapter, all references to the ‘‘correlation coefficient’’ correspond to this normalized value.

Convolution analysis uses two-dimensional event ‘‘masks’’, calculated using sensor group correlations, to extract event information from the data. To obtain correlated sensor groups, the RWSE sensor network data was first analyzed for cross-sensor correlations before performing convolution-based analysis. Because distances, walking times, and other variables (such as path reversal, lingering, and course changes) were not enumerated in the empirical dataset, it was necessary to determine actual lag / lead times between sensors. To accomplish this, the sensors were cross-correlated with each other with varying time lags. The data were condensed into a spatiotemporal  $N$  by  $K$  matrix, where  $N$  is the number of sensors and  $K$  is time in steps of  $k$ .

Equation (96) gives the discrete auto-covariance calculation of a matrix  $A$ , offset by a time-shift,  $\tau$ . This approach is also known as cross-covariance; normalizing by the standard deviation of the data yields the cross-correlation. The resulting cross-covariance matrix of Equation (96) is of size  $N$  by  $N$  by  $T$ , where  $T$  is the maximum time lag.

$$\begin{aligned} COV(A, \tau) &= \frac{1}{K-1} AA_{k-\tau}^T \\ C(i, j, \tau) &= \frac{1}{K-1} \left[ \sum_{k=1}^K A_{i,k} A_{j,k-\tau} \right]_{i,j=1:N} \quad (96) \end{aligned}$$

Each sensor was cross-correlated with all other sensors in increments of 3 to 6 seconds, up to 180 seconds, after which time there was no significant correlative pairing present between any sensors in the network. The results of this correlation allowed generation of image masks for

event analysis. An example event mask is shown in Figure 57, in which Sensor 5 correlates most strongly with itself at a delay of 12 seconds, and then with Sensor 16 at a delay of 18 seconds.

This delay is expected, as the travel time between sensors is approximately 15 seconds.

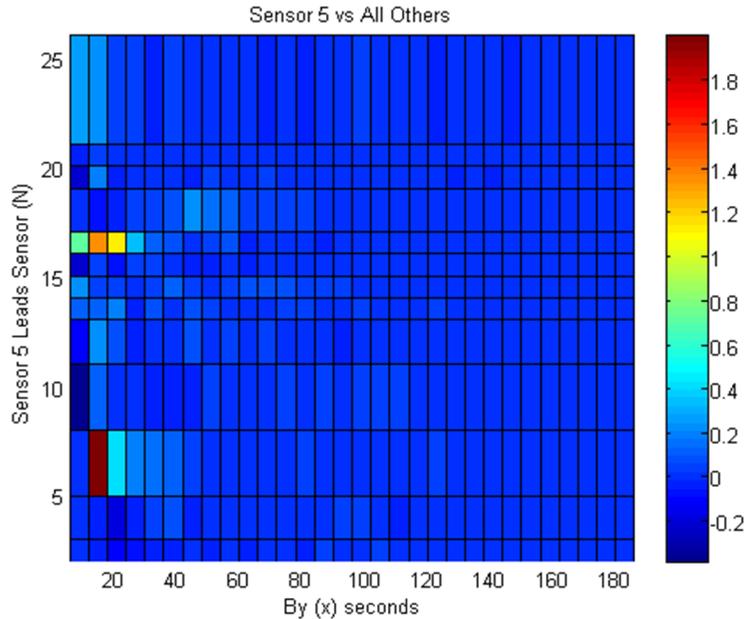


Figure 57. Sensor 5 vs. All Others: Correlation Result Matrix

While empirically-determined masks can be used for detecting events that commonly occur in the data, masks can also be designed to specify events that should not normally occur. For example, since sensors 2, 3, and 16 do not show any significant correlation against each other, an event mask combining all three sensors should be rare within the spatiotemporal data. This enabled the analysis of predicted or user-specified events, as well as naturally-occurring events within the data.

## 5.7 Discrete Convolution Model Analysis and Results

After manually selecting the principally correlated sensor groups, the correlation matrices were used to generate event matrices. Event matrices are built on a similar principle to a convolution mask for image filtering. The strongest correlation matrix was the arrival of a

vehicle (Sensor 5 leading 16 by ~20 seconds), shown above in Figure 57. The convolution mask was normalized to a net value of zero, as governed by Equation (97).

$$\sum_{i=1}^I \sum_{j=1}^J M(i, j) = 0 \quad \text{Example, } M = \begin{bmatrix} 0 & 0 & 0 \\ -1 & -1 & 2 \\ 2 & -1 & -1 \end{bmatrix} \quad (97)$$

When the event mask is convolved with sensor data (in the time domain), the points of high amplitude in the resulting array correspond to occurrences of the event in time. Equation (98) shows the discrete convolution of a spatial mask  $M$  against a spatiotemporal matrix,  $A$ .

$$R(t) = \sum_{t=1}^T \sum_{n=1}^N M_{n,t} A_{n,t+T} = M \otimes (A_{t,t+T})^T \quad (98)$$

Using Thanksgiving week as an example, the data for several weeks prior to Thanksgiving Day (Thursday) were compared to the week of November 24. Sensor saturation events were also analyzed using non-holiday data as training segments. Arrival events, departure events, and intra-network traffic were incorporated into the final data analysis.

## 5.7.1 Event-Space Results: Nearest Neighbor Method

### 5.7.1.1 Real World Sensor Environment Results

The comparison metric for Nearest Neighbors is best expressed as a Euclidian distance between individual elements of different vectors. After processing with SAX, the data are represented symbolically as letters on the domain  $\{a, b \dots n\}$  where  $n$  is the cardinality of the data. In Equation (99), two vectors  $X$  and  $Y$  of the same length can be compared using the difference magnitude through a piecewise comparison:

$$DIST^2 = \sum_{i=1}^k (X_k - Y_k)^2 \quad (99)$$

where the difference between units is the distance between letters. For example, the distance between the vectors “hello” and “world” would be

$$DIST_{hello|world} = \sqrt{(h-w)^2 + (e-o)^2 + (l-r)^2 + (l-l)^2 + (o-d)^2} \quad (100)$$

In this way, two days (or any length of time) can be compared holistically to determine distance from each other. In order to classify anomalous data, a training set is specified from days which contain known normal activity, and each day is assigned a position in the training set for comparison. Candidate days are compared to each day of the training set; candidate days that are farthest from their nearest neighbor in the training set are considered “anomalous”.

Figure 58 shows arrival activity for all weekdays leading up to and including Thanksgiving. The right-hand plot shows the distance from nearest neighbor (Euclidian Distance), computed against all prior weekdays in the dataset. The results show that Thanksgiving Day shows a significantly different arrival rate than other weekdays.

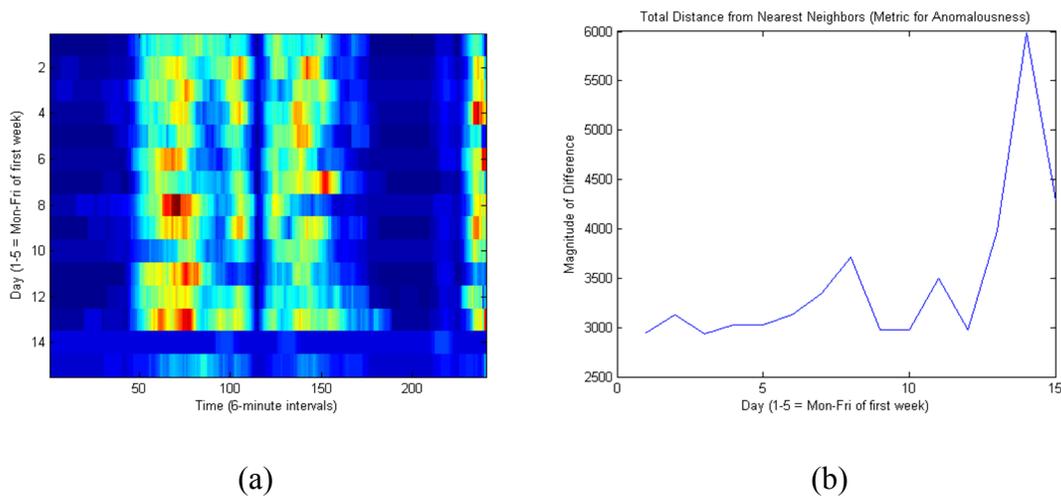


Figure 58. "Arrival" Event, Nov. 8-28, Weekdays Only (Day 14 = Thanksgiving Day)

(a) Arrival Event Convolution Results (Sensor #5 leads #16 by 12 to 18 s)

(b) Total Distance from Nearest Neighbor in Training Set (Anomalousness Metric)

Sensor 14 also exhibits a period of abnormally high activity on day 5 of the candidate period. Figure 59 shows the distance from nearest neighbor for this day to be several orders of magnitude higher than any other candidate day. This identifies Day 5 (Friday, two weeks prior to Thanksgiving) as containing an abnormal event, but also masks the effect of any other anomalies that might be present for Sensor 14 in this time period.

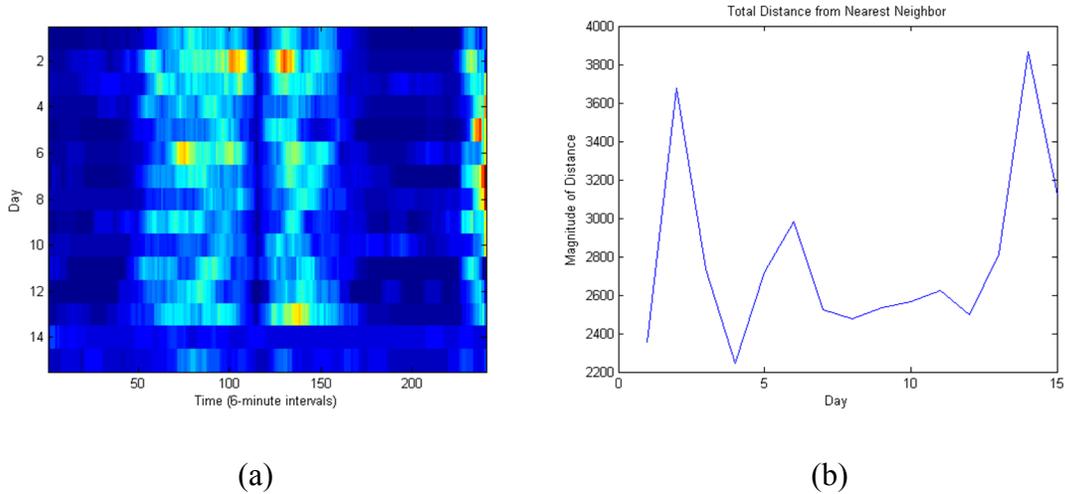


Figure 59. Double-Trigger Event, Nov 8-28, Weekdays Only

(a) Double-Trigger Event Convolution Results (Sensor #14 leads #14 by 6 to 12 s)

(b) Total Distance from Nearest Neighbor in Training Set (Anomalousness Metric)

The results show that Nearest Neighbors searches can detect large-scale anomalies such as overall changes in sensor activity, as well as spans of data which contain small-scale anomalies such as sensor saturation, in real-world experimental data.

### 5.7.1.2 Simulated Environment Results

The same Nearest Neighbors technique was applied to data from the simulated environment. Convolution analysis was performed using masks generated from the system constants (walking speed, distance, etc.), as well as physical information from GEOINT, and analyzed accordingly. Using the ground truth information provided with the simulated dataset, a

training period was selected without any anomalous activity. The Nearest Neighbor distance of day-at-a-time was then analyzed and compared against the training set. Figure 60 shows the points of high amplitude in the result, which correspond to large differences between the candidates and the "normal" set.

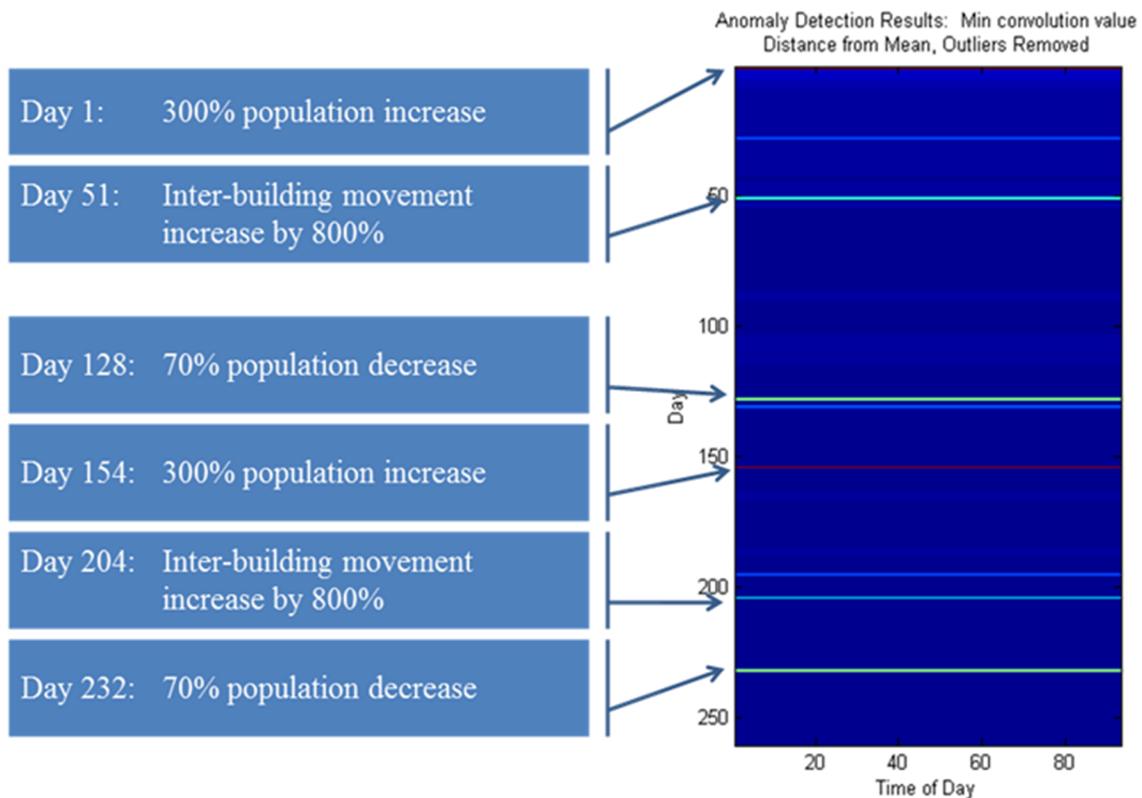


Figure 60. Convolution Analysis Results – Nearest Neighbor, Simulated Data

The six largest differences between candidate and training data correspond to the six tagged events. This method demonstrates successful detection of extended periods of increased, decreased, and modified activity in a sensor network.

### 5.7.2 Event-Space Results: Event Likelihood Model

After event masks are convolved with the parent dataset, the resulting matrix can be used to discern the likelihood of an event (corresponding to the pattern in the event mask) occurring at any given time. This is accomplished by fitting small windows of event-space time to a

distribution such as a Gaussian distribution as in Equation (84). The training dataset is broken into small increments (in this case, 30 minutes), and the governing parameters for that distribution  $P(y|\mu, \sigma)_{[t, t+30]}$  are derived. Figure 61 shows an example plot of this resulting distribution table. Each horizontal bin represents a time of day (in 30-minute increments), and the Probability Density Function (PDF) of the likelihood of an event is shown along the x-axis, normalized to a maximum value of 1. This set of PDFs enables the prediction of event occurrences at given times.

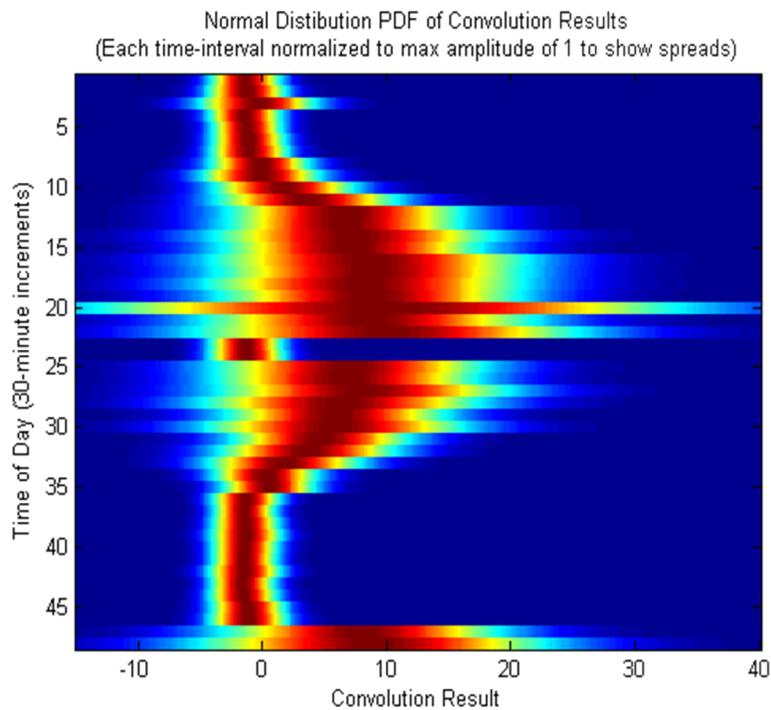


Figure 61. Example distribution of expected “arrival event” activity

The candidate datasets are analyzed using the discrete convolution technique as before, and “anomalousness” in this case is represented by the Z-score of a candidate set relative to the training distribution. For example, if the training distribution for an arrival event occurring between 0800 to 0830 hrs resulted in parameters  $\mu = 20$ ,  $\sigma = 7$  then a convolution result of 18 would have a Z-score of 0.29 as calculated by Equation (85). Because the data are windowed, a

variety of metrics can be analyzed in this way; the scope of this chapter includes first-order statistical metrics. Event bins with high-magnitude Z-scores are considered to be more anomalous.

The convolution results for a simulated "arrival" event were then analyzed as a distribution -- max value, min value, and variance were all analyzed in an effort to detect short-term anomalous behavior. Figure 62 shows an example resulting plot, with day numbers in the y-axis, time-of-day in the x-axis. Amplitude represents the distance from the mean, in terms of the standard deviation of that bin. The resulting peak represents a period of high activity for Sensor 1 (saturation), during which time the sensor is constantly triggered.

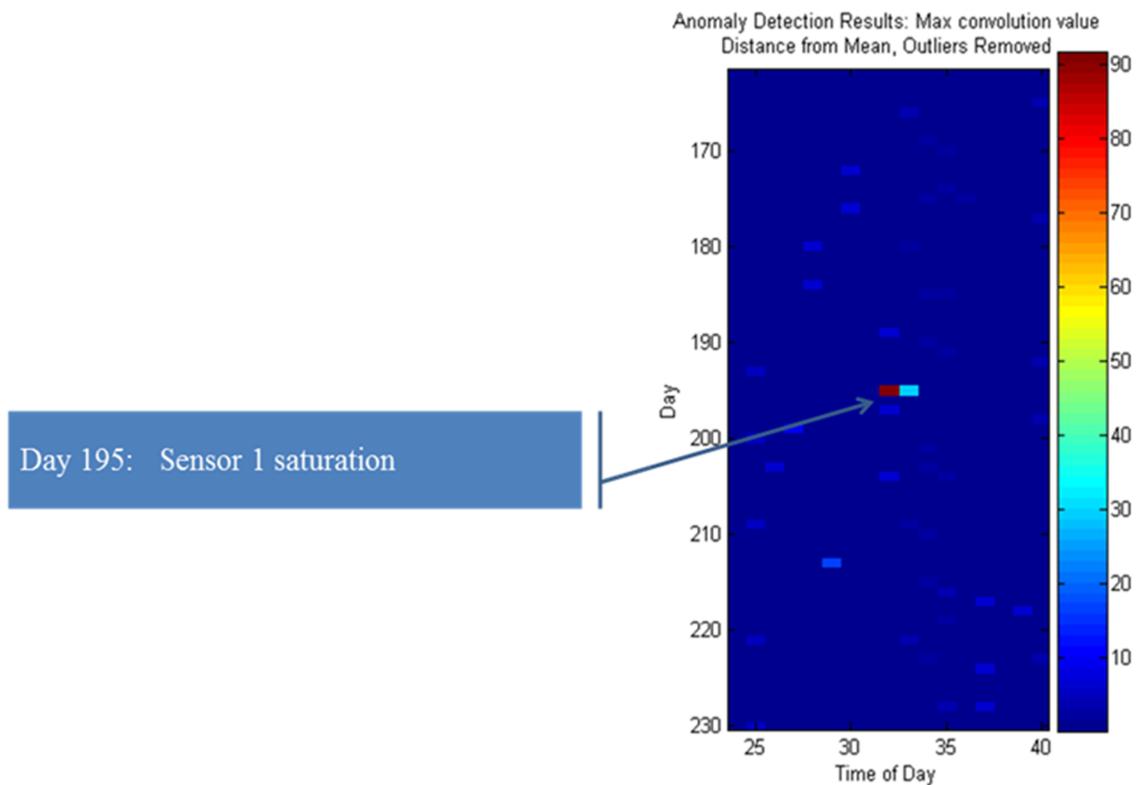


Figure 62. Short-term Anomalousness Metric for Sensor 1 vs. 2

The single discrete event not detected by this approach was the sensor drop-out (no triggering). It is possible that the spread and variability of the data caused this drop-out to appear

normal. It is not uncommon for a sensor to not trigger for several minutes at a time. Therefore, the definition of "anomalous" in this case fails to encompass an event that a human may judge to be abnormal. Further research into the classification of these events, and pattern recognition for surrounding or similar events, would be beneficial. The approach of 2D convolution analysis for spatiotemporal data yielded a high true-positive success rate of 87.5% and a low false-positive failure rate of 0.9%.

## **5.8 Conclusions**

### **5.8.1 Poisson probability data model.**

The Poisson data model and numerical scores were shown to have positive anomaly detection and classification characteristics on both simulated data and real data. This method was found to be most effective on single sensor alarms. Given that the real data is more sparse and random than the simulation, it is unclear how much the ROC performance would degrade compared to simple summation and average methods. One extension of this work would be to introduce the concept of relating neighboring time bins (minutes) for a sensor. This approach allows for the variability of an individual's (or agent's) tendency to perform an action at regular time intervals; e.g., variation in the time of arrival to work. Instead of applying the Poisson distribution to a specific time bin across a given dimension, it could be applied to overlapping windows of time.

### **5.8.2 Spatiotemporal model & discrete convolution analysis.**

The application of discrete convolution integrals to spatiotemporal data significantly reduces the space necessary to store and classify unique events. These techniques apply to sensor network data linked by GEOINT, simulated sensor network data driven by agent-based behavior, and to video data after being unwrapped and normalized using dimensionality

reduction metrics. Utilizing a combination of these techniques, a posterior statistical model can be established relating specific events (or clusters of events) to a probabilistic time-model. This purely temporal model effectively classifies event-based behavior and identifies anomalous – or statistically unlikely – behavior in the environment.

Further research is required to move this process out of the supervised-learning stages and into real time, automated anomaly detection. Overall, the approaches outlined in this chapter have proven efficacy in the classification and detection of anomalous behavior. These methods and approaches are shown to be efficient and viable metrics for analyzing sparse spatiotemporal data from multi-sensor networks, and should be extended to other fields outside those of area- and activity-monitoring.

## 6 SUMMARY AND CONCLUSIONS

This dissertation processes three distinct and heterogeneous data types, and for each type develops methods to perform automated anomaly detection. The proposed methods improve upon existing techniques for handling these data types, and combine them all into a hierarchical framework, which is itself a novel and significant advancement in the field of anomaly detection. Each data type is appropriately fitted to the hierarchical paradigm of system behavior: that data is driven by a system, which is driven by parameters, which have inherent variability, which can be quantitatively measured and monitored for the purpose of anomaly detection. The key advancements for each data type include shorter processing times, improvements to accuracy, robustness against noise, and reductions in sample size requirements.

### 6.1 State Transition Anomaly Detection

The proposed Hierarchical Transition Matrix Method (HTMM) represents a significant contribution to the field of anomaly detection in time-series data. This method treats elements of state transition matrices (TMs) as random variables. The proposed approach differs from previous work by parameterizing these random variables into a hierarchical framework with statistical rigor. Previous work assumes probability values for sparse TMs; HTMM makes no such assumption (and in fact is designed to accommodate sparse TM's). HTMM demonstrates clear advantages over existing industry-standard approaches, even those approaches which similarly rely on transition matrix analysis.

HTMM was shown to outperform comparable Nearest Neighbor analysis methods, both in synthetic and experimental data. The synthetic dataset used for HTMM benchmarking involved bistochastic transition probabilities. That is, the likelihood of each state's occurrence was equal relative to all the others, resulting in equiprobable state frequencies. This test set

represented a “worst case” scenario for data analysis: a situation in which frequency-domain analysis and state likelihood analysis would inherently fail to identify changes in system behavior. The proposed HTMM method resulted in high accuracy (99.8% TPR : 0.15% FPR) when the sample size met or exceeded  $2N^2$ , where  $N$  is the number of possible discrete system states.

Due to its computational simplicity (and ability to be performed using parallel computing), HTMM achieved these results in approximately  $2 \times 10^{-4}$  seconds per test window. This demonstrates a significant improvement in processing speed over a similarly accurate Nearest Neighbors anomaly detection algorithm. The HTMM algorithm outperformed the NN algorithm by up to two orders of magnitude. I submit that this drastic improvement in processing speed will contribute to the developing field of real-time anomaly detection, especially as it pertains to the active monitoring of mechanical system health.

To demonstrate this usefulness, HTMM was applied to experimental data from rolling element bearings. With limited learning data (80 seconds of data sampled at 12 kHz), HTMM was able to successfully and automatically identify over 99.9% of anomalous signals with less than 0.1% false positive error at a confidence interval of 99.5%. This approach also outperformed the Nearest Neighbors method both in accuracy (0.91 TPR : 0.14 FPR) and speed (10x faster). Because the final metric for anomaly detection is a statistically-defined confidence interval, not a user-defined arbitrary threshold, HTMM requires little user input. This ease of use reduces the risk of human error in determining appropriate thresholds for “normalcy”, and instead relies on rigorous statistical definitions to identify anomalous data.

In related work, transition matrices are assumed to be non-sparse; (Ye, 2000) manually reassigns zero-valued TM elements with user-defined positive probability values. This is done to

avoid errors later in the detection process (such as dividing by zero), and to ensure that every observed data sequence has a positive real-valued probability of occurrence. HTMM makes no such assumption, and instead allows for zero values in the hierarchical parameterization of TM elements. This enables HTMM to successfully identify anomalous data even when TMs are sparsely populated.

To demonstrate robustness against multiple data types, HTMM was tested against multiple stochastic datasets: data generated by autoregressive moving average models, random data generated by frequency domain sampling, and equiprobable bistochastic data. In all cases, HTMM was found to require smaller sample sizes (with better detection rates) than other similar algorithms. Overall, these state-based approaches merit application in such industries as biometric monitoring for healthcare, machine health monitoring, and signal processing in general.

## **6.2 Parameter-Driven Anomaly Detection**

The proposed Bayesian Posteriors Updated Sequentially and Hierarchically (BPUSH) also makes a significant contribution to the field of precision monitoring. This method makes use of hierarchical Bayesian statistics as well as the practice of sequential updating. Because of these key aspects, BPUSH is agglomerative: as more data becomes available, the model updates with improved estimates of system parameters. This approach also merits consideration for use in active system health monitoring, but finds specific application in the field of precision manufacturing, especially for small-scale parts such as MEMS equipment manufacturing.

In order to apply this framework to precision monitoring, BPUSH (as it is applied in this dissertation) relies on the closed-form derivation of the area between two Gamma distribution CDF's. In cases where the Gamma distribution is not an appropriate prior distribution for the

data in question, a new area metric would need to be calculated. Alternatively, computer integration of the inter-CDF area would be possible in cases of arbitrary posterior CDFs, such as when Monte Carlo / Metropolis Hastings algorithms are used to estimate the posterior. The derivation of a closed-form integral in this work merely serves to rigorously define the mathematical basis for the method, and to reduce the computational requirement of CDF comparison to a single equation.

BPUSH also significantly reduces the number of units required for robust anomaly detection. (Bartlett II, Kotrlik, & Higgins, 2001) and (Jiroutek, Muller, Kupper, & Stewart, 2003) perform similar point-change analysis on the order of 100 to 200 samples for testing. BPUSH cuts this requirement by more than half; in simulated precision manufacturing data, outperforming CUSUM / MOSUM and variance tracking analysis methods at sample sizes as small as  $n=20$ . Using sample sizes in this range yields detection rates above 98% TPR : 3% FPR. Increasing the sample size in BPUSH to 50 yielded 100% TPR : 0.1% FPR in simulated precision data.

The proposed method showed similar results when applied to experimental manufacturing data. Using wafer etch data, BPUSH was able to achieve 100% detection of anomalies (in this case, manufacturing faults) using aggregate sample sizes of  $n=20$  units, and a 94% TPR with an aggregate window size of  $n=10$ . This represents a significant advance in the field of real-time anomaly detection in cases where parts are not necessarily time-series data. The BPUSH algorithm is not dependent on time scaling, making it an excellent approach for situations where unit sampling is non-sequential or periodic (e.g., every 1000'th unit used for testing). Because BPUSH also reduces the computational burden of testing, achieving the above results at real-time speeds of 0.5ms for a 10-unit window of data. These factors make BPUSH

an excellent candidate for use in small-scale parts manufacturing and precision monitoring, as well as other applications where sampling is sparse or expensive.

### **6.3 Spatiotemporal Network Anomaly Detection**

The proposed hierarchical framework for anomaly detection is finally extended to spatiotemporal sensor networks. In this case, the application of interest is area security, where geospatial intelligence yields additional information about the area being monitored. For the experiments performed in this dissertation, agent traffic between network “nodes” (sensors) is limited to certain routes or movement paths. This enabled the simulation of an environment similar to the real-world sensor network used for experimental data testing.

For this data type, the hierarchical model for anomaly detection assumes that spatiotemporal data signals are caused by “events” in the sensor network. These events can be classified according to the type of signal they generate, and identified in the data using one of several approaches (in this case, discrete convolution kernels). Because these events occur with relative predictability over the course of hours / days / weeks, the event-space model parameterizes the likelihood of an event occurring at a given time relative to the hour of day and day of week. This hierarchical parameterization, as before, leads to a simple confidence interval test on the parent distribution that drives the “normalcy” of an event’s likelihood over time.

In this way, the method proposed by this dissertation expands on the work of (Ji, Xu, Yang, & Yu, 2010), which used discrete convolution integrals to identify events in spatiotemporal data. Unlike previous approaches, this dissertation formally parameterizes the event-space method into a separate distribution, and requires little user input to perform the final classification of anomalousness. Additionally, this method benefits from not requiring prior knowledge about anomaly types or presentations; any change from “normal” events in a

system's behavior is automatically identified. High-traffic, low-traffic, sensor failure, and changes in traffic direction are all successfully identified without any prior learning about how these anomalies present within the data. In a small-scale simulated test, 7 of 8 ground truth anomalies were successfully detected with a low false-positive rate of 0.9%. The results from this experiment demonstrate the ability of discrete convolution event space analysis to identify anomalous behavior within a real-world sensor network with very low error rates.

#### **6.4 Future Work**

In this dissertation, three data types are explored using a hierarchical framework for analysis. While this work touches on a significant portion of the data types encountered in many mechanical systems, it is important to note that other data types may be appropriate for study using this approach. For example, system metadata or technicians' notes may be quantitatively analyzed using semantic scoring. Spatiotemporal data analysis can also be extended to data types with higher dimensionality than the sensor network data explored in this dissertation. Video monitoring is a logical extension of this approach, though preprocessing may be required for noise removal and relevant signal extraction.

The approaches derived in this work are also well suited to application in fields where computational resources are limited. Most of the calculations performed for HTMM and BPUSH do not require significant computational resources, and should operate on low-power and high-efficiency platforms. By developing these analysis algorithms on smaller CPU's or programmable microcontrollers, it would be possible to create in-place solutions for system health monitoring without the requirements of expensive data acquisition and server load.

Notably, the analysis performed in BPUSH is developed and derived before any computing takes place; actual data handling only requires the use of basic math functions such as

summation, bitwise comparators (less-than, equal-to, etc.), and storage for learned data. This entire process could be deployed on small-scale platforms such as programmable microcontrollers. HTMM is slightly more computationally expensive due to the nature of matrix operations and data storage requirements, but both should be easily extensible to small-scale computing platforms.

## **6.5 Final Conclusions**

The approaches derived in this dissertation individually contribute to the fields relevant to the specific data types, and the overall hierarchical framework shows promise for extension into other fields. The state-based anomaly detection method demonstrates clear advantages over prior approaches, including reduced computing time and significant improvements to anomaly detection accuracy. Point-change anomaly detection also benefits from this work, as reducing the number of samples required for testing will reduce the overall cost of system monitoring. Performing this analysis in real-time also allows preventative maintenance and stop-loss measures to prevent or mitigate the cost of catastrophic failures, such as in rolling element bearing systems. Finally, the extension of discrete convolution and event-space analysis into a hierarchical structure for “normalcy” shows that this dissertation’s proposed framework can be extended to other data types with little modification to the basic approach. I conclude that the methods developed here are both novel and significant in their contribution to the fields of anomaly detection, mechanical engineering, data mining & analytics, and signal processing in general.

## 7 REFERENCES

- Adamec, J., & Adamec, R. (2008). *ECG Holter*. Springer.
- Allen, L. V. (2010). *Verification and Anomaly Detection for Event-Based Control of Manufacturing Systems*. Deep Blue at the University of Michigan.
- Asok, R. (2006). Symbolic Time Series Analysis via Wavelet-Based Partitioning. *Signal Processing*.
- Azoff, E. M. (1994). *Neural Network Time Series Forecasting of Financial Markets*. New York, NY: John Wiley & Sons, Inc.
- Baba, N. (2006). Utilization of NNs for Improving the Traditional Technical Analysis in the Financial Markets. *Lecture Notes in Computer Science*, 663-669.
- Barnard, G. A. (1959). Control Charts and Stochastic Processes. *Journal of the Royal Statistical Society*, 239-71.
- Bartlett II, J. E., Kotrlik, J. W., & Higgins, C. C. (2001). Organizational Research: Determining Appropriate Sample Size in Survey Research. *Information Technology, Learning, and Performance Journal*, 19(1).
- Benjamin, M. A., Rigby, R. A., & Stasinopoulos, M. (2003). Generalized Autoregressive Moving Average Models. *Journal of the American Statistical Association*, 214-223.
- Berrou, C., Glavieux, A., & Thitimajshima, P. (1993). Near Shannon limit error-correcting coding and decoding: Turbo-codes. *International Conference on Communications* (pp. 1064 - 1070). Geneva: IEEE.
- Billinger, D. R. (2001). *Time Series: Data Analysis and Theory*. Society for Industrial and Applied Mathematics.
- Billingsley, P. (1995). *Probability and Measure* (3rd ed.). New York: Wiley, Inc.

- Bin, Z. (2008). Rolling Element Bearing Feature Extraction and Anomaly Detection Based on Vibration Monitoring. *IEEE*.
- Bo, L., Zhang, Z., & Yu, P. S. (2010). *Relational Data Clustering: Models, Algorithms, and Applications*. New York: Chapman & Hall.
- Box, G., Jenkins, G. M., & Reinsel, G. C. (1994). *Time Series Analysis: Forecasting and Control* (3rd Ed. ed.). Prentice Hall.
- Brodsky, B., & Darkhovsky, B. (2000). *Nonparametric Statistical Diagnosis: Problems and Methods*. Netherlands: Kluwer Academic Publishers.
- Brogan, W. (1991). *Modern Control Theory*. Prentice Hall.
- Buhl, M., & Kennel, M. B. (2005). Statistically Relaxing to Generating Partitions for Observed Time Series Data. *Physical Review*.
- Burgess, M. (2002). Two Dimensional Time-Series for Anomaly Detection and Regulation in Adaptive Systems. *DSOM*, 169-180.
- Cappellini, V. (2009). Random Bistochastic Matrices. *Journal of Physics A: Mathematical and Theoretical*.
- Cassisi, C., Montalto, P., Aliotta, M., Cannata, A., & Pulvirenti, A. (2012). Similarity Measures and Dimensionality Reduction Techniques for Time Series Data Mining. *Advances in Data Mining Knowledge Discovery and Applications*, 71-96.
- Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly Detection: A Survey. *ACM Computing Surveys (CSUR)*, 41(3).
- Chen, D. (2008). A Novel Method for Network Anomaly Detection Using Superstatistics. *CISIS* (pp. 595-598). IEEE.

- Chen, S. L. (2010). Anomaly Detection of the Tapered Roller Bearing with Statistical Data-Driven Approaches. *Insight*.
- Cheyner, A., & Julia, L. (1998). MVIEW: Multimodal Tools for the Video Analyst. *Intelligent User Interfaces (IUI '98)* (pp. 55-62). ACM.
- Chin, S. C., Ray, A., & Rajagopalan, V. (2004). Symbolic Time Series Analysis for Anomaly Detection: A Comparative Evaluation. *Signal Processing*.
- Chiu, B., Lonardi, S., & Keough, E. (2003). Probabilistic Discovery of Time Series Motifs. *ICKDDM* (pp. 493-498). Washington, DC: ACM.
- Chu, C.-S. J., Stinchcombe, M., & White, H. (1996). Monitoring Structural Change. *Econometrica*, 64(5), 1045-1065.
- Chuah, M. C., & Fu, F. (2007). *ECG Anomaly Detection via Time Series Analysis*. Lehigh University, Department of Computer Science & Engineering.
- Das, M., & Parthasarathy, S. (2009). Anomaly Detection and Spatio-Temporal Analysis of Global Climate System. *SensorKDD '09* (pp. 142-150). ACM.
- Davies, E. (2005). *Machine Vision: Theory, Algorithms, Practicalities* (2nd ed.). New York, New York: Morgan Kaufmann Publishers.
- Dell, R., Holleran, S., & Ramakrishnan, R. (2002). Sample Size Determination. *ILAR*, 43(4), 207-213.
- Demmel, J. W. (1997). *Applied Numerical Linear Algebra*. SIAM.
- Devore, J. L. (1995). *Probability and Statistics for Engineering and the Sciences*. Pacific Grove: Duxbury.
- Draney, R. K. (2007). High Temperature Sensor for Bearing Health Monitoring. *IEEE*.

- Du, R., Elbestawi, M., & Wu, S. (1995). Automated Monitoring of Manufacturing Processes, Part 1: Monitoring Methods. *Journal of Engineering for Industry*, 117, 121-132.
- Duda, R. O., Hart, P. E., & Stork, D. G. (2001). *Pattern Classification* (2nd ed.). New York, NY, United States: John Wiley and Sons.
- Eftekharnjad, B., Carrasco, M., Charnley, B., & Mba, D. (2011). The Application of Spectral Kurtosis on Acoustic Emission and Vibrations from a Defective Bearing. *Mechanical Systems and Signal Processing*, 25(1), 266-284.
- Ericsson, e. a. (2003). Towards Automatic Detection of Local Bearing Defects in Rotating Machines. *Mechanical Systems and Signal Processing*.
- Estivill-Castro, V. (2002). Why so many clustering algorithms? *ACM SIGKDD Explorations Newsletter*, 4, p. 65.
- Everitt, B. S. (2002). *The Cambridge Dictionary of Statistics*. CUP.
- Friedman, N., & Goldszmidt, M. (1997). Sequential Update of Bayesian Network Structure. *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufman Publishers, Inc.
- Froidevaux, R. (2004). Sequential Updating Simulation. *Geostatistics for Environmental Applications*, 13(5), 307-318.
- Gill, J. (2009). *Bayesian Methods: A Social and Behavioral Sciences Approach* (2nd ed.). Boca Raton: Taylor and Fancis Group.
- Gipps, P. G., & Marksjo, B. (1985). A Micro-Simulation Model for Pedestrian Flows. *Mathematics and Computers in Simulation*, 27(2-3), 95-105.

- Giurgiutiu, V., & Kropas-Hughes, C. (2002). Comparative Study of Neural-Network Damage Detection from a Statistical Set of Electro-Mechanical Impedance Spectra. *Symposium on Smart Structures and Materials*. San Diego, CA: SPIE.
- Gonzalez, R. C., & Woods, R. E. (2008). *Digital Image Processing* (3rd ed.). Upper Saddle River, New Jersey, United States: Pearson Education, Inc.
- Gonzalez, R., & Woods, R. (1992). *Digital Image Processing (1st Ed.)*. Addison-Wesley.
- Gupda, S., Ray, A., & Keller, E. (2007). Symbolic Time Series Analysis of Ultrasonic Data for Early Detection of Fatigue Damage. *Mechanical Systems and Signal Processing*, 866-884.
- Hamburg, M. (1985). *Basic Statistics: A Modern Approach* (3rd ed.). Harcourt Brace Jovanovich.
- Hand, D. J. (2009). Measuring Classifier Performance: A Coherent Alternative to the Area Under the ROC Curve. *Machine Learning*, 77, 103-123.
- Hanley, J. A., & McNeil, B. J. (1982). The Meaning and Use of the Area Under a Receiver Operating Characteristic (ROC) Curve. *Radiology*, 29-36.
- Ho, S. L., Xie, M., & Goh, T. N. (2002). A Comparative Study of Neural Network and Box-Jenkins ARIMA Modeling in Time Series Prediction. *Computers and Industrial Engineering*, 42, 371-375.
- Hou, T.-H., Liu, W.-L., & Lin, L. (2003). Intelligent Remote Monitoring and Diagnosis of Manufacturing Process using an Integrated Approach of Neural Networks and Rough Sets. *Journal of Intelligent Manufacturing*, 14, 239-253.
- Hsu, C.-C. (2007). The MOSUM of Squares Test for Monitoring Variance Changes. *Finance Research Letters*, 4, 254-260.

- Hu, X., Subbu, R., Bonissone, P., Qiu, H., & Iyer, N. (2008). Multivariate Anomaly Detection in Real-World Industrial Systems. *Neural Networks*, 2766-2771.
- Huang, C., & Darwiche, A. (1996). Inference in Belief Networks: A Procedural Guide. *International Journal of Approximate Reasoning*, 15(3), 225-263.
- Ide, T., & Kashima, H. (2004). Eigenspace-based Anomaly Detection in Computer Systems. *KDD* (pp. 440-449). Seattle, Washington, USA: ACM.
- International Organization for Standardization. (2009). *10816-3: Evaluation of Machine Vibration by Measurements on Non-Rotating Parts*. International Organization for Standardization.
- Ji, S., Xu, W., Yang, M., & Yu, K. (2010). 3D Convolutional Neural Networks for Human Action Recognition. *International Conference for Machine Learning*.
- Jiroutek, M. R., Muller, K. E., Kupper, L. L., & Stewart, P. W. (2003). A New Method for Choosing Sample Size for Confidence Interval-Based Inferences. *Biometrics*, 59(3), 580-590.
- Jordan, M. I. (2010). The Conjugate Prior for the Normal Distribution. Berkeley, CA: UCA, Berkeley. Retrieved from <http://www.cs.berkeley.edu/~jordan/courses/260-spring10/lectures/lecture5.pdf>
- Karlin, S. (1998). *An Introduction to Stochastic Modeling* (3rd ed.). Academic Press.
- Kasetty, S. (2008). Real-Time Classification of Streaming Sensor Data. *ICTAI* (pp. 149-156). Dayton, OH: IEEE.
- Keogh, E. (May, 2001). Locally Adaptive Dimensionality Reduction for Indexing Large Time-Series Databases. *ACM SIGMOD*. Santa Barbara, California.

- Keogh, E., Lin, J., & Fu, A. (2005). HOT SAX: Finding the Most Unusual Time Series Subsequence: Algorithms and Applications. *ICDM*.
- Kfir, H., & Kanter, I. (2003). Parallel Versus Sequential Updating for Belief Propagation Decoding. *Physica A: Statistical Mechanics and its Applications*, 330(1-2), 259-270.
- Khatkhate, A., Gupta, S., Ray, A., & Patankar, R. (2008). Anomaly Detection in Flexible Mechanical Couplings via Time Series Analysis. *Journal of Sound and Vibration*, 311, 608-622.
- Khatkhate, A., Ray, A., Keller, E., Gupta, S., & Chin, S. (2006). Symbolic Time-Series Analysis for Anomaly Detection in Mechanical Systems. *Mechatronics*, 11(4), 439-447.
- Kim, M., Kim, T., Shin, Y., & Lam, S. (August, 2004). A wavelet-based approach to detect shared congestion. *Proceedings of ACM SIGCOMM*. Portland, OR.
- Kong, W., Katahira, N., Watanabe, M., Katayama, T., Hisazumi, K., & Fukuda, A. (2011). Formal Verification of Software Designs in Hierarchical State Transition Matrix with SMT-based Bounded Model Checking. *18th Asia-Pacific Software Engineering Conference* (pp. 81-88). IEEE.
- Kumar, K., & Tandon, N. (2003). Detection of Defects at Different Locations in Ball Bearings by Vibration and Shock Pulse Monitoring. *Noise and Vibration Worldwide*.
- Kumar, N., Lolla, N., Keough, E., Lonardi, S., & Ratanamahatana, C. A. (2005). Time-series Bitmaps: A Practical Visualization Tool for working with Large Time Series Databases. *SIAM International Conference on Data Mining*, (pp. 531-535). Newport Beach, CA.
- Lacey, S. J. (2007). Using Vibration Analysis to Detect Early Failure of Bearings. *Insight*.
- Lacey, S. J. (2008). An Overview of Bearing Analysis. *Maintenance and Asset Management*.

- Lee, D., Hwang, I., Valente, C., Oliveria, J., & Dornfeld, D. (2006). Precision Manufacturing Process Monitoring with Acoustic Emission. *International Journal of Machine Tools & Manufacture*, 46, 176-188.
- Li, W., Yue, H., Valle-Cervantes, S., & Qin, S. (2000). Recursive PCA for Adaptive Process Monitorin. *Journal of Process Control*, 10, 471-486.
- Lin, J. (2003). A Symbolic Representation of Time Series with Implications for Streaming Algorithms. *ACM*.
- Lin, J., Wei, L., Lonardi, S., & Keough, E. (2007). Experiencing SAX: a Novel Symbolic Representation of Time Series. *DMKD*.
- Lucas, B., & Kanade, T. (1981). An Iterative Image Registration Technique with an Application to Stereo Vision. *Proceedings of Imaging Understanding*, (pp. 121-130).
- MacQueen, J. (1967). Some Methods for Classification and Analysis of Multivariate Observations. *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*. 1, pp. 281-297. Berkeley, CA: University of California Press.
- Mallat, S. (2008). *A Wavelet Tour of Signal Processing*. Academic Press.
- Monostori, L., & Prohaszka, J. (1993). A Step Towards Intelligent Manufacturing Modelling and Monitoring of Manufacturing Processes through Artificial Neural Networks. *CIRP Annals - Manufacturing Technology*, 42(1), 485-488.
- Murphy, K. P. (2007). Conjugate Bayesian Analysis of the Gaussian Distribution.
- Nakashima, H., & al., e. (2009). *Handbook of Ambient Intelligence and Smart Environments*. New York: Springer Publishing.
- Ng, H.-F. (2006). Automatic Thresholding for Defect Detection. *Pattern Recognition Letters*.

- Ott, L. R., & Longnecker, M. (2001). *An Introduction to Statistical Methods and Data Analysis*. Pacific Grove, CA, United States: Duxbury / Thomson Learning.
- Page, E. S. (1954). Continuous Inspection Scheme. *Biometrika*, 41(1), 100-115.
- Page, E. S. (1963). *Controlling the Standard Deviation by CUSUMS and Warning Lines*. Chapel Hill, N. C.: University of North Carolina, Department of Statistics.
- Panangadan, A. (2004). Detecting anomalous human interactions using laser range-finders. *International Conference on Intelligent Robots and Systems* (pp. 2136-2141). IEEE.
- Paschalidis, I., & Smaragdakis, G. (2009). Spatio-Temporal Network Anomaly Detection by Assessing Deviations of Empirical Measures. *IEEE/ACM Transactions on Networking*.
- Patcha, A., & Park, J.-M. (2007). An Overview of Anomaly Detection Techniques: Existing Solutions and Latest Technological Trends. *Computer Networks*, 51, 3448-3470.
- Patel, P., Keogh, E., Lin, J., & Lonardi, S. (2002). Finding Motifs in Massive Time Series Databases. *Proceedings of the 2002 IEEE International Conference on Data Mining*. Maebashi City, Japan: IEEE.
- Patil, G. P., & Taillie, C. (2001, March). A Multiscale Hierarchical Markov Transition Matrix Model for Generating and Analyzing Thematic Raster Maps. *Environmental and Ecological Statistics*, 8(1), 71-84.
- Paxson, V. (1999). Bro: a system for detecting network intruders in real-time. *Computer Networks*, 31(23-24), 2435-2463.
- Phan, V. D., & Wang, X. (2002). Bayesian Turbo Multiuser Detection for Nonlinearly Modulated CDMA. *Signal Processing*, 82(1), 43-68.
- Qin, S. (2003). Statistical Process Monitoring: Basics and Beyond. *Journal of Chemometrics*, 17, 480-502.

- Rajagopalan, e. a. (1997). Anomalous ENSO Occurrences: An Alternate View. *Journal of Climate*, 2351-2357.
- Rajagopalan, V., & Ray, A. (2006). Symbolic Time Series Analysis via Wavelet-Based Partitioning. *Signal Processing*, 86(11), 3309-3320.
- Rao, C., & Ray, A. (2009). Review and Comparative Evaluation of Symbolic Dynamic Filtering for Detection of Anomaly Patterns. *Journal of Signal, Image and Video PProcessing*, 101-114.
- Ray, A. (2004). Symbolic Dynamic Analysis of Complex Systems for Anomaly Detection. *Signal Processing*, 84, 1115-1130.
- Ribiero, B. (2007). Support, Relevance and Spectral Learning for Time Series. *Lecture Notes in Computer Science*, 4432, 199-207.
- Ricchetti, E. (2000). Multispectral Satellite Image and Ancillary Data Integration for Geological Classification. *Photogrammetric Engineering & Remote Sensing*, 429-435.
- Roesch, M. (1999). Snort - lightweight intrusion detection for networks. *Proceedings of the 13th USENIX conference on System administration* (pp. 229-238). Seattle, WA: LISA.
- Rubenstein, D., Kurose, J., & Towsley, D. (June, 2000). Detecting shared congestion of flows via end-to-end measurement. *Proceedings of ACM SIGMETRICS*.
- Russell, S., & Norvig, P. (2003). *Artificial Intelligence: A Modern Approach* (2nd ed.). Upper Saddle River, NJ, United States: Pearson Education, Inc.
- Sadettin, e. a. (2005). Vibration Monitoring for Defect Diagnosis of Rolling Element Bearings as a Predictive Maintenance Tool. *NDT&E International*.
- Sankar, P. K., & Mitra, P. (2004). *Pattern Recognition: Algorithms for Data Mining*. New York, NY, USA: Chapman Hall / CRC Press.

- Sanso, B., & Schmidt, A. M. (2004). *Spatio-Temporal Models Based on Discrete Convolutions*. American Mathematical Society.
- Sant Anna, A., & Wickstrom, N. (2011). Symbolization of Time-Series: An Evaluation of SAX, Persist, and ACA. *IEEE*.
- Shlens, J. (2005). *A Tutorial on Principal Component Analysis (2nd Ver.)*. La Jolla, CA, 92093: Institute for Nonlinear Science, University of California, San Diego.
- Singla, G., & al., e. (2010). Recognizing Independent and Joint Activities Among Multiple Residents in Smart Environments. *J. Ambient Intelligent Human Computing, 1*, pp. 57-63.
- Snell, R. S. (2006). *Clinical Neuroanatomy* (6th ed.). New York, United States: Lippincott Williams & Wilkins.
- Sohn, H., Law, & Kincho, L. H. (1998). A Bayesian Probabilistic Approach for Structure Damage Detection. *Earthquake Engineering and Structural Dynamics, 26*, 1259-1281.
- Spiegelhalter, D. J., & Lauritzen, S. L. (1990). Sequential Updating of Conditional Probabilities on Directed Graphical Structures. *Networks, 20*(5), 579-605.
- STI Vibration Monitoring, Inc. (2009, 02 01). *Tech Note 117: Interpretation of the International Standard ISO 10816-3*. Retrieved 03 02, 2013, from STI : <http://www.stiweb.com/TechNote117.html>
- Sunhil, K. K., & Desai, U. B. (2001). *A Bayesian Approach to Image Interpretation*. Norwell, Massachusetts: Kluwer Academic Publishers.
- Tan, P.-N., Kumar, V., & Steinback, M. (2005). *Introduction to Data Mining*. Addison-Wesley.
- Thode, H. C. (2002). *Testing for Normality*. New York: Marcel Dekker, Inc.
- Trefethen, L. N., & Bau, D. (1997). *Numerical Linear Algebra*. SIAM.

- Trevor, H., & Tibshirani, R. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (2nd Ed. ed.).
- Tzanetakis, G. (2001). Audio Analysis using the Discrete Wavelet Transform. *Transform*, 318-323.
- Varun, e. a. (2009). Anomaly Detection: A Survey. *ACM Computing Surveys*.
- Wang, H., Zhang, D., & Shin, K. G. (2004). Change-point monitoring for the detection of DoS attack. *IEEE Transactions on Dependable and Secure Computing*, 1(4), 193-208.
- Wang, X. (2001). Blind Turbo Equalization in Gaussian and Impulsive Noise. *IEEE Transactions on Vehicular Technology*, 50(4), 1092-1105.
- Wang, X., Chen, R., & Liu, J. S. (2002). Monte Carlo Bayesian Signal Processing for Wireless Communications. *Journal of VLSI Signal Processing*, 30, 89-105.
- Weber, R., Schek, H. J., & Blott, S. (1998). A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces. *VLDB*, 24. New York.
- Winston, W. (1994). *Operations Research: Applications and Algorithms*. Belmont, CA: Duxbury Press.
- Wise, B., Gallagher, N., Butler, S., White, D., & Barna, G. (1999). A Comparison of Principal Components Analysis, Multi-way Principal Components Analysis, Tri-Linear Decomposition and Parallel Factor Analysis for Fault Detection in a Semiconductor Etch Process. *Journal of Chemometrics*, 13, 379-396.
- Wren, C., & al, e. (2006). Similarity-based Analysis for Large Networks of Ultra-Low Resolution Sensors. *Pattern Recognition*, 39.
- Xin, J. (2011). Anomaly Detection in Nuclear Power Plants via Symbolic Dynamic Filtering. *IEEE Transactions on Nuclear Science*, 277-288.

- Yashchin, E. (1994). Monitoring Variance Components. *Technometrics*, 36(4), 379-393.
- Ye, N. (2000). A Markov Chain Model of Temporal Behavior for Anomaly Detection. *SMCIAC*. 166, pp. 171-174. Oakland: IEEE.
- Yegneswaran, V., Griffin, J. T., Barford, P., & Jha, S. (2005). An Architecture for Generating Semantics-Aware Signatures. *USENIX Security Symposium*, (pp. 97-112). Baltimore, MD.
- Yi, B.-K., & Faloustos, C. (2000). Fast Time Sequence Indexing for Arbitrary LP Norms. *Very Large Data Base (VLDB) Conference*, (pp. 385-394). Cairo, Egypt.
- Yuan, Y., Lee, E. K., Pompili, D., & Liao, J. (2011). Thermal Anomaly Detection in Datacenters. (I. o. Engineers, Ed.) *Journal of Mechanical Engineering Science*.
- Zhang, B., Sconyers, C., Byington, C., Patrick, R., Orchard, M., & Vachtsevanos, G. (2008). Anomaly Detection: A Robust Approach to Detection of Unanticipated Faults. *International Conference on Prognostics and Health Management*. IEEE.
- Zhang, S., & Turk, M. (2008). Eigenfaces. 3. Retrieved 11 5, 2012, from <http://www.scholarpedia.org/article/Eigenfaces>

## APPENDIX A: FULL DERIVATION OF GAMMA POSTERIOR

Assume that some sample data is a subset of a population, and data are normally distributed:

$$\tilde{y} = [y_1, y_2, \dots, y_n] \in y \quad \tilde{y} \sim N(\mu, \phi^{-1}), \quad \phi \sim \text{Gam}[\alpha, \beta], \quad \mu \in C$$

Applying this to Bayes theorem:

$$P(\phi | [y_1, y_2, \dots, y_n]) = \frac{P([y_1, y_2, \dots, y_n] | \phi)P(\phi)}{P([y_1, y_2, \dots, y_n])}$$

Note that the denominator yields a constant through the total law of probability:

$$\int [P(\tilde{y} | \phi)P(\phi)]d\phi \equiv C$$

$$P(\phi | \tilde{y}) \propto P(\tilde{y} | \phi)P(\phi)$$

Inferring  $y$  from  $\tilde{y}$  requires the likelihood function:

$$LH[y | \phi, \tilde{y}]$$

$$P(\phi | y) \propto \prod_{i=1}^n [f(y_i | \phi, \mu)]P(\phi)$$

$$P(\phi | y) \propto \prod_{i=1}^n \left[ \left( \frac{\phi}{2\pi} \right)^{\frac{1}{2}} e^{-\frac{\phi}{2}(y_i - \mu)^2} \right] \frac{\beta^\alpha}{\Gamma[\alpha]} \phi^{\alpha-1} e^{-\beta\phi}$$

Solving the discrete product:

$$P(\phi | y) \propto \left( \frac{\phi}{2\pi} \right)^{\frac{n}{2}} e^{-\frac{\phi}{2} \sum_{i=1}^n (y_i - \mu)^2} \frac{\beta^\alpha}{\Gamma[\alpha]} \phi^{\alpha-1} e^{-\beta\phi}$$

$$P(\phi | y) \propto \phi^{\frac{n}{2} + \alpha - 1} e^{-\phi \left( \frac{1}{2} \sum_{i=1}^n (y_i - \mu)^2 \right)}$$

Recognizing that this result is proportional to the Gamma distribution PDF:

$$P(\phi | y) \propto \phi^{A-1} e^{-\phi B} \propto \frac{B^A}{\Gamma[A]} \phi^{A-1} e^{-\phi B} = \text{Gam}[\phi | A, B]$$

$$\phi \sim \text{Gam}[A, B] \text{ where } A = \alpha + \frac{n}{2} \text{ and } B = \beta + \frac{1}{2} \sum_{i=1}^n (y_i - \mu)^2$$

## APPENDIX B: FULL DERIVATION OF DIFFERENCE METRIC $w$

Below are the identities of the lower, upper, and complete Gamma functions:

$$\gamma(a, bx) + \Gamma(a, bx) = \Gamma(a)$$

$$\Gamma(a, bx) = \int_{bx}^{\infty} t^{a-1} e^{-t} dt$$

$$\gamma(a, bx) = \int_0^{bx} t^{a-1} e^{-t} dt$$

Noting that the integral in the upper-incomplete Gamma function is lower-bounded by  $b$ , it follows that two upper-incomplete gamma functions that share the same  $a$  must never intersect for all identical values of  $x$ :

$$\Gamma(a, b_1 x) \leq \Gamma(a, b_2 x) \quad \text{where } b_1 > b_2$$

As two such Gamma functions must never intersect, the area between the two Gamma distribution CDF's becomes a simple integral subtraction. The CDF of the Gamma distribution is substituted into the area integral as follows:

$$CDF = \frac{\gamma(A, B\phi)}{\Gamma(A)}$$

$$w = \int_0^{\infty} \left[ \frac{\gamma(A, B_1\phi)}{\Gamma(A)} - \frac{\gamma(A, B_2\phi)}{\Gamma(A)} \right] d\phi$$

$$w = \int_0^{\infty} \left[ \left( 1 - \frac{\Gamma(A, B_1\phi)}{\Gamma(A)} \right) - \left( 1 - \frac{\Gamma(A, B_2\phi)}{\Gamma(A)} \right) \right] d\phi$$

$$w = \frac{1}{\Gamma(A)} \int_0^{\infty} [\Gamma(A, B_2\phi) - \Gamma(A, B_1\phi)] d\phi$$

From <http://functions.wolfram.com/06.06.21.0001.01>, the integral of the upper-incomplete Gamma function with a coefficient term on the parameter  $\phi$  is:

$$\int \Gamma(a, bx) dx = x\Gamma(a, bx) - \frac{\Gamma(a+1, bx)}{b}$$

Integrating and reducing yields:

$$w = \frac{1}{\Gamma(A)} \left[ \left( \phi\Gamma(A, B_2\phi) - \frac{1}{B_2}\Gamma(A+1, B_2\phi) \right) - \left( \phi\Gamma(A, B_1\phi) - \frac{1}{B_1}\Gamma(A+1, B_1\phi) \right) \right] \Bigg|_0^\infty$$

$$w = \frac{1}{\Gamma(A)} \left[ \frac{1}{B_2}\Gamma(A+1) - \frac{1}{B_1}\Gamma(A+1) \right]$$

$$w = \frac{\Gamma(A+1)}{\Gamma(A)} \left[ \frac{1}{B_2} - \frac{1}{B_1} \right]$$

Finally, note that the Gamma function is equal to the factorial of a positive integer  $n$  such that:

$$\Gamma(n) = (n-1)! \text{ where } n \in \{1, 2, \dots\}$$

This applies to the reduction of the coefficient terms such that:

$$\frac{\Gamma(n+1)}{\Gamma(n)} = \frac{(n-1+1)!}{(n-1)!} = \frac{(n)!}{(n-1)!} = n$$

This yields the final area difference between two Gamma distribution curves which share the same parameter  $A$ :

$$w = A \left( \frac{1}{B_2} - \frac{1}{B_1} \right)$$

## APPENDIX C: ANNOTATED LIST OF FIGURES

Figure 1. Example of a point anomaly when tracking multiple dimensions 4

Illustrates a spatial anomaly in 3 dimensions. This figure specifically shows an application of anomaly detection to the elements of a typical 2x2 covariance matrix (where the off-diagonal elements are identical). These point anomalies are commonly considered "outliers".

Figure 2. Example contextual anomaly in an ECG pattern. 4

Shows a contextual anomaly in an Electro-Cardiogram (ECG) obtained from (Chuah & Fu, 2007). This figure shows data that may not exceed thresholds for detection in previous approaches. These types of anomalies are typically brief with respect to time, and distinct from surrounding data, but differ from point anomalies in that they are not necessarily outliers.

Figure 3. ROC Curve Axis and Quadrant Labeling 8

Shows Receiver Operating Characteristic (ROC) curve layout. ROC curves are typical measures for assessing detector performance by comparing the rate of correct detection (y-axis) to the rate of false positive (x-axis). Performance rates above the Random Selector line are considered "positive" detection.

Figure 4. Reduction of time-series data into symbolic bins. 9

Shows the generalized application of Symbolic Aggregate Approximation (SAX) to time-series data. In SAX, time-series data are aggregated into fixed window lengths and assigned symbols from equiprobable bins of a Gaussian distribution. This amplitude distribution is shown on the y-axis; in this figure, an alphabet of 3 letters was used for dimensionality reduction.

Figure 5. Markov chain with 5 discrete states (a), and corresponding transition matrix (b) 20

Shows a Markov chain and corresponding transition matrix for a 5-state system. The Markov chain (left, a) is a graphical representation of the probabilistic progression between system states. The transition matrix (right, b) combines the transition probabilities between states into row-normalized matrix format.

Figure 6. Cumulative Density Function for determining MCMC transition probability. 22

Shows the CDF used for determining MCMC transition probability based on row 1 of the TM used in Figure 6. Random numbers drawn between 0 and 1 determine the system's subsequent state. In this simplified example, the system may only proceed to state S1 or S2 given that the previous state is S1; the transition probabilities from Figure 6 (b) correspond to the y-axis of the CDF.

Figure 7. Transition Matrix for predator-prey scenario with two absorbing states 23

An example TM with two absorbing states. An absorbing state is one from which the system can never depart once it is reached. The example system used for this figure is a predator-prey relationship. If the system state (distance) between predator and prey is ever 0 (S1), the system remains there forever. The approaches in this work are not intended for use on TMs with absorbing states.

Figure 8. Flowchart of Sequential Updating & Parameter Refinement 24

A data flow chart for a basic Sequential Updating algorithm. In this flowchart, the initial prior is typically an uninformative or flat prior. The posterior distribution is calculated, and used as a new prior distribution when more data is added. Assuming that the data come from a consistent source, the estimation of the posterior distribution becomes more refined over time.

Figure 10. Basic diagram of a physical neuron (a) and an analogous artificial neuron (b) 28

Compares a physical neuron (left, a) to an artificial neuron (right, b). Elements of the physical neuron (dendrites, axon, and nucleus) are approximated by mathematical elements of the artificial neuron (input weights, output, and comparator respectively).

Figure 10. Example Feed-Forward Artificial Neural Network 29

Shows an ANN with a hidden layer. In a feed-forward ANN, input weights are passed to multiple neurons which comprise a layer distinct from the output layer. This facilitates more complex decision-making by the overall ANN.

Figure 11. Digits 0 through 9 as 5x5 image arrays 31

An example of images for digital numerical text recognition. In this example, the digits 0 through 9 are represented by 5x5-pixel arrays. These 25-element arrays could be analyzed by a neural network to determine if a digit was present, and if so, which one.

Figure 12. An anomaly letter "A" relative to known pattern "8" 31

Shows how a digital numeral ("8") may differ from a distinct pattern for a letter (capital "A"). For the above example, this may result in false detection when using a neural network trained to recognize digits.

Figure 13. An example Bayesian hierarchical model for mean parameters with fixed variance 32

Shows a hierarchy of parameterization for various Bayesian posterior/prior combinations. In this example, the posterior for the mean of a Gaussian distribution is also a Gaussian distribution, which has a mean parameter which may also be estimated using a Gaussian distribution, and so on.

Figure 14. Example transition matrix probability calculation scheme 36

Shows how a transition matrix can be used to calculate the discrete probability of a series of system states. For this example, the sequence of states is {1 2 4 1 3 5 3 4 2}.

Transition  $i=1$  is {1,2},  $i=2$  is {2,4}, and so on. Multiplying TM elements at these indexes results in the discrete probability of the sequence of states.

Figure 15. D-Markov transition matrix where  $D=2$ ,  $N=2$  37

An example of a higher-order transition matrix. In a D-Markov TM, the number of previous states  $D$  for a subsequent state determines the order of the TM. To reduce this TM to two dimensions, some methods define a "state" in the 2D matrix as a sequence of states from the D-Markov matrix.

Figure 16. Nested hierarchy of pixels. 39

Shows a previous approach to transition matrices. One such method from (Patil & Taillie, 2001) assumes a hierarchical model for TM element values such. This model assumes that a parent pixel at hierarchy level  $\gamma_n$  governs multiple sub-pixel values at hierarchy level  $\gamma_{n+1}$

Figure 17. Example ball bearing assembly, cross-section 48

Shows the various components of a typical ball bearing assembly in cross-section view. The primary elements of this type of bearing assembly are the ball (rolling element), outer and inner raceways (contact surfaces for the rotating machinery), and the cage (holds the rolling elements in place as they rotate).

Figure 18. ISO Velocity Chart from <http://www.stiweb.com/TechNote117.html> 50

An ISO damage/risk chart for use in analyzing the vibration of rotating machinery under various operating conditions. This type of damage assessment relies on measuring and thresholding the amplitude of system vibration. Small faults may not cause significant

changes in vibration amplitude, but can propagate and cause larger faults and catastrophic failure.

Figure 19. Representation of symbolic bin assignment in SAX 55

Shows how SAX may be applied to rolling element bearing vibration signature data. In this case, the SAX window size is 0.01 seconds and the signal is approximated using an alphabet length of 6 letters.

Figure 20. Markov Chain with transition probabilities for a three-state system 57

Shows a Markov Chain for a 3-state system. As opposed to the Markov Chain in Figure 6, in this system transitions are possible between all states. That is, there would be no zero-valued elements in the TM for this Markov chain.

Figure 21. Example Transition Matrix for a 3-state System 58

Shows an example 3x3 TM for a 3-state system such as the one shown in Figure 22. This example TM has no zero-valued elements and has been row-normalized such that the sum of all probabilities for each row is 1.

Figure 22. Hierarchical Parameterization Structure for Anomaly Detection 60

Illustrates the hierarchical structure of the time-series approach developed in this work. At the bottom levels, time-series data is modeled using Markov Chains and TMs; at higher levels, the TMs are parameterized and compared to all other training data to determine a range of "normal" values for TMs. At the top layer distribution, TMs with Z-scores that fall outside a range (confidence interval) of these "normal" values are automatically classified as being anomalous.

Figure 23 (a)-(h). Several examples of sparse Transition Matrices from normal data 61

Shows several TMs generated from small windows of time-series data. Because the windows used to generate these TMs are on the order of  $N^2$ , the data must be insufficient to fully populate the TM. The approach developed in this work is robust in cases where TMs are sparsely populated.

Figure 24. Training Algorithm Flowchart 62

Shows the training algorithm for HTMM. This machine learning algorithm preprocesses time-series data into a discrete number of system states, generates TMs from subsets of this training data, and parameterizes the TMs. The resulting distribution of Z-scores is used for anomaly detection in the testing algorithm.

Figure 25. Testing Algorithm Flowchart 63

Shows the testing algorithm for HTMM. Subsets of time-series data are used to create TMs, which are compared to the TM distributions from the training data. Depending on the rank relative to a user-specified confidence interval, these TMs are classified as either normal or anomalous.

Figure 26. Time-series data generated by ARMA, normal and anomalous models 65

Illustrates two example sets of time-series data generated by ARMA. The two data sets are visually indistinguishable by viewing in the time domain. The normal and anomalous ARMA models differ only in the sign of the first coefficient of both the AR and MA components (shown in Table 5).

Figure 27. Power spectral density of normal (a) vs. anomalous (b) data 65

Shows the frequency domain (power spectral density) of both normal and anomalous ARMA-generated data. This figure illustrates the difference between the two datasets in

the frequency domain, but also highlights need for sufficient data sampling when using frequency domain techniques.

Figure 28. Normal and anomalous transition matrices, sparse diagonal case. 66

Shows two transition matrices used for sparse probabilistic MCMC data generation. The many zero-valued elements in these TMs may invalidate other approaches that rely on multiplication of probabilities.

Figure 29. Normal and anomalous transition matrices, purely random case. 66

Shows two transition matrices used for random probabilistic MCMC data generation. The random nature of these TMs results in data which are indistinguishable in the frequency domain, and may invalidate other approaches that rely on frequency characteristics.

Figure 30. Bistochastic Matrix of Normal Data (a) and Anomalous Data (b) 68

Shows two transition matrices used for bistochastic MCMC data generation. The data generated by these bistochastic TMs are indistinguishable in the frequency domain and each state has an equal probability of occurrence, and may invalidate other approaches that rely on frequency domain and amplitude distribution characteristics.

Figure 31. Histogram (a) and Frequency Analysis (b) of Normal Bistochastic Data 69

Shows the frequency domain and histogram characteristics of the "normal" data generated using the TM from Figure 32 (a).

Figure 32. Histogram (a) and Frequency Analysis (b) of Anomalous Bistochastic Data 69

Shows the frequency domain and histogram characteristics of the "anomalous" data generated using the TM from Figure 32 (b). Comparing this figure to Figure 33, it is

clear that analysis methods relying on these metrics would fail to differentiate anomalous data in this case.

Figure 33. Cumulative Distribution Functions for TM Z-scores vs. Gaussian Model 75

Shows multiple CDFs of the Z-scores of training TMs at various sample sizes. This figure graphically shows the validity of the Gaussian distribution for modeling the Z-score distribution at the top level of the hierarchy shown in Figure 24. Further verification of this model is performed using the Kolmogorov-Smirnov and Cramér-von Mises criterion.

Figure 34. Histogram of Normal Data Transition Matrix Z-scores 78

Shows the distribution of only "normal" Z-scores from training data. This figure illustrates the range of normal Z-scores for TMs in this example data set, and further validates the Gaussian distribution approximation from Figure 35.

Figure 35. Example Z-Score distribution of example HTMM analysis 79

Shows a graphical example of how anomaly detection is performed on the Z-scores of TMs. By drawing a user-specified Confidence Interval (CI) around the histogram of normal Z-scores, this figure shows how anomalous Z-scores fall outside the expected range of this distribution.

Figure 36. ROC Curves for HTMM applied to ARMA-generated data 80

Shows ROC curves for ARMA-generated data at a variety of TM window lengths. The largest window size ( $2N/2$ ) is sufficient to populate a TM; the smallest window ( $N/6$ ) would not populate a TM, but still achieves positive detection rates.

Figure 37. ROC Curves tested against 500 anomalous spans for various anomaly lengths. 82

Shows ROC curves for bistochastic MCMC-generated data. At window lengths over  $2N/2$ , detection was very robust. At windows smaller than  $N/2$ , this method approached (but still outperformed) a random selector.

Figure 38. ROC curves for diagonal (a), random (b) TMs 83

Shows ROC curves for sparse diagonal MCMC-generated data. In this data, window lengths as small as  $N/2$  or  $N/4$  provided robust detection. These results show that HTMM's performance in a variety of data types relies partly on the structure of the transition matrix, but yields positive detection even when there is insufficient sampling to fully populate a TM.

Figure 39. Z-Scores for DB1 / SAX window 1 experimental normal (a) and anomalous (b) data 84

Shows the Z-scores for experimental rolling element data for normal (left, a) and anomalous (right, b) cases. In this case, SAX preprocessing was performed at a window of 1 (no aggregation). Separation between the two histograms is so large that the two cannot be shown clearly on the same plot.

Figure 40. Z-Scores for DB1 / SAX window 10 experimental normal (a) and anomalous (b) data 84

Shows the Z-scores for experimental rolling element data for normal (left, a) and anomalous (right, b) cases. In this case, SAX preprocessing was performed at a window of 10. Separation between the two histograms is less than in Figure 41, but still significant. This figure shows that HTMM performs robustly in a variety of time-series data resolutions.

Figure 41. Nearest Neighbor detection for experimental normal (a) and anomalous (b) data 86

Shows the distance to Nearest Neighbor for experimental rolling element data for normal (left, a) and anomalous (right, b) cases. Using the NN algorithm, there is some overlap between these histograms, which causes a higher rate of false-positive detection at certain user-specified CIs.

Figure 42. Nearest Neighbor detection for synthetic normal (a) and anomalous (b) data 87

Shows the distance to Nearest Neighbor for synthetic bistochastic data for normal (left, a) and anomalous (right, b) cases. Using the NN algorithm, there is almost complete overlap between these histograms, which invalidates the use of NN analysis on this type of data.

Figure 43. Example production of parts  $y_i$  with mean  $\mu$  and precision  $\phi$ . 92

Illustrates an example manufacturing process for parts with specified tolerance. The signal of interest is measured as a part dimension  $y$  with some expected value  $\mu$  and some precision  $\phi$ . This model shows how the precision parameter may be useful for determining changes in a manufacturing process.

Figure 44. Flowchart for obtaining mean normal posterior CDF 96

Shows the algorithm used in BPUSH for obtaining the mean normal posterior distribution. Subsets of training data are used to obtain posteriors based on a fixed prior (previously obtained through the sequential updating algorithm). These posteriors are averaged to find the mean posterior parameters.

Figure 45. (a) 100 distinct Gamma CDFs, (b) Mean CDF vs. CDF of means. 97

Shows a set of Gamma posterior CDFs generated from the same prior (left, a) and a comparison between the mean of the CDF posteriors (right, b, solid) and the CDF of the

mean of the posterior parameters (right, b, dashed). This further validates the approximation used in Equation (83).

Figure 46. Flowchart for parameterizing difference metric  $w$ . 97

Shows the algorithm used in BPUSH for obtaining and parameterizing the difference between posteriors and the mean posterior. The mean posterior from Figure 46 is compared to all other normal posteriors; these differences  $w_m$  are further parameterized using a Gaussian distribution with parameters  $\mu_w$  and  $\sigma_w$ .

Figure 47. Flowchart for Anomaly Detection Process 98

Shows the testing portion of the BPUSH algorithm. Subsets of test data  $Y$  are used to generate posteriors using the same fixed prior as in Figure 46, and compared to the same mean posterior to generate difference metrics as in Figure 48. The resulting Z-score of this difference metric is used to classify the subset of data as either normal or anomalous.

Figure 48. Simulated Precision Manufacturing Data with Anomalies,  $L=20$  100

Shows data generated from a normal precision manufacturing process (blue lines) and a process in which the precision was reduced by half (red dots). This figure illustrates the difficulty of using threshold-based techniques for anomaly detection. Because some anomalies do not exceed the "normal" range of the data, these techniques would fail to classify certain anomalies in parameter-driven data.

Figure 49. Comparison of Experimental Measurement Parameters 102

Shows data from various parameters of the experimental wafer etch manufacturing process. This figure illustrates the ergodic and stationary nature of the RF Tuner and RF Load parameters.

Figure 50. Precision Tracking in Simulated Precision Process Data,  $n=10$  103

Shows the results of a precision tracking algorithm on simulated precision manufacturing data with a sample size of  $n=10$ . This figure shows that precision monitoring yields positive detection in small sample sizes.

Figure 51. MOSUM Residuals in Simulated Precision Process Data,  $n=10$  104

Shows the results of the MOSUM algorithm on simulated precision manufacturing data with sample size of  $n=10$ . This figure shows that the MOSUM algorithm yields positive detection in small sample sizes, but may not consistently outperform other methods.

Figure 52. ROC Curves for Simulated Data, Various Anomaly Lengths 106

Shows the receiver operating characteristic curves for BPUSH applied to simulated data at various sample sizes and anomaly lengths. This figure shows that for sample sizes greater than  $n=20$ , detection is highly robust. For sample sizes as small as  $n=5$ , detection still outperforms a random selector.

Figure 53. Real World Sensor Environment (RWSE) Sensor Layout 109

Illustrates the geographic layout of sensors in the experimental sensor network. The roadway near sensor 5 is the primary ingress and egress point for vehicular traffic in this environment. Pedestrian traffic occurs between orange buildings, and is primarily responsible for activity at building doors and passive infrared sensor locations.

Figure 54. Simulated Sensor Environment Layout 110

Illustrates the geographic layout of sensors in the simulated sensor network. The roadway near sensor 8 is the only ingress and egress point for all traffic in this environment. Pedestrian traffic occurs between buildings and at door sensors 4 and 5; vehicular traffic is isolated to sensors 1, 2, and 8.

Figure 55. (a) Single Sensor and (b) Multi-Sensor Movement Increase Detected 115

Shows a detected change in inter-sensor movement using Poisson modeling on simulated sensor network data.

Figure 56. Numerical Score ROC Curves 115

Shows ROC curves for a Poisson model on simulated sensor network data anomaly types.

Figure 57. Sensor 5 vs. All Others: Correlation Result Matrix 118

Shows an example correlation result matrix for sensor 5 vs all others in the experimental sensor network. Sensor 5 typically correlates with sensor 16 at a delay of 12 to 18 seconds; the travel time between the two sensors is approximately 15 seconds.

Figure 58. "Arrival" Event, Nov. 8-28, Weekdays Only (Day 14 = Thanksgiving Day)

(a) Arrival Event Convolution Results (Sensor #5 leads #16 by 12 to 18 s)

(b) Total Distance from Nearest Neighbor in Training Set (Anomalousness Metric) 120

Shows the detection results using Nearest Neighbors for an "arrival" event (Sensor 5 leads sensor 16) for the weeks leading up to and including Thanksgiving Day (Thursday). The activity on this day is significantly different than the expected weekday activity in the network.

Figure 59. Double-Trigger Event, Nov 8-28, Weekdays Only

(a) Double-Trigger Event Convolution Results (Sensor #14 leads #14 by 6 to 12 s)

(b) Total Distance from Nearest Neighbor in Training Set (Anomalousness Metric) 121

Shows the detection results using Nearest Neighbors for a "door" event (Sensor 14 triggers twice) for the weeks leading up to and including Thanksgiving Day.

Figure 60. Convolution Analysis Results – Nearest Neighbor, Simulated Data 122

Shows the anomaly detection results using the Nearest Neighbors algorithm on the simulated data environment. Various anomaly types are detected using this approach, including changes in population as well as movement patterns.

Figure 61. Example distribution of expected “arrival event” activity 123

Shows the distribution of expected activity for an "arrival" event for weekdays in the experimental sensor network. At different times of day, the expected activity for this event varies. Events that fall outside the bounds of these expected distributions are automatically classified as anomalies.

Figure 62. Short-term Anomalousness Metric for Sensor 1 vs. 2 124

Shows the detection of a short-time anomaly in the simulated sensor network. The Z-score of this event’s frequency relative to the distributions from Figure 63 automatically classifies this as an anomaly.