2010-03

# Correlating temporal rules to time-series data with rule-based intuition

Kearton, Kristian

Monterey, California. Naval Postgraduate School


# NAVAL POSTGRADUATE SCHOOL

**MONTEREY, CALIFORNIA**

# THESIS

**CORRELATING TEMPORAL RULES TO TIME-SERIES DATA WITH RULE-BASED INTUITION**

by

Kristian Kearton

March 2010

| | |
|---|---|
| Thesis Advisor: | Simson L. Garfinkel |
| Second Reader: | Andrew I. Schein |

THIS PAGE INTENTIONALLY LEFT BLANK

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 0704–0188*



| 1. REPORT DATE *(DD–MM–YYYY)* | 2. REPORT TYPE | 3. DATES COVERED *(From — To)* |
|---|---|---|
| 10–3–2010 | Master's Thesis | 2008-01-03—2010-03-31 |

**4. TITLE AND SUBTITLE**

Correlating Temporal Rules to Time-Series Data With Rule-Based Intuition

**5a. CONTRACT NUMBER**

**5b. GRANT NUMBER**

**5c. PROGRAM ELEMENT NUMBER**

**6. AUTHOR(S)**

Kristian Kearton

**5d. PROJECT NUMBER**

**5e. TASK NUMBER**

**5f. WORK UNIT NUMBER**

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**

Naval Postgraduate School
Monterey, CA 93943

**8. PERFORMING ORGANIZATION REPORT NUMBER**

**9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)**

Department of the Navy

**10. SPONSOR/MONITOR'S ACRONYM(S)**

**11. SPONSOR/MONITOR'S REPORT NUMBER(S)**

**12. DISTRIBUTION / AVAILABILITY STATEMENT**

Approved for public release; distribution is unlimited

**13. SUPPLEMENTARY NOTES**

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. **IRB Protocol Number: n/a**

**14. ABSTRACT**

Analysts are frequently confronted with time-series data. A simple form is magnitude (or count) and time frame, whether the data is number of e-mails sent, number of cell phones called, purchases made by volume or cost, or a variety of other time-derived data. Studying the temporal dimension of data allows analysts more opportunities to find relational ties and trends in data, classify or group like activity, and even help narrow the search space of massively complex and large datasets. This thesis presents a new approach called the Rule Based Intuition (RBI) system that can evaluate time-series data by finding the best fitting rule, from a repository of known rules, to quickly infer information about the data. This approach is most applicable for analysts viewing large sets of data who wish to classify or correlate data from users' temporal activity.

**15. SUBJECT TERMS**

Temporal Analysis, Time-Series Data, Rule Based Evaluation, Supervised Learning

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT | b. ABSTRACT | c. THIS PAGE | | | 19b. TELEPHONE NUMBER *(include area code)* |
| Unclassified | Unclassified | Unclassified | UU | 83 | |

**Standard Form 298 (Rev. 8–98)**
**Prescribed by ANSI Std. Z39.18**

THIS PAGE INTENTIONALLY LEFT BLANK

**Approved for public release; distribution is unlimited**

**CORRELATING TEMPORAL RULES TO TIME-SERIES DATA WITH RULE-BASED INTUITION**

Kristian Kearton
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1997

Submitted in partial fulfillment of the
requirements for the degree of

**MASTER OF SCIENCE IN COMPUTER SCIENCE**

from the

**NAVAL POSTGRADUATE SCHOOL**
**March 2010**

Author: Kristian Kearton

Approved by: Simson L. Garfinkel
Thesis Advisor

Andrew I. Schein
Second Reader

Peter J. Denning
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

# ABSTRACT

Analysts are frequently confronted with time-series data. A simple form is magnitude (or count) and time frame, whether the data is number of e-mails sent, number of cell phones called, purchases made by volume or cost, or a variety of other time-derived data. Studying the temporal dimension of data allows analysts more opportunities to find relational ties and trends in data, classify or group like activity, and even help narrow the search space of massively complex and large datasets. This thesis presents a new approach called the Rule Based Intuition (RBI) system that can evaluate time-series data by finding the best fitting rule, from a repository of known rules, to quickly infer information about the data. This approach is most applicable for analysts viewing large sets of data who wish to classify or correlate data from users' temporal activity.

THIS PAGE INTENTIONALLY LEFT BLANK

# Table of Contents

# List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

# List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

# Acknowledgements

There are several people whom I wish to thank. First and foremost, I would like to thank Simson Garfinkel, whose encouragement, guidance, patience, and persistence ensured I completed this thesis. It would have been impossible to do this without you. Thank you, Dr. Garfinkel.

To Dr. Schein, your enthusiasm and support in using tools and your real-world experience in predictive forecasting has made this thesis better. Professor Koyak, from the Operations Research Department, for helping me select the best method to compare the different rules.

To Dr. Avi Mandelbaum, Professor of Operations-Research and Service-Engineering at TECHNION, Haifa, Israel for letting me use the Israeli bank data.

To the members of the National Consortium for the Study of Terrorism and Responses to Terrorism (START) for the use of their database.

Thanks Tuck, I most assuredly would not have done as well without your help.

To my loving and supportive wife who is tirelessly dedicated to our family and my career. Without your support, I would not be the person I am today. Thanks JED.

THIS PAGE INTENTIONALLY LEFT BLANK

# CHAPTER 1:
# Introduction

Analysts are frequently confronted with time-series data. A simple form often encountered is magnitude (or count) and time frame, whether the data is number of e-mails sent, number of cell phones called, purchases made by volume or cost, or a variety of other time derived data. Studying the temporal dimension of data allows analysts more opportunities to find relational ties and trends in data, classify or group like activity, and even help narrow the search space of massively complex and large datasets.

There are three basic methods of finding these patterns today. First, by hand—requiring heavy human interaction. It is slow, but can be aided by software visualization tools. Second, supervised learning—requiring both human and machine interaction. If done right, this approach can combine the strengths of both human and machine. Third, fully automated—requiring no human interaction. However, automatically generated rules can be nonsensical and of limited value. As computer speeds increase, there is hope that someday computers might be able to "think" like a human. While the idea of thinking machines is the goal of many good science fiction books, the state of the art in artificial intelligence is well below the sentient mark and looks to remain there for some time. That is why supervised learning methods are the most practical and relevant for today's data mining efforts.

This thesis presents a new supervised learning approach called the Rule Based Intuition (RBI) system. The RBI methodology can evaluate time-series data by finding the best fitting rule, from a repository of known rules, to quickly infer information about the data. Currently, the best scientists can do is to optimize and combine the strengths of both human and computer to help find the needed information. This concept is the idea behind RBI. The RBI method attempts to maximize the best capabilities of both humans and machines. By using known temporal patterns, analysts can combine the power and speed of computers with their own knowledge to reduce the necessary search space, find relevant information, and identify necessary causal relationships.

Finding temporal patterns is a very difficult problem, although it seems humans are good at this type of pattern recognition. For example, when a large company with several thousand personnel work overtime on a time critical project, the number of pizzas to be delivered to

the company building drastically increases. This pattern is repeated several times throughout the year. A competitor notices that immediately after pizza sales increase at the company, it announces a hostile takeover. Imagine instead the company is a news agency and pizza orders increase just before a major breaking news story or perhaps the company is the Pentagon and this trend is a precursor to military operations. In 1990, *Time* magazine published "And Bomb The Anchovies" [1], which correlates the purchases of pizza at the Pentagon to Iraq's invasion of Kuwait. If a local pizza delivery person can make these causal connections, what can trained analysts with better tools do?[1]

To the author's knowledge, the RBI methodology is a new application to data mining. In the extensive article reviews in Chapter 2, no one has tried this approach to data discovery. The RBI methodology is designed to be modular and extensible, as the rules can be developed and stored in a database for shared access. The modular design is well suited for remote analysis and allows knowledge experts to develop rules while lesser trained field collectors can automatically correlate data with reach-back to the experts. This approach can be developed for analysts viewing large sets of data or locally captured data providing data correlation of users temporal activity.

Using the RBI methodology, this thesis investigates the following questions: If we already know a temporal-spatial pattern, can we use what we know to help us find what we need? Is there a fast, proven method to take our temporal knowledge, evaluate it, and apply it to data we have not seen before to tell us something new? What DoD applications might this approach have?

To investigate these questions, this thesis presents the simple RBI framework written in the Python programming language for creating *temporal rules*—which we define as simple boolean functions which, when given an event located at a specific time, will return either `True` (meaning that the event is covered by the rule) or `False` (meaning that the event is not covered by the rule). Each rule is evaluated using a Poisson linear regression to determine which rule or rules best fit each dataset. This framework is used to create 137 rules. This set of rules is then used to process three data sets:

- The number of calls during the 1999 calendar year to the Automated Computer Time Service (ACTS) operated by the National Institute of Standards and Technology [3].

[1] For a more thorough review of the history of and the application to military/intelligence security, read "Introducing Traffic Analysis" [2].

- The number of calls each day to a bank call center in Israel during the 1999 calendar year [4].
- The number of terrorist attacks on each day of the 1999 calendar year, as tracked by the Global Terrorism Database.

RBI is a new temporal analysis approach and is applicable in several areas of research, intelligence collection, information operations and user classification. Its simple modular design and implementation lend itself as a new addition to the analysts tool set.

## 1.1 Motivation

Much of today's event-based research is geared toward finding temporal patterns, identifying change events, and discovering useful repeating patterns. Another facet of temporal data mining is data discovery, the ability to find relevant information by looking at how and when events occur and place them in the proper context.

The amount of digital data has increased exponentially in the last 20 years, causing an exploitation of ubiquitous and interconnected information. As the sea of information grows, agencies and businesses struggle to quickly find repeatable patterns to identify causal relations with significant events of interest. Many current methods are computationally expensive, can only be done in large database warehouses, or requires unique expertise to find the data and their connections. The increased demands placed on analysts to find useful, relevant information make automated information retrieval a requirement. Temporal analysis adds an additional capability not widely available to analysts.

Many intelligence analysts have calendars with anniversary dates and workflow wheels collected over years of dedicated observation. However, this is a very manually intensive process. The data has been collected but there is no method to automate this extensive temporal expertise—until now. The RBI methodology is the tool that can bridge this capability gap.

## 1.2 Temporal Analysis

In order to understand these concepts better, we should begin with a rudimentary understanding of the requirements of temporal analysis and different philosophical and computational approaches to the understanding of time. Any effective temporal systems should have the following criteria:

- Must allow for imprecise measurements of time (IMT). For example, computer generated logs are often incorrect by hours or sometimes days. By looking specifically at just dates and time, one might miss temporal relations.

- Must allow for imprecision in data (ID). One might not know the exact relationship between two events, but the system should be robust enough to understand or determine partial relations.

- Conform to the right degree of time (RDT). Some events happen in years and others happen in hours or even microseconds [5].

With these criteria in place, we will evaluate the different philosophical views of time and how they relate to computational implementation. Then grade each of the approaches to the criteria on a simple (+) or (-) system and provide capabilities and limitations of each approach. A (+) sign indicates that the criterion is easy to implement in that view of time while a (-) indicates not a failure of the view of time, but rather is difficult to implement in terms of complexity, cost of time, or cost of resources.

### 1.2.1 Different Views of Time

There have been centuries of research on the topic of time from philosophical to modern computational. Many brilliant minds have struggled with different aspects of time. Understanding the different views of time and their origins is important. There are three fundamental views of time: one is that time is moving or flowing with events in the past, present and future (*Date/-Time Line Systems*); another is that time is based on causality or observed events (*State Space and Formal Methods Modeling*); the third is that time is based on perceived instances defined by relative observation (*Relative Sequential Chaining*). Each of these philosophical views of time affect the method used in finding the data and are fundamental for analysts and computer scientists to understand the capabilities and limitations of the different implementations of temporal analysis.

**Aristotle and Newton (Date/Time Line Systems)** This view is often called the classical view of time. Aristotle viewed time as a magnitude of movement. Newton framed time in the physical world much in the same way as Aristotle. One of Newton's contributions to time is the idea that time is flowing. An example would be as a man walks across the room, time flows as he moves. In this view of time events are temporally anchored in the physical world. This view is

similar to looking at a calendar and events happen on Wednesday or the event happened after December 15, 1993. This is also called anchored time as it is set by a date or dates with the first instance being the anchor.

For example, computer system clocks use an anchored time to determine the "time." In this case, time is a set of counts in seconds where $t$ marks a count $t = (1, 2, 3, \ldots n)$ from an arbitrary date January 1, 1970, at 00:00 in UNIX systems. In this example January 8, 2010, at 08:06 is 1,262,937,960 seconds from January 1, 1970. The computer counts the seconds from the anchored date and then displays local time of January 8, 2019. This can be thought of outside of the observation or occurrence and is used in measure or relate events. This is helpful when dealing with multiple timelines because they can be compared together easily.

This temporal view focuses on building timelines from instances of specific dates/times. This approach is useful and easy for computers as the date/time becomes the reference. This is a good model for work flow analysis (the study of when and in what order people do work). Workflow is often connected to date/time like sunrise, sunset, and holidays. This date line approach is not flexible as events may not be able to be set to a precise date. In order to make it more flexible, these systems can define time in terms of a window, which adds complexity and ambiguity to the system. This model seems to do a poor job of capturing relative temporal information when window sizes overlap. This overlapping leads to greater complexity and less accuracy (Figure 1.1). The RBI system presented in this thesis uses this temporal view, but does not suffer from this form of complexity, because the uncertainty of time is dealt with in the Poisson Regression discussed in Chapter 3.

**Kant (State Space and Formal Methods Modeling)** Kant explains the "experience is possible only through the representation of a necessary connection of perceptions." [6] He summarized all perceptions are grounded in time. He goes on further to say "all changes take place according to the law of connection between cause and effect." [6]

This type of reasoning can be viewed as a state machine, with time being the connector between states. As a connector to causal events, each of the temporal ticks $t$ happens when there is a transition from state A (starting point of a man in a room) and state B (ending point across the room). Every discrete effect is modeled as a state and every transition is a unit of time.

This temporal view is useful for simple problem solving tasks and does not suffer from issues of complexity due to IMT or ID. However, this approach has limitations as it requires remem-

bering, storing, and searching all previous states. An important note is that each $t$ might be of different length, which can lead to difficulties if state transitions of two events are happening in parallel, as the temporal length of one state transition does not necessarily match the other transition (Figure 1.2). As technology improves, there is hope that some of these shortcomings can be surmounted.

**Einstein (Relative Sequential Chaining)** Einstein is famous for many ideas, but arguably, the most important to science are his thoughts on relativity. He describes time as:

> Every reference body... has its own particular time; unless we are told the reference body to which the statement of time refers, there is no meaning in a statement of the time of an event. [7]

Relative Sequential Chaining captures relative temporal information. However, as the amount of temporal information grows, the system suffers from search and memory issues (Figure1.3). Temporal logic helps elevate some of these challenges. Temporal logic is propositional logic with a temporal twist. An example is, if A happens before B and B happens before C, then A happens before C. James Allen defined thirteen basic possible temporal relationships and developed a transitive table, that is a fundamental cornerstone in relative temporal logic [5]. This has been a growing field of interest especially in the business community as people and organizations attempt to make personal interactions and market predictions more effective. Much of this area of study focuses on individuals and their work and consumption activities.

### 1.2.2 A Historical Example

Perhaps one of the most famous uses of data line temporal analysis is that of John Snow, a doctor in London in 1855. His work is unique in that it combined not only date line temporal analysis but also spatial analysis with incredible effect. He describes the event as:

> The most terrible outbreak of cholera which ever occurred in this kingdom, is probably that which took place in Broad Street, Golden Square, and the adjoining streets, a few weeks ago. Within two hundred and fifty yards of the spot where Cambridge Street joins Broad Street, there were upwards of five hundred fatal attacks of cholera in ten days. The mortality in this limited area probably equals any that was ever caused in this country, even by the plague; and it was much more sudden, as the greater number of cases terminated in a few hours. The mortality would

undoubtedly have been much greater had it not been for the flight of the population [8].

The method in which cholera was spread was not well understood, which is why John Snow's use of temporal and spatial mapping was so revolutionary. He correlated the possible water contamination to a rain storm, which burst sewer piping and overflowed into kitchen drinking water, which eventually contaminated a local water source. He organized the event by numbers of dead and sick per day by location and combined them with drawings of public works piping in the city. With this, he was quickly able to deduce the water source as the only possible source of contamination and that is was confined to a single water pump. Upon physical investigation of the water, he confirmed the contaminated source and had the handle removed from the pump. His quick deduction of the outbreak to a single hand pump water supply, helped by temporal and spatial analysis, ensured the removal of the hand pump handle and eliminated risk to others around the pump. A few weeks after the handle was removed, he was able to return to the area for further study.

Snow's book uses a map (Figure 1.4) to mark the deaths of the people around the pump. Each death is shown as a small black rectangle. He noted two areas that had fewer than expected deaths. The workhouse had 535 people living in it at the time of the outbreak. Given the number of death surrounding the work house, there should have been approximately 100 deaths; there were only five. The brewery employed 70 people had no deaths. As it turned out, both areas had other sources of water on their property. This is clear case where temporal and spatial pattern recognition helped end a devastating epidemic.

### 1.2.3 Local Time

What time is it? Asked this question, most people would look at their watch, a cell phone, or a nearby clock. Asked this question before the industrial age, people would answer by looking at a water clock, a sundial, or the sun. All of these different techniques for learning the time report *local time*—the time that people experience.

For thousands of years, local-observed time was the primary time reference. People rarely had the need to synchronize time accurately. When they did, they were able to use bells, drums, or later, clock towers.

**Time Zones**

Local time is entirely dependent upon Latitude; if one city is 3 degrees to the East of a second, then it will take 12 minutes between the instant that the Sun passes through the zenith of the first city and the time that the Sun passes through the zenith of the second. Still, this didn't present much of a problem to humanity until the development of bidirectional instantaneous long-distance communications (necessitating two parties to synchronize their actions), and congested single-track long-distance trains (necessitating that trains time their usage of the single track resource).

Scottish-born Canadian inventor Sir Sandford Flemming suggested a worldwide system for timezone in 1878. He proposed 24 meridians, each 15 degrees or one hour apart in longitude, starting from Greenwich. The local time for each zone would be the time of the meridian that bisected it. On November 18, 1883, most of the United States and Canadian railroads began to use this system, which reduced the number of time zones from 56 to four we use today [9]. Despite being adopted by the railroads in 1883, the United States did not legally adopt Standard Time until the passage of the Standard Time Act on March 19, 1918.

**Daylight-Saving Time (DST)**

Benjamin Franklin is credited with the invention of Daylight-saving time. He discussed his observations and ideas in an essay titled, "An Economical Project." He wrote this essay in 1784 while in Paris as an American delegate. The original purpose of the idea was to save on the cost of lamp oil and candles in Paris [10]. Given that people's day-to-day activities were pegged to the clock even in the late Eighteenth Century, Franklin's idea was to shift clocks back an hour in the fall so that people would experience an additional hour of daylight during the afternoon working hours (and have an hour of daylight less in the morning, when most people were asleep). He estimated that in a single year, French shopkeepers could save one million frances on candles alone. The United States adopted DST in 1918 then repealed it after the end of World War I, because it was unpopular. President Johnson signed the Uniform Time Act of 1966 making DST law. The Energy Policy Act of 2005 amended the 1966 act and started DST on the second Sunday in March and ends the first Sunday in November [11].

**Coordinated Universal Time (UTC)**

UTC is the worldwide system for civil time. Atomic clocks are kept in labs around the world. The International Bureau of Weights and Measures uses this timing clocks and to determine the international standard UTC, which is accurate to almost a nanosecond or one billionth of a

second per day. UTC is distributed from various radio stations and from the Global Positioning System or GPS. U.S. and its territories timezones are set to hours from UTC, though not all countries follow UTC year round. The United Kingdom is one exception as UTC is the local time because Greenwich is located there [11].

With an understanding of temporal physiological and computational restrictions and different ways time is calculated and observed, it is obvious as to why time and understanding temporal events can be challenging. That is why a simple system like RBI could be so important to analysts in the field today. The RBI methodology is simple, effective, and has numerous DoD applications.

## 1.3 Application to DoD

Intelligence agencies and military command staffs must view massive amounts of data in order to categorize and place the data in context. Placing data in context transforms it in into information. This transformation is a critical step to making informed decisions. RBI can help transform data into information.

There is no panacea for intelligence. The RBI methodology does not replace human interaction, rather, it is required. It does not solve all of the collection or analysis requirements; it is not intended to. However, this methodology shows significant promise as a useful and effective tool for analysts, intelligence agencies, and law enforcement.

Below are possible applications that directly support current DoD intelligence and analyst's needs.

Some include:

- **Terrorist Activity**: look for failed terrorist attempts, identify probable locations and times of Improvised Explosive Device (IED) placements, attribute activity to certain organizations, classify different social network activity, identify the planning phase of an on going terrorist operation, sort large data sets quickly for relevant data, identify changes in operational tempo.

- **Criminal Activity**: find financial activity, classify behavior of personnel in an organi-

zation or the organization itself, identify non standard activity, determine social network activity and classify that activity.

- **Nation State Activity**: classify specific organizational activity, determine irregular activity, alert analysts to indications and warnings, identify relevant data in large data sets, and predict military movement.

## 1.4 Outline of this Thesis

Chapter 2 gives an overview of supporting and related work. Chapter 3 goes into the mathematics and theory behind the techniques and concepts used in the experiments for this thesis. Chapter 4 describes the experiments conducted, the data sets, and any pre-processing done. Chapter 5 discusses these results and lists ideas for future work. Chapter 6 is closing thoughts and conclusions.

| IMT | + | ID | - | RDT | + |
|---|---|---|---|---|---|

**POSITIVES:**

1. Systems are easily parallelizable with multiple timelines because timelines can be reduced to smallest common denominator of time
2. Easy to comprehend for humans, think of a calendar
3. Easy to model for computers
4. Good for workflow analysis

**NEGATIVES:**

1. Anchored time is difficult to implement if the time given is not correct. Because of this it is easy to miss temporal correlations. One method to overcome this issue is a time window. Another method, the one this paper explores is the use of regression analysis to solve this sliding window.
2. Certain implementations can be difficult to model and become more complex when time windows overlap.

Figure 1.1: Summary of Date/Time Line Systems

| IMT | + | ID | + | RDT | - |
|---|---|---|---|---|---|

**POSITIVES:**

1. Easy for computers and humans to understand
2. Concepts and models are well understood. A Turning machine is an example of a state space machine.
3. Formality can ensure both completeness and correctness

**NEGATIVES:**

1. This approach it not easy to implement when evaluating multiple event state machines as the transition for states are not guaranteed to be the same length.
2. Longer temporal patterns can be more time consuming and results in heavy resource or time penalties.

Figure 1.2: Summary of State Space and Formal Methods Modeling

| IMT | + | ID | + | RDT | - |
|---|---|---|---|---|---|

**POSITIVES:**

1. These approached are not time dependent making implementation easier and faster.
2. The temporal logic system implemented is well understood and easy to model particular trends.

**NEGATIVES:**

1. Implementers need to understand propositional logic. This takes formal training.
2. Defining sub-events within larger events becomes more complicated.

Figure 1.3: Summary of Relative Sequential Chaining

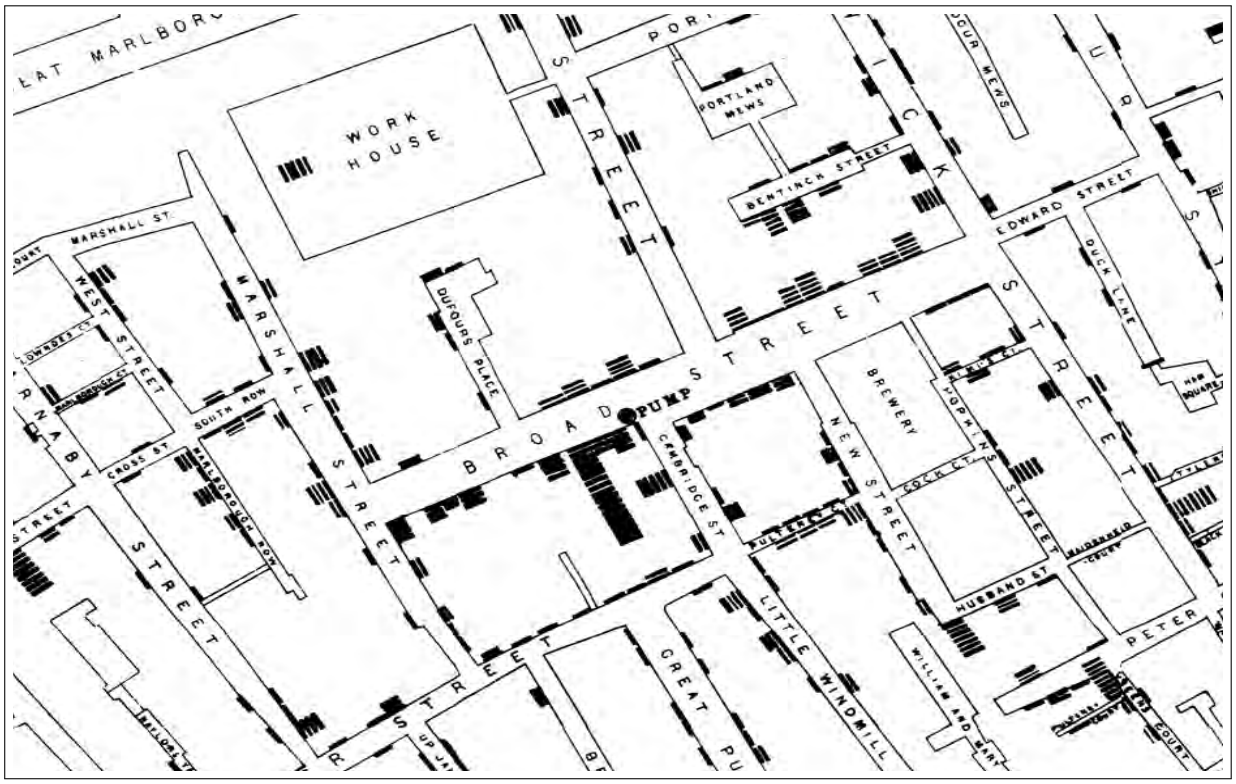


Figure 1.4: Section of John Snow's Map Showing Location of the Water Pump Infected with Cholera and the Resulting Deaths from the epidemic. From [8].

# CHAPTER 2:
# Prior and Related Work

Understanding events and/or activities in a temporal context is important to many fields in business, science, mathematics, and philosophy. Through the years, several different methods and techniques have tried to capture a sense of activity or detect significant, relevant events with temporal data. Understanding some of these approaches is important to see how current research is conducted.

This chapter covers prior work in these areas.

## 2.1 Survey of Temporal Analysis Research

There have been decades of research on the topic of time. Below is a comprehensive but not inclusive review of articles and applications of temporal research. These papers are grouped into the three views of time as discussed in Chapter 1 (Date/Time Line Systems, State Space and Formal Methods Modeling, and Relative Sequential Chaining). Many different techniques have been used to research temporal data, but arguably all research knowingly or unknowingly use one of these views of time.

Table 2.1: Temporal Analysis Papers—Date/Time Line Systems

| Title | Year | Short Description |
|---|---|---|
| Logical Modeling Of Temporal Data [12] | 1987 | Discusses fundamentals of temporal data issues and defines a new type of temporal model. |
| Automated Temporal Reasoning About Reactive Systems [13] | 1996 | Helps define formal syntax and semantics for propositional temporal logic. |
| Visualization Of Spatio-Temporal Information In The Internet [14] | 2000 | Uses a dynamic temporal visualization framework for placing objects in time and space. |
| Discovering Calendar-Based Temporal Association Rules [15] | 2001 | Attempts to discover temporal association rules derived from calendar dates. |
| *Date/Time Line Systems—Continued on next page* | | |

| Title | Year | Short Description |
|---|---|---|
| Work Rhythms: Analyzing Visualizations Of Awareness Histories Of Distributed Groups [16] | 2002 | Develops visualizations of business work flow and does histogram analysis of daily activity over time. |
| Rhythm Modeling, Visualizations And Applications [17] | 2003 | Is a refinement of their work done in 2002 using clustering techniques and different visualizations. |
| Visually Mining And Monitoring Massive Time Series [18] | 2004 | Product description of a developmental visualization and time series tool. |
| Mining And Visualizing The Evolution Of Subgroups In Social Networks [19] | 2006 | Recognizes the importance of temporal changes of online communities and discusses ways to model them. |
| Learning recurrent behaviors from heterogeneous multivariate time-series [20] | 2007 | Demonstrates the utility of learning meaningful patterns in multidimensional and heterogeneous data from information automatically collected from sensors worn by people. |
| Exploring Global Terrorism Data: A Web-Based Visualization Of Temporal Data [21] | 2008 | Develops visualization techniques to help analysts find interesting patterns in a Global Terrorism Database. |
| Google News Timeline [22] | 2009 | Innovative way to display news from different venues organized in a customizable temporal view. |

Table 2.2: Temporal Analysis Papers—State Space and Formal Methods Modeling

| Title | Year | Short Description |
|---|---|---|
| Mining Sequential Patterns: Generalizations And Performance Improvements [23] | 1996 | Gives an organization algorithm for itemsets. |
| Discovery of Frequent Episodes in Event Sequences [24] | 1997 | Develops a framework for discovering frequent episodic data. |
| Discovering Frequent Event Patterns With Multiple Granularities In Time Sequences [25] | 1998 | Discusses the ideas of an event structure and temporal granularity. |
| Knowledge-Based Event Detection In Complex Time Series Data [26] | 1999 | Uses medical sensor data to find and detect events in temporal data. |
| Correlation Mining Between Time Series Stream And Event Stream [27] | 2008 | Presents a new algorithm to correlate temporal data and events. |
| *State Space and Formal Methods Modeling—Continued on next page* | | |

| Title | Year | Short Description |
|---|---|---|
| Temporal Mining For Interactive Workflow Data Analysis [28] | 2009 | Develops a state space approach for evaluating process control logs with workflow graphs. |

Table 2.3: Temporal Analysis Papers—Relative Sequential Chaining

| Title | Year | Short Description |
|---|---|---|
| Mining Association Rules Between Sets Of Items In Large Databases [29] | 1993 | Introduces the notion of itemsets and how they can be applied to determining buying behavior. |
| Segmenting Time Series: A Survey And Novel Approach [30] | 1993 | Completes a survey of three time series segmentation algorithms, sliding window, top-down and bottom-up. The author states that a combination of sliding window and bottom-up yield drastically better results than any other combination. |
| Wide Area Traffic: The Failure Of Poisson Modeling [31] | 1995 | Discusses assumptions of Poisson regression and exceptions to those assumptions for network traffic. |
| Discovery Of Frequent Episodes In Event Sequences [24] | 1997 | Presents a framework for discovering frequent episodes in sequential data. |
| A Framework For Knowledge-Based Temporal Abstraction [32] | 1997 | Describes a domain-independent knowledge-based inference structure. |
| Rule Discovery From Time Series [33] | 1998 | Introduces two different problems; one, data clustering and two, development of rule induction using these clusters. |
| Efficient Time Series Matching By Wavelets [34] | 1999 | Uses Discrete Wavelet Transform (DWT) to analyze and match time series data. |
| Event Detection From Time Series Data [35] | 1999 | Discusses time series data and defines a method to determine change point or event detection. |
| *Relative Sequential Chaining—Continued on next page* | | |

| Title | Year | Short Description |
|---|---|---|
| Learning Recurrent Behaviors From Heterogeneous Multivariate Time-Series [20] | 2007 | Develops a supervised model that creates an unsupervised learning algorithm of temporal activity for people in their homes. Tuning the unsupervised learning portion turned out to be difficult and severally effected system performed. |
| Data mining with Temporal Abstractions: learning rules from time series [36] | 2007 | Users develop formal temporal patterns using Allen's temporal operators. Then their algorithm identifies events based on these formal patterns. |
| Unsupervised Pattern Mining From Symbolic Temporal Data [37] | 2007 | Builds a framework to view temporal concepts and differing data models for data mining using unsupervised learning methods. |
| Discovery Of Activity Patterns Using Topic Models [38] | 2008 | Uses modern Natural Language Processing (NLP) techniques to determine activity patterns. |
| Spatial-Temporal Causal Modeling For Climate Change Attribution [39] | 2009 | Develops a spatial-temporal regression model based on a Graphical Granger Model. |
| Spatial-Temporal Association Between Fine Particulate Matter and Daily Mortality [40] | 2009 | The authors investigates the spatial-temporal nature of pollution and mortality using a Bayesian framework. |

### 2.1.1 Date/Time Line Systems Examples

Combining the concepts of space and time is another way to look at data and discover relations. This method is important in information discovery of objects or events that have temporal and spatial relations. (Figure 2.1) shows different ways to look at the time: as single event, two moments in time, interval (passed or current), and how they apply to a space or location.

"Rhythm modeling, visualizations and applications" builds on previous work in rhythm detection and describes algorithms to detect and model temporal patterns from online geolocation data. The tools the authors built generate visualizations for users to see their workflow processes. The tools use heuristics to determine the threshold values, then cluster work events by minimizing Euclidean distance. Probability distributions are recalculated and the process is repeated until the initial and refined estimates converge. The paper then discusses several visualizations created by the program and evaluates them and proposes different possible applications

| | Existence | | Spatial location | Shape and size | Thematic data |
|---|---|---|---|---|---|
| | Instant events | Durable objects | | | |
| Single moment $t$ | What events occurred and where? | What objects existed and where? | Where was each object at $t$? | What shapes/sizes had the objects at $t$? | What were values of an attribute at $t$? How were they distributed? |
| Two moments $t1$ and $t2$ | What is the difference in number, kind, or spatial distribution of events between $t1$ and $t2$? | What objects remained, appeared, died? How did the spatial distribution change? | Where/how far did each object move? | What is the difference between shapes/sizes at $t1$ and $t2$? | What is the difference between values/spatial variations of the attribute at $t1$ and $t2$? |
| Interval $[t1, t2]$ (summary) | What events occurred during $[t1, t2]$? | What objects existed, appeared, died during $[t1, t2]$? | How did the objects move? (trajectory) | How often did the objects change? How much? | What are average (minimum, maximum, dominant) values on $[t1, t2]$? |
| Interval $[t1, t2]$ (progress) | How did the number, kind, spatial distribution pattern of events/objects change in time? | | How fast did the objects move? Did they meet? How did the speed change? | How did the shapes/sizes develop with the time? | How did the values and their spatial distribution develop in time? |
| | When did maximum changes occur? Were there still periods? Is there any temporal trend? Was (where was) the development monotonous/periodic? | | | | |

Figure 2.1: Framework for Dealing with Events that have Both a Temporal and Spatial Relation. From [14].

for these tools [17].

On April 20, 2009, Google announced *Google News Timeline*. It organizes news search results in a zoomable, graphical timeline Figure 2.2. This webtool is another example of Date/Time Line Systems. The view is an anchored scaleable calendar view with a selectable temporal granularity of day, week, month, year, and decade [22].

### 2.1.2 State Space and Formal Methods Modeling Example

"In Temporal Mining for Interactive Workflow," Berlingerio, Pinelli, Nanni, and Giannotti build upon work done in workflow mining. Their approach reads computer log data and builds temporal process models by grouping sets of execution statements with similar execution times, grouping semantics of executions, and interfacing with domain experts to select the appropriate models. This method can help identify abnormalities in the logs and can in some cases, generate new process models [28].

### 2.1.3 Relative Sequential Chaining Examples

In the early 1990s, regression was used to model network traffic analysis. It was accepted that all network traffic arrival rates could be modeled using regression analysis. In their paper "Wide

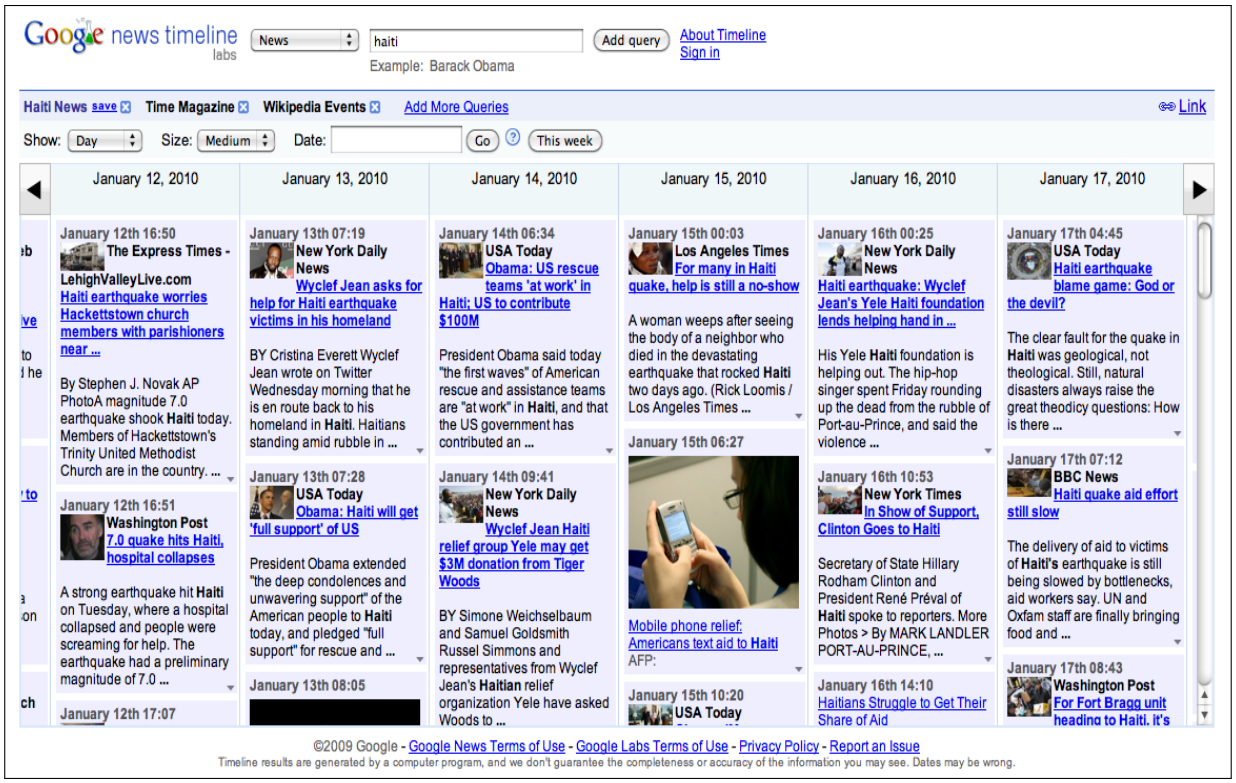


Figure 2.2: Screen Capture from Google's News Timeline Tool

Area Traffic: The Failure of Poisson Modeling", Paxson and Floyd discuss several statistical methods used to model computer network traffic. They found that Poisson Linear Regression does not adequately model all forms of network traffic. They state Poisson distributions are only valid for modeling the arrival of user sessions and that the protocols are too "bursty" and therefore have different time scales which prevent the model from performing well. Again, temporal granularity is an issue and must be understood for the Poisson distribution to work well.

Das, Lin and Mannila developed a method to create and evaluate rules for stock market prediction. They were able to generate rules based on exploratory induction of discrete time series data. The first step in their process was to use K-means clustering to classify stock data. They then developed an algorithm to discover simple rules from these different sequences. Their method created a large range of rules, some with limited value. To compensate for the large number of rules, they used the J-measure for rule-ranking developed by Smyth and Goodman in 1991. They found their technique needed the help of human interpretation to find the most useful rules for the particular dataset [33].

Guralnik and Srivastava developed a data mining method to separate temporal data into events when the model changes overtime. This technique is often called *change-point detection*. Their method requires the desired number of change points to be given, which can be a drawback. The take-away of this paper is that incremental optimization is not nearly as effective as global optimization over the whole set of data [35].

Another paper dealing with change point detection combines two standard approaches, mainly, finding the change points given a desired number of change points and uses a best fit curve to determine the interval between successive change points. The authors studied detecting change points by using Maximum Likelihood Estimation (MLE). If the number of change points are known before hand, then the statistical likelihood, $L$, of the change point is equal to

$$L = \begin{cases} \prod_{i=1}^{k} \sigma_i^{-m_i} \\ \left[\sum_{i=1}^{k} m_i \sigma_i^2\right]^{-n/2} \end{cases} \tag{2.1}$$

where $k$ is the number of change points, $m_i$ is the number of time points in segment $i$, and $n$ is the total number of points.

If the change points are not known, the maximum likelihood estimate of the $\theta_i$'s can be found by maximizing the likelihood $l$ over all possible sets of $\theta_i$'s, or equivalently, by minimizing $-2 \log L$ the function is equivalent to,

$$-2 \log L = \begin{cases} \sum_{i=1}^{k} m_i \log \sigma_i^2 \\ n \log(\sum_{i=1}^{k} m_i \sigma_i^2) \end{cases} \tag{2.2}$$

Shahar developed a general framework for reusing domain-independent knowledge for solving temporal abstraction and enabled sharing of domain-specific knowledge with other tasks in the same domain. This framework has been used in several different areas of medical research and has proven useful in the organization of his temporal work. Specifically, he defines five knowledge-based temporal-abstraction methods: temporal-context restriction, vertical temporal inference, horizontal inference, temporal interpolation, and temporal pattern matching [32].

Huynh, Fritz, and Schiele use Natural Language Processing (NLP) machine learning methods to automatically annotate users' daily activity. Subjects wore two tracking devices for several days. The output from the device was converted into documents of discrete activity labels.

Using Latent Dirichlet Allocation, the documents were associated with these activity labels. The authors then showed how labeling of events could be done with unsupervised learning, though supervised learning yielded the best results. This technique has potential to prove more useful and robust then other unsupervised learning algorithms because many of the techniques used in NLP are understood [38].

# CHAPTER 3:
# Techniques

This chapter documents the techniques, concepts, and technical approaches used in the experiments for this thesis. This chapter will cover certain fundamental concepts and terms necessary for basic understanding of this research.

## 3.1 Generalized Linear Models and Logistic Regression

Generalized linear models (GLM) are a set of models that approximate more complex phenomenon. Linear models are an important class of probabilistic model. In the 1950's, logistic regression became an important tool in biostatistics and, today, is used in many areas of science, engineering, business, and economics [41].

### 3.1.1 Poisson Linear Models

Within the GLM there are special sets of logistic regression models for univariate response data. A Poisson linear regression model works best with independent count data such as the number of calls to a call center [41]. In a Poisson Linear Model, the variance is a function of the mean.

**Terms and Assumption**

Terms and assumption in Poisson Linear Regression:

**Covariates or Regressor Variables** ($x_1, x_2, \cdots, x_k$) are the items one wishes to test against. For example, if one wanted to know the effect of certain drugs based on age, sex, dose, the covariates would be age, sex, dose.

**Regression Coefficients** ($\beta$) are the unknown model parameters that are calculated using the Poisson LM.

**Response Variable** ($y$) item of interest or collected data. This count could be the number of calls per day to a call center or the number of IED attacks in a given area per week.

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} \tag{3.1}$$

**Z Value** - In the case of this test data the higher $Z$ value the better. As shown in Chapter 4 the $Z$ values are used to rank the individual rules for the reasons mentioned above. Additionally, the $Z$ values are in absolute terms because it is a logistic regression model.

**P-value Pr($>|$Z$|$)** - The probability that the rule added appears useful, in the case where it is not. Therefore a $P$ value close to zero indicates a good predictor. Data derived from Chapter 4 show extremely small $P$ values. These values quickly rounded to zero as the $Z$ value grows as shown in table. This means $P$ values cannot be used to prioritize the rules.

| $Z$ value | $P$ value |
|---|---|
| 0.5 | 0.6170 |
| 1 | 0.3173 |
| 2 | 0.0455 |
| 3 | 0.0027 |
| 4 | 6.3e-05 |
| 5 | 5.7e-07 |
| 6 | 1.9e-09 |
| 7 | 2.5e-12 |
| 8 | 1.2e-15 |

**Mean ($\mu$)** The mean number of occurrences or arithmetic mean.

$$\mu(\mathbf{x}) = \frac{1}{n} \cdot \sum_{i=1}^{n} x_i, i = 1, 2, \cdots, n \tag{3.2}$$

**Maximum Likelihood Estimator (MLE)** provides a estimate for how well a model fits the data.

**Independence** - We say that two random variables are independent when of two events $A$ and $B$ such that P($A \cap B$) = P($A$)P($B$).

The assumptions for Poisson Regression are:

1. Observations are independent.

2. Variance and mean are equal:

$$E(x_i) = \mu(\mathbf{x}_i), i = 1, 2, \cdots, n \tag{3.3}$$

3. A set of regressors $x_1, x_2, \cdots, x_k$ influence $\mu$ via the model.

$$\mu_i = e^{\mathbf{x}_i'\boldsymbol{\beta}}, i = 1, 2, \cdots, n \tag{3.4}$$

With these assumptions in place we can set:

where $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$,

$\mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1k} \\ 1 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix}$,

$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}$,

and E($\boldsymbol{\varepsilon}$) = 0

In order to find the Maximum Likelihood Estimator (MLE), we start with

$$L = \ln \mathfrak{L}(\mathbf{y},\boldsymbol{\beta}) = \sum_{i=1}^{n} [-e^{\mathbf{x}_i'\boldsymbol{\beta}} + y_i \mathbf{x}_i'\boldsymbol{\beta} - \ln y_i!] \tag{3.5}$$

Using this equation, the MLE is derived using an unsigned numerical search procedure like iteratively reweighed least squares [42].

## 3.2 Program Design

Temporal rules and data manipulation were programmed in Python version 2.6, while statistical analysis was done in R version 2.10.0 using the Rpy version 2.0.8 interface to send commands to and retrieve data from R.

The Python temporal rules were created as class objects to allow for logical design of rules and polymorphic rule creation. In many cases, rules were combined to make other rules. For example, *US Workweek* is a combination of the common work days Monday thru Friday and the removal of national holidays.

The temporal assumptions in this thesis are as follows. One, all temporal rules relate to daily activity. Two, both the scope of the rules and the data have the same temporal granularity, meaning if the data are counts in days, then the rules are applicable to days.

### 3.2.1 Formatting Data

In some cases, the data had a finer temporal granularity than days so the information was transformed to meet a daily format. For example, one dataset had a temporal granularity of seconds and was therefore transformed into a daily number of calls. This transformation ensured the rules and dataset had the same temporal measure (i.e., days). If the dataset where not set to a daily count, different temporal rules would have had to have been created so that these rules matched the granularity of the data. Since doing this would not have added to the experimental value of the thesis, the Israeli data was made to have the same temporal granularity as the ACTS data (i.e., day). That said, the rules could have been created to match any temporal granularity desired.

Data formatting was done using simple Python expressions. Modules were created for each dataset to make sure it configured to the correct temporal dimension (i.e., day). Additionally, only the first instance of a unique rule was used. In the example below Muslim and Jewish Workweeks are the same. Because *MuslimWorkweek* rule was analyzed first the *JewishWorkweek* was dropped from consideration. This will be discussed in Chapter 4 in greater detail.

### 3.2.2 Developing and Verifying Rules

Rules were developed using a simple hypothesis, cultural norms and work patterns are predictive of human behavior. Therefore, rules can be developed and used to correlate their behavior such as daylight saving activity, or calls to a call center.

DST change dates where generated from the computer's own time zone algorithm. The computer actually calculates over 500 different zones around the world. Many of these zones have the same DST transitions. For example, *Atlantic.Bermuda* and *America.Denver* and *America.New_York* all have the exact same DST transitions. Because of this duplication only the first unique rule was tested while all other similar rules were dropped from the rule list.[2]

Some of the rules used in these experiments were developed from workday patterns (standard workweek minus holidays), religious regional holidays from the Jewish or Muslim. The Western workweek starts on Monday and ends on Friday, while the Israeli and Muslim workweeks start on Sunday and end on Thursday. Holidays are unique to a country and/or a geographical region. An example of a unique holiday would be Independence Day (July 4) in the United States. Of course, there are other holidays that multipule cultures such as New Years. However, taken as a whole, the cultural work year is unique to every country and therefore can be used to correlate data to location. Even within predominately Muslim countries, where holidays are very closely tied to the religious holiday, there are differences which allow for differentiation.

Again rules chosen were based on their tight binding to a specific cultural behavior like observed holidays and DST. However, these rules could have as easily have been known patterns of a military unit's operational tempo, spending patterns of certain demographics, or timing of terrorist attacks.

## 3.3 Implementation

The actual rules are implemented as instances of classes that subclass the `TimeRule` abstract super class in the Python file `time_rules.py`.

The abstract superclass defines a simple abstract rule that matches no time events:

```
class TimeRule:
    DESCRIPTION = "Abstract rule Class."
    def inRule(self,tval):
        """The default inRule is that it is never in the rule.
        tval is in local time."""
        return False
    def __str__(self):
        return "%s" % (self.__class__.__name__)
```

[2] As it turned out, this pre-filtering step was not necessary because R could do stepwise regression which would have had the same effect, by only adding rules that change the models accuracy. Because adding the same rule would not change the models accuracy it would not have been added to the list.

Subclasses of this abstract class implement specific rules. For example, this rule implements the Israeli work days:

```
class Israeli_WorkDays(TimeRule):
    """Return "True" if it is a day on which somebody works in Israel."""
    def __init__(self):
        self.Israel_Holiday_Rule = Jewish_Holidays_1999
        self.WEEKDAY_RULE = IsraeliWeekday()
    def inRule(self,tval):
        # A day is an israel workday if it is workday and not a holiday
        return self.WEEKDAY_RULE.inRule(tval) and self.Israel_Holiday_Rule.inRule
```

The Python list `rule_list` holds every rule that is to be tested. Rules are added as objects, which are typically instances of a rule class. This allows a single generic class to generate many specialized rules, although some rule classesa re used to generate just a single instance. Adding instances to this list is straightforward:

```
rule_list.append(Israeli_WorkDays())
```

A special `Almanac` class allows the creation of rules (instances) that match a specific day of the year. For example, this code creates a rule that matches the actual US Independence day in 1999 and adds it to the `rule_list`:

```
rule_list.append(Almanac("US Independence Day 1999",[date(1999,7,4)],True))
```

With this in place, multiple rules can be combined to make a larger rule. For example, *US_WorkDays* is a combination of the *WesternWeekday* rule and the days that are not *US_Holidays*.

```
class US_WorkDays(TimeRule):
    def __init__(self):
        self.US_HOLIDAY_LIST = [US_NewYears(),US_MartinLuther(),
        US_WashingtonBDay(),US_MemorialDay(),US_IndependanceDayObserved(),
        US_LaborDay(),US_ColumbusDay(),US_VeteriansDayObserved(),
        US_Thanksgiving(),US_ChristmasDayObserved()]
        self.WEEKDAY_RULE = WesternWeekday()
    def inRule(self,tval):
        """ It is not a workday if it is a US holiday
```

```
        Go through the list of US holidays and see if any of the rules match.
        If so, return false"""
        for r in self.US_HOLIDAY_LIST:
            if r.inRule(tval)==True: return False
        # It is a Workday if the day is a weekday
        return self.WEEKDAY_RULE.inRule(tval)
rule_list.append(US_WorkDays())
```

The following rules were used:

Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
January
February
March
April
May
June
July
August
September
October
November
December
WesternWeekday
MuslimWeekday
IsraeliWeekday
US_NewYears
US_MartinLuther
US_WashingtonBDay
US_MemorialDay
US_IndependanceDayObserved
US_LaborDay
US_ColumbusDay
US_VeteriansDayObserved
US_Thanksgiving
US_ChristmasDayObserved
US_WorkDays
US_BimonthlyPay
Israeli_WorkDays
Muslim_WorkDays
DSTRule(Africa/Abidjan)
DSTRule(Africa/Addis_Ababa)
DSTRule(Africa/Algiers)
DSTRule(Africa/Blantyre)
DSTRule(Africa/Cairo)
DSTRule(Africa/Ceuta)
DSTRule(Africa/Tunis)
DSTRule(Africa/Windhoek)
DSTRule(America/Adak)
DSTRule(America/Anchorage)
DSTRule(America/Anguilla)
DSTRule(America/Araguaina)
DSTRule(America/Argentina/Buenos_Aires)
DSTRule(America/Argentina/Catamarca)
DSTRule(America/Argentina/San_Luis)
DSTRule(America/Asuncion)
DSTRule(America/Atikokan)
DSTRule(America/Belem)
DSTRule(America/Belize)
DSTRule(America/Boa_Vista)
DSTRule(America/Boise)
DSTRule(America/Campo_Grande)
DSTRule(America/Cancun)
DSTRule(America/Caracas)
DSTRule(America/Dawson)
DSTRule(America/Dawson_Creek)

DSTRule(America/Detroit)
DSTRule(America/Fortaleza)
DSTRule(America/Glace_Bay)
DSTRule(America/Godthab)
DSTRule(America/Goose_Bay)
DSTRule(America/Havana)
DSTRule(America/Indiana/Indianapolis)
DSTRule(America/Indiana/Knox)
DSTRule(America/Miquelon)
DSTRule(America/Montevideo)
DSTRule(America/Noronha)
DSTRule(America/Recife)
DSTRule(America/Resolute)
DSTRule(America/Santiago)
DSTRule(America/Sao_Paulo)
DSTRule(America/Scoresbysund)
DSTRule(America/St_Johns)
DSTRule(Antarctica/Casey)
DSTRule(Antarctica/Davis)
DSTRule(Antarctica/DumontDUrville)
DSTRule(Antarctica/Mawson)
DSTRule(Antarctica/McMurdo)
DSTRule(Asia/Almaty)
DSTRule(Asia/Amman)
DSTRule(Asia/Anadyr)
DSTRule(Asia/Aqtau)
DSTRule(Asia/Ashgabat)
DSTRule(Asia/Baghdad)
DSTRule(Asia/Baku)
DSTRule(Asia/Beirut)
DSTRule(Asia/Colombo)
DSTRule(Asia/Damascus)
DSTRule(Asia/Dubai)
DSTRule(Asia/Gaza)
DSTRule(Asia/Irkutsk)
DSTRule(Asia/Jayapura)
DSTRule(Asia/Jerusalem)
DSTRule(Asia/Kabul)
DSTRule(Asia/Kathmandu)
DSTRule(Asia/Krasnoyarsk)
DSTRule(Asia/Magadan)
DSTRule(Asia/Nicosia)
DSTRule(Asia/Novosibirsk)
DSTRule(Asia/Rangoon)
DSTRule(Asia/Sakhalin)
DSTRule(Asia/Tbilisi)
DSTRule(Asia/Tehran)
DSTRule(Asia/Yakutsk)
DSTRule(Asia/Yekaterinburg)
DSTRule(Atlantic/Canary)
DSTRule(Atlantic/Cape_Verde)
DSTRule(Atlantic/South_Georgia)
DSTRule(Atlantic/Stanley)
DSTRule(Australia/Adelaide)
DSTRule(Australia/Currie)
DSTRule(Australia/Darwin)
DSTRule(Australia/Eucla)
DSTRule(Australia/Lord_Howe)
DSTRule(Europe/Moscow)
DSTRule(Europe/Riga)
DSTRule(Pacific/Apia)
DSTRule(Pacific/Chatham)
DSTRule(Pacific/Easter)
DSTRule(Pacific/Efate)
DSTRule(Pacific/Enderbury)
DSTRule(Pacific/Fakaofo)
DSTRule(Pacific/Fiji)
DSTRule(Pacific/Funafuti)
DSTRule(Pacific/Gambier)
DSTRule(Pacific/Kiritimati)
DSTRule(Pacific/Marquesas)
DSTRule(Pacific/Midway)
DSTRule(Pacific/Norfolk)
DSTRule(Pacific/Pitcairn)
DSTRule(Pacific/Tongatapu)

Programming rule details can be found in the Appendix.

# CHAPTER 4:
# Experiments

This section details the dataset and results of the experiments. There were three dataset used the ACT Data, Israeli Bank Data, and the GTD data. All of the datasets overlapped in 1999. Because of this overlap, only one temporal set of rules was needed.

In conducting the experiments, the Poisson regression was run against each rule independently ensuring the rules did not over-fit the data. While not shown in the data tables below, the Poisson regressions were also accomplished using a stepwise function, meaning rules are added and removed from the list based on significance of modelling effect. Because duplicate rules (like countries observing the same DST changes, were removed before making the stepwise regression, the results where not significantly different. However, if pre-filtering were not done, stepwise regression would have automatically removed duplicate rules. In the future this might be a better method because it eliminates a preprocessing/filtering step.

Again, as discussed in Chapter 3, it is difficult to show a substantial difference in the $P$ value when the $Z$ values are so high; therefore, $Z$ values not $P$ values were used to prioritize the individual rules. In ACTS and Israeli Bank datasets rules were created with the data in mind and have very high $Z$ scores. Because no rule were created for the GTD dataset, these prioritizations have lower $Z$ scores and are less meaningful.

Upon analysis, the findings support the RBI methodology. A rule bank of known temporal data can be used to correlate temporal activity to different data.

## 4.1 NIST Data

This dataset was collected by the National Institute of Standards and Technology (NIST). It was collected from the Automated Computer Time Service (ACTS), which distributes Coordinated Universal Time (UTC) to computer systems via analog modems over ordinary telephone lines, operating mainly from Boulder, Colorado. Figure 4.1 shows ACTS timing requests for 1999. These data were taken from a dataset that had 10 years worth of ACTS data [3].

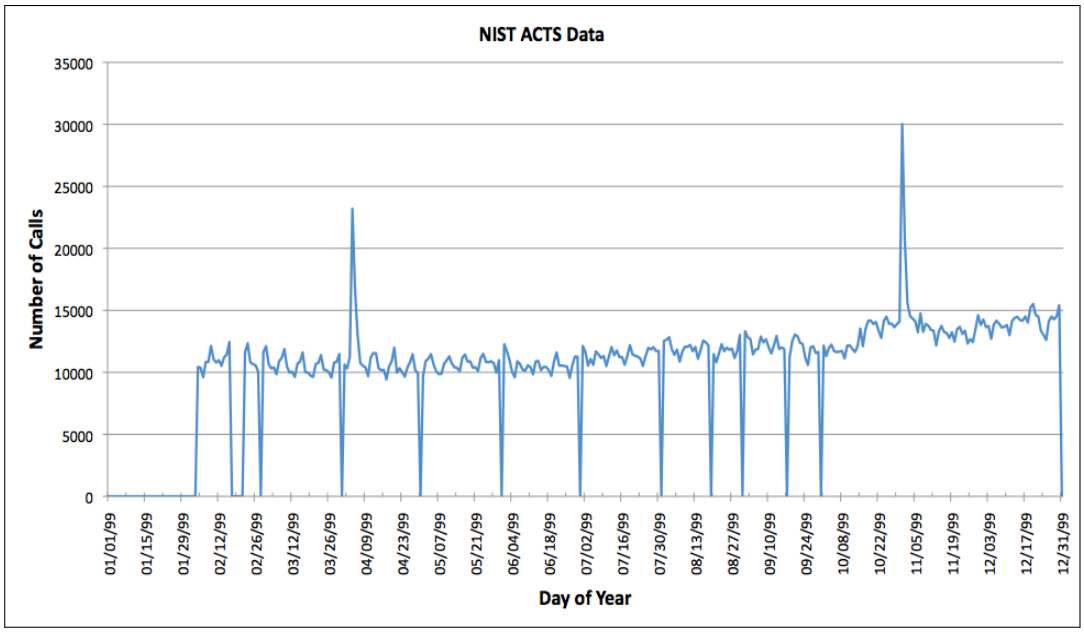


Figure 4.1: ACTS Data—Number of Phone Calls Per Day.

### 4.1.1 Results

As seen in Summary of Findings table from the ACTS Data, *DSTRule.America.Adak* was the most strongly correlated rule with the dates for this rule being 4/4/99 and 10/31/99. Adak is a city in Alaska so this rule represents the DST date changes for most of the United States. Specifically this includes Denver, Colorado where the ACTS are located. As seen in the figure 4.1, the two highest data spikes are on 4/4/99 (23192 calls that day) and 10/31/99 (30024 calls that day). The $Z$ value for this rule was 219.56, which means there is a strong correlation between this rule and where the data originated. The next closest rule *October* has a $Z$ value of 177.29, this make sense as the second DST change is in October. The third rule is *February* with a value of 153.2. At first look this makes no sense, however, the rules are ranked by their absolute $Z$ value. The estimate for this rule is negative, which means the rule demonstrates negative associativity. In other words *February* is the third on the list, but in reality it is strongly anticorrelated to the data. Again as discussed in Chapter 3, $Z$ values of more than 40 have a $P$ value which truncates to zero because 64 bits is not enough precision for values so small.

In this case the RBI methodology works and shows a significant correlation to DST in the United States for this dataset. It is interesting to note that *DSTRule.Africa.Ceuta* and *DSTRule.America.Havana* are only one day from each other. *DSTRule.America.Havana* has a $Z$ value of 130.5.

Table 4.1: Summary of Findings—ACTS Data—Every Rule Independent

| **Rule Name** | **Estimate** | **Std. Error** | **ABS(Z Value)** | **Pr(>\|z\|)** |
|---|---|---|---|---|
| DSTRule.America.Adak. | 0.9586 | 0.0044 | 219.56 | 0.0000 |
| October | 0.2917 | 0.0016 | 177.29 | 0.0000 |
| February | -0.345 | 0.0023 | 153.2 | 0.0000 |
| DSTRule.Africa.Ceuta. | 0.6894 | 0.005 | 138.48 | 0.0000 |
| DSTRule.America.Havana. | 0.6596 | 0.0051 | 130.57 | 0.0000 |
| DSTRule.Africa.Windhoek. | 0.5353 | 0.0054 | 99.7 | 0.0000 |
| Muslim_WorkDays | 0.111 | 0.0012 | 96.14 | 0.0000 |
| MuslimWeekday | 0.1056 | 0.0012 | 90.49 | 0.0000 |
| Israeli_WorkDays | 0.1012 | 0.0011 | 89.11 | 0.0000 |
| DSTRule.Africa.Cairo. | -0.7301 | 0.01 | 72.73 | 0.0000 |
| DSTRule.Asia.Amman. | -0.6142 | 0.0095 | 64.81 | 0.0000 |
| DSTRule.America.Araguaina. | -0.6003 | 0.0094 | 63.78 | 0.0000 |
| Saturday | -0.0963 | 0.0015 | 62.99 | 0.0000 |
| Monday | 0.0851 | 0.0014 | 59.4 | 0.0000 |
| September | 0.105 | 0.0018 | 58.38 | 0.0000 |
| Friday | -0.0811 | 0.0015 | 53.79 | 0.0000 |
| August | 0.0862 | 0.0018 | 48.27 | 0.0000 |
| Tuesday | 0.0648 | 0.0014 | 44.92 | 0.0000 |
| Sunday | 0.0641 | 0.0014 | 44.41 | 0.0000 |
| July | 0.0745 | 0.0018 | 41.49 | 0.0000 |
| DSTRule.Pacific.Fiji. | -0.3351 | 0.0083 | 40.62 | 0.0000 |
| US_ChristmasDayObserved | 0.2946 | 0.0085 | 34.56 | 0.0000 |
| April | 0.0624 | 0.0018 | 34.08 | 0.0000 |
| DSTRule.America.Goose_Bay. | 0.2086 | 0.0063 | 33.08 | 0.0000 |
| US_VeteriansDayObserved | 0.2667 | 0.0086 | 30.85 | 0.0000 |
| US_MemorialDay | -0.1363 | 0.0047 | 28.74 | 0.0000 |
| US_WorkDays | 0.0319 | 0.0011 | 28.72 | 0.0000 |
| DSTRule.America.Godthab. | 0.1409 | 0.0065 | 21.62 | 0.0000 |
| DSTRule.Asia.Tehran. | 0.137 | 0.0065 | 20.97 | 0.0000 |
| | | | | ACTS Data - Continued on next page |

Table 4.1—Summary of Findings—ACTS Data—Continued

| **Rule Name** | **Estimate** | **Std. Error** | **ABS(Z Value)** | **Pr(>\|z\|)** |
|---|---|---|---|---|
| US_Thanksgiving | 0.181 | 0.009 | 20.06 | 0.0000 |
| US_ColumbusDay | 0.168 | 0.0091 | 18.5 | 0.0000 |
| DSTRule.Asia.Jerusalem. | 0.1118 | 0.0066 | 16.9 | 0.0000 |
| Wednesday | -0.0244 | 0.0015 | 16.4 | 0.0000 |
| US_LaborDay | 0.1463 | 0.0092 | 15.94 | 0.0000 |
| DSTRule.Asia.Baghdad. | 0.1019 | 0.0066 | 15.34 | 0.0000 |
| Thursday | -0.0225 | 0.0015 | 15.12 | 0.0000 |
| DSTRule.Australia.Currie. | 0.0998 | 0.0067 | 15.01 | 0.0000 |
| WesternWeekday | 0.0165 | 0.0011 | 14.4 | 0.0000 |
| DSTRule.Antarctica.McMurdo. | 0.0954 | 0.0067 | 14.3 | 0.0000 |
| DSTRule.Pacific.Tongatapu. | 0.1262 | 0.0093 | 13.61 | 0.0000 |
| DSTRule.America.Santiago. | 0.0833 | 0.0067 | 12.42 | 0.0000 |
| DSTRule.Asia.Gaza. | 0.0805 | 0.0067 | 11.98 | 0.0000 |
| US_WashingtonBDay | 0.1049 | 0.0094 | 11.19 | 0.0000 |
| DSTRule.America.Argentina.Buenos_Aires. | -0.0956 | 0.0094 | 10.16 | 0.0000 |
| DSTRule.America.Asuncion. | 0.0283 | 0.0069 | 4.1 | 0.0000 |
| US_IndependanceDayObserved | 0.0299 | 0.0097 | 3.08 | 0.0021 |
| March | -0.0052 | 0.0019 | 2.8 | 0.0051 |
| June | -0.0053 | 0.0019 | 2.8 | 0.0051 |
| DSTRule.Atlantic.Stanley. | 0.0182 | 0.0069 | 2.63 | 0.0086 |
| May | 0.0044 | 0.0018 | 2.38 | 0.0171 |
| January | -19.6305 | 18.8083 | 1.04 | 0.2966 |
| US_BimonthlyPay | -16.5472 | 16.5224 | 1 | 0.3166 |
| DSTRule.Asia.Damascus. | -16.5472 | 16.5224 | 1 | 0.3166 |
| US_NewYears | -16.5445 | 23.3662 | 0.71 | 0.4789 |
| US_MartinLuther | -16.5445 | 23.3662 | 0.71 | 0.4789 |

## 4.2 Israel Bank Center Call Data

This data was downloaded from *http://iew3.technion.ac.il/serveng/callcenterdata*. The data is an archive of all the calls handled by a bank call center in Israel for the year 1999. The original data detailed information about the calls down to the second. However, the data used for this experiment was modified to only count the number of calls per day. During weekdays (Sunday to Thursday), the call center was staffed from 7:00 am to midnight local time. The call center closed at 2:00 pm on Friday and reopened at around 08:00 pm on Saturday. The automated service operated 7 days a week, 24 hours a day [4].

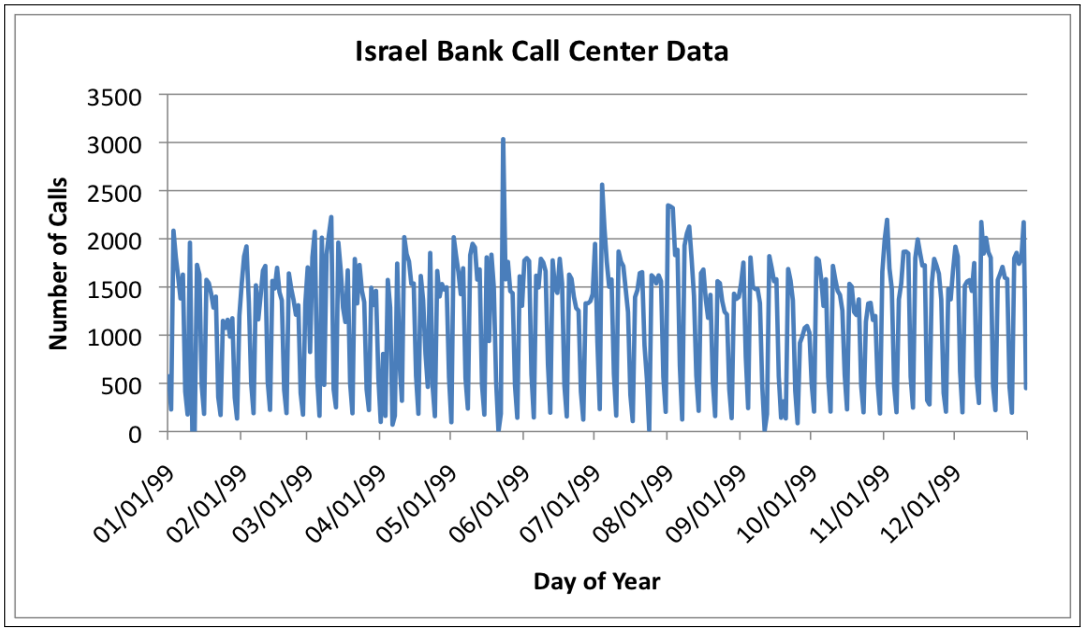


Figure 4.2: Call Center Data—Number of Phone Calls Per Day

### 4.2.1 Results

From the Summary of Findings table for the bank call center data, the most correlated rule was *Israeli_WorkDay*. The $Z$ value was 271.95, again this shows a strong correlation of the rule and where the data originated, which was in Israel. The next rule is *MuslimWeekday*, which the same as the *IsrealiWeekday* rule. Because of the preprocessing step for duplicate rule the *IsrealiWeekday* rule was not examined. *Muslim_WorkDays* is the next and is closely bound to the Israeli data because they have the same workweek rule. The next two rules on the list are *Saturday* and *Friday*. Note both of these rules have negative estimate valuing, meaning there is a negative correlation to the data. This makes sense as the Israeli weekend is Friday and Saturday so one would expect the calling activity to show a negative correlation.

Table 4.2: Summary of Findings—Bank Data—Every Rule Independent

| **Rule Name** | **Estimate** | **Std. Error** | **Z Value** | **Pr(>\|z\|)** |
|---|---|---|---|---|
| Israeli_WorkDays | 1.3061 | 0.0048 | 271.95 | 0.0000 |
| MuslimWeekday | 1.4268 | 0.0053 | 270 | 0.0000 |
| Muslim_WorkDays | 1.3182 | 0.0049 | 266.3 | 0.0000 |
| Saturday | -1.9038 | 0.0098 | 194.87 | 0.0000 |
| Friday | -0.9061 | 0.0061 | 148.11 | 0.0000 |
| Sunday | 0.3806 | 0.0038 | 100.64 | 0.0000 |
| WesternWeekday | 0.3473 | 0.0036 | 95.86 | 0.0000 |
| US_WorkDays | 0.2766 | 0.0034 | 81.48 | 0.0000 |
| Tuesday | 0.2967 | 0.0039 | 76.43 | 0.0000 |
| Thursday | 0.2744 | 0.0039 | 70.2 | 0.0000 |
| Monday | 0.2709 | 0.0039 | 69.22 | 0.0000 |
| Wednesday | 0.2453 | 0.0039 | 62.15 | 0.0000 |
| DSTRule.America.Goose_Bay. | -1.821 | 0.0503 | 36.22 | 0.0000 |
| DSTRule.America.Asuncion. | -1.7669 | 0.0489 | 36.11 | 0.0000 |
| DSTRule.America.Santiago. | -1.7645 | 0.0489 | 36.1 | 0.0000 |
| DSTRule.America.Araguaina. | -1.7065 | 0.0475 | 35.94 | 0.0000 |
| DSTRule.Atlantic.Stanley. | -2.3305 | 0.0648 | 35.94 | 0.0000 |
| DSTRule.America.Godthab. | -1.6776 | 0.0468 | 35.84 | 0.0000 |
| DSTRule.Asia.Baghdad. | -1.3499 | 0.0397 | 33.96 | 0.0000 |
| January | -0.1928 | 0.0058 | 33.03 | 0.0000 |
| September | -0.1648 | 0.0059 | 28.14 | 0.0000 |
| US_BimonthlyPay | -0.7978 | 0.0302 | 26.43 | 0.0000 |
| US_IndependanceDayObserved | 0.5622 | 0.0217 | 25.9 | 0.0000 |
| DSTRule.America.Argentina.Buenos_Aires. | -1.6688 | 0.066 | 25.3 | 0.0000 |
| April | -0.1422 | 0.0058 | 24.52 | 0.0000 |
| DSTRule.Antarctica.McMurdo. | 0.4049 | 0.0166 | 24.33 | 0.0000 |
| August | 0.1193 | 0.0051 | 23.29 | 0.0000 |
| DSTRule.Asia.Tehran. | 0.3737 | 0.0169 | 22.11 | 0.0000 |

Bank Data - Continued on next page

Table 4.2—Summary of Findings—Bank Data—Continued

| **Rule Name** | **Estimate** | **Std. Error** | **Z Value** | **Pr(>\|z\|)** |
|---|---|---|---|---|
| US_MemorialDay | 0.243 | 0.0115 | 21.18 | 0.0000 |
| DSTRule.Africa.Windhoek. | 0.3456 | 0.0171 | 20.17 | 0.0000 |
| US_VeteransDayObserved | 0.4332 | 0.0231 | 18.72 | 0.0000 |
| DSTRule.Australia.Currie. | 0.3188 | 0.0174 | 18.37 | 0.0000 |
| US_ChristmasDayObserved | -0.7159 | 0.041 | 17.46 | 0.0000 |
| US_NewYears | -0.7092 | 0.0409 | 17.36 | 0.0000 |
| DSTRule.America.Adak. | 0.3037 | 0.0175 | 17.36 | 0.0000 |
| October | -0.094 | 0.0056 | 16.8 | 0.0000 |
| DSTRule.Asia.Jerusalem. | -0.4033 | 0.0248 | 16.25 | 0.0000 |
| DSTRule.Asia.Damascus. | -0.3996 | 0.0248 | 16.13 | 0.0000 |
| DSTRule.Africa.Ceuta. | 0.2786 | 0.0177 | 15.73 | 0.0000 |
| DSTRule.Pacific.Tongatapu. | 0.2789 | 0.025 | 11.17 | 0.0000 |
| US_ColumbusDay | 0.2783 | 0.025 | 11.14 | 0.0000 |
| DSTRule.Asia.Amman. | -0.2479 | 0.023 | 10.79 | 0.0000 |
| DSTRule.America.Havana. | -0.2285 | 0.0228 | 10.04 | 0.0000 |
| US_MartinLuther | 0.253 | 0.0253 | 10 | 0.0000 |
| May | 0.0512 | 0.0053 | 9.71 | 0.0000 |
| US_LaborDay | 0.2297 | 0.0256 | 8.97 | 0.0000 |
| DSTRule.Asia.Gaza. | 0.1578 | 0.0188 | 8.4 | 0.0000 |
| US_WashingtonBDay | 0.2158 | 0.0258 | 8.37 | 0.0000 |
| June | 0.0412 | 0.0054 | 7.67 | 0.0000 |
| July | 0.0363 | 0.0053 | 6.86 | 0.0000 |
| DSTRule.Pacific.Fiji. | 0.1255 | 0.0191 | 6.57 | 0.0000 |
| March | 0.0301 | 0.0053 | 5.67 | 0.0000 |
| US_Thanksgiving | 0.1385 | 0.0268 | 5.17 | 0.0000 |
| February | -0.0241 | 0.0057 | 4.23 | 0.0000 |
| DSTRule.Africa.Cairo. | 0.0573 | 0.0198 | 2.9 | 0.0037 |

Again, the RBI methodology showed the tight binding to the Israeli holiday and a decreased correlation to the Israeli weekend. In this case, the RBI methodology seems to be not only

explain what rules correlate but also show which strongly do not correlate. From an analysis perspective, a negative correlation can be just as informative as a strong positive correlation.

## 4.3 Global Terrorism Database

The Global Terrorism Database (GTD) is a open-source database with records starting in 1970 and ending in 2007. There are over 80000 events in the database and every event includes where, when, and how each event occurred. The recorded data was derived from open-source material such as books, journals, and legal documents. The data from 1970-1997 was collected by the Pinkerton Global Intelligence Services (PGIS)–a private security agency. Cases between 1998 and 2007 were developed from a partnership from Center for Terrorism and Intelligence Studies (CETIS), and the Study of Terrorism and Responses to Terrorism (START) groups. Additional events were added from the Conflict Archive on the Internet; the Australian Turkish Media Group, Armenian Terrorism: The Past, Present, the Prospects, by Francis Hyland; the Nation Abortion Federation; and the Further Submission and Responses by the ANC to Questions Raised by the Commission for Truth and Reconciliation 5/12/97.

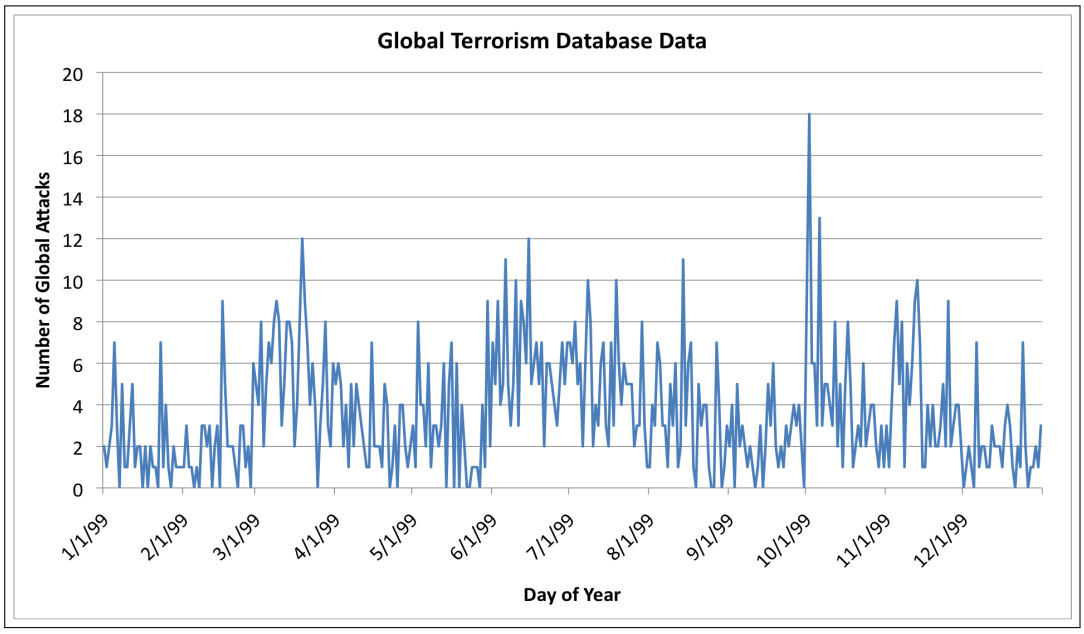


Figure 4.3: Global Terrorism Database Data—Number of Attacks Per Day

### 4.3.1 Results

The GTD data is the control dataset. There were no rules created or designed for this data. If the control is correct, there are no rules that standout as particularly strong for this data.

When compared to the other dataset, results the top rules both had $Z$ scores over 200. This is reasonable of the incidents in 1999 32% of the attacks happened in Algeria, Columbia, India, and Turkey. None of these countries workweeks are coded in the rule set. Twenty-three percent of the attacks happened in the Middle East and Northern Africa this region is predominately Muslim; however, it is not reasonable to believe global or even regional terrorist activity follows a regular workweek pattern unless it is to target populated areas. Again, this would not follow a holiday schedule, per say, and this is reflected in the low Z scores.

The top rule is *June* with a $Z$ value of 7.31, which is much different than the 200 seen in the other rules. This is interesting by looking at the graph both the months of June and March show more consistent activity 4.3. The second rule *DSTRule.America.Argentina.Buenos_Aires* is interesting and points to a important issue when using automated systems such as discussed in this thesis. It seems in 1999 several countries adopted a standard daylight saving change in the later part of the year. The *DSTRule.America.Argentina.Buenos_Aires* is an example of this. This rule has only one day change and it is on 10/2/99, which also happens to correspond to the high spike shown on the graph. If an analysis is not paying attention, he or she might think there is a correlation to global terrorist attacks and daylight saving in Argentina. When in fact this is an anomaly, as it is a rule with only one day. The likelihood of random rules showing a high correlation decrease significantly as the rules become more complex (i.e., more rules than one). The reason similar activity was not seen in the other datasets is easily explained; the temporal rules created were created with these datasets in mind so they should have high $Z$ values and over shadow a single day with a high spike.

Table 4.3: Summary of Findings—GTD Data—Every Rule Independent

| **Rule Name** | **Estimate** | **Std. Error** | **Z Value** | **Pr(>\|z\|)** |
|---|---|---|---|---|
| June | 0.5834 | 0.0799 | 7.31 | 0.0000 |
| DSTRule.America.Argentina.Buenos_Aires. | -1.6189 | 0.2373 | 6.82 | 0.0000 |
| March | 0.5089 | 0.081 | 6.28 | 0.0000 |
| DSTRule.America.Asuncion. | 1.257 | 0.2019 | 6.22 | 0.0000 |
| January | -0.6298 | 0.1301 | 4.84 | 0.0000 |
| DSTRule.America.Araguaina. | 1.03 | 0.2253 | 4.57 | 0.0000 |
| GTD Data - Continued on next page | | | | |

Table 4.3—Summary of Findings—GTD Data—Continued

| **Rule Name** | **Estimate** | **Std. Error** | **Z Value** | **Pr(>\|z\|)** |
|---|---|---|---|---|
| July | 0.3708 | 0.0853 | 4.35 | 0.0000 |
| February | -0.5711 | 0.1332 | 4.29 | 0.0000 |
| September | -0.4815 | 0.1237 | 3.89 | 0.0001 |
| October | 0.2727 | 0.0886 | 3.08 | 0.0021 |
| US_Thanksgiving | 0.9189 | 0.3345 | 2.75 | 0.0060 |
| DSTRule.Asia.Baghdad. | 0.6687 | 0.2687 | 2.49 | 0.0128 |
| DSTRule.Australia.Currie. | 0.6687 | 0.2687 | 2.49 | 0.0128 |
| DSTRule.Pacific.Fiji. | 0.6687 | 0.2687 | 2.49 | 0.0128 |
| May | -0.2465 | 0.1098 | 2.25 | 0.0247 |
| April | -0.2475 | 0.1115 | 2.22 | 0.0265 |
| DSTRule.Antarctica.McMurdo. | 0.5938 | 0.2787 | 2.13 | 0.0331 |
| Saturday | 0.1505 | 0.0748 | 2.01 | 0.0444 |
| DSTRule.Africa.Cairo. | -1.9803 | 1.0004 | 1.98 | 0.0477 |
| DSTRule.Atlantic.Stanley. | -1.2864 | 0.7076 | 1.82 | 0.0691 |
| DSTRule.Asia.Gaza. | 0.513 | 0.29 | 1.77 | 0.0769 |
| Muslim_WorkDays | -0.0926 | 0.0592 | 1.57 | 0.1174 |
| WesternWeekday | -0.0916 | 0.0599 | 1.53 | 0.1265 |
| DSTRule.America.Adak. | -0.8802 | 0.578 | 1.52 | 0.1278 |
| MuslimWeekday | -0.0889 | 0.0598 | 1.49 | 0.1371 |
| Israeli_WorkDays | -0.0835 | 0.0586 | 1.43 | 0.1540 |
| US_ChristmasDayObserved | -1.2844 | 1.0004 | 1.28 | 0.1992 |
| US_IndependanceDayObserved | 0.5111 | 0.4092 | 1.25 | 0.2116 |
| US_VeteriansDayObserved | 0.5111 | 0.4092 | 1.25 | 0.2116 |
| US_BimonthlyPay | -0.5918 | 0.5008 | 1.18 | 0.2373 |
| DSTRule.Africa.Windhoek. | -0.5918 | 0.5008 | 1.18 | 0.2373 |
| US_WorkDays | -0.0637 | 0.0583 | 1.09 | 0.2743 |
| DSTRule.America.Santiago. | 0.3291 | 0.3174 | 1.04 | 0.2998 |
| US_MemorialDay | -0.2559 | 0.2687 | 0.95 | 0.3408 |
| Monday | -0.0668 | 0.0808 | 0.83 | 0.4084 |
| US_NewYears | -0.5905 | 0.7076 | 0.83 | 0.4040 |
| Wednesday | -0.0538 | 0.0804 | 0.67 | 0.5034 |

GTD Data - Continued on next page

Table 4.3—Summary of Findings—GTD Data—Continued

| **Rule Name** | **Estimate** | **Std. Error** | **Z Value** | **Pr(>\|z\|)** |
|---|---|---|---|---|
| August | -0.0681 | 0.1017 | 0.67 | 0.5033 |
| DSTRule.Africa.Ceuta. | 0.223 | 0.3345 | 0.67 | 0.5050 |
| DSTRule.Asia.Amman. | 0.223 | 0.3345 | 0.67 | 0.5050 |
| DSTRule.Asia.Tehran. | 0.223 | 0.3345 | 0.67 | 0.5050 |
| Thursday | -0.0474 | 0.0802 | 0.59 | 0.5550 |
| DSTRule.America.Havana. | -0.1848 | 0.4092 | 0.45 | 0.6516 |
| DSTRule.Asia.Damascus. | -0.1848 | 0.4092 | 0.45 | 0.6516 |
| DSTRule.Asia.Jerusalem. | -0.1848 | 0.4092 | 0.45 | 0.6516 |
| US_LaborDay | -0.1843 | 0.578 | 0.32 | 0.7498 |
| US_ColumbusDay | -0.1843 | 0.578 | 0.32 | 0.7498 |
| DSTRule.Pacific.Tongatapu. | -0.1843 | 0.578 | 0.32 | 0.7498 |
| DSTRule.America.Godthab. | 0.1044 | 0.3546 | 0.29 | 0.7684 |
| DSTRule.America.Goose_Bay. | 0.1044 | 0.3546 | 0.29 | 0.7684 |
| Tuesday | 0.0156 | 0.0784 | 0.2 | 0.8428 |
| Friday | -0.0067 | 0.0784 | 0.09 | 0.9320 |
| Sunday | -0.003 | 0.079 | 0.04 | 0.9695 |
| US_MartinLuther | -14.5878 | 469.3236 | 0.03 | 0.9752 |
| US_WashingtonBDay | -14.5878 | 469.3236 | 0.03 | 0.9752 |

THIS PAGE INTENTIONALLY LEFT BLANK

# CHAPTER 5:
# Future Work

This thesis has shown a simple, reliable methodology to assist analysts correlate temporal rules to datasets of interest. This supervised learn techinique is straightforward and easy to code. The integration of free statical tools like R are available to anyone with internet connectivity and can be used with a little research. There are many relevant and useful research questions that are left unanswered. Future work should involve these areas where this methodology can be expanded.

The first most obvious is to test different data using different temporal granularity. For example, the Israeli bank data has phone records down to the second; it would be interesting to see if you can localize the phone activity based on the observation of local sunset. Rules could be created to localize the celestial observation down to the tens of minutes. Muslim and Jewish holidays both start based on local celestial observation. As stated in Chapter 1, this is of interest because the observed phases of the moon or cycles of the sun are different depending on the observers location on the Earth, these can change the observation of the holiday by minutes or even days. In some cases these differences can be used to localize not only what country, but where in a country. The rules for this thesis do not encompass all cultures, nor do they account for phases of the moon or observed sunrise or sunset, but they could. The rules are left to the imagination of the analysts. Building these rules into the program would provide more flexibility and allow researchers to test different temporal rules.

Another application might be rules for operational tempo. Operating norms of critical enemy units could be helpful. For example, subordinate units often have to report their daily SITREP earlier to higher echelon commands. Therefore, identifying when the lower echelon command report might help in understanding the command's location in a military hierarchy or alert analysts of abnormal behavior.

The data types used in the study where only small samples of different data domains. NPS has a hard drive corpus of third world used drives. This corpus consists of several terabytes of data. By using the rules it should be possible to categorize the location were the data was created using only temporal data found on the drives (i.e., holiday activity for the different counties). Some of the drives were collected from Spain and Mexico which have siestas. Rules created to find this should be able to further increase the confidence of original area of creation.

Another possible test would be to identify common inconsistencies of computer logs based on location and time. Rules may help find changes in log data caused by the computer changing the local time zone based on location. These changes in the logs could be tested on laptops to see if temporal analysis can used to determine the most probable correct time thereby identifing the computers correct location. This is an area of extreme interest in the fields of computer security and forensics.

Yet another possibility would be to combine data collected from different sources (i.e., e-mail timestamps, chat, Web surfing history, and installed programs) to predict future activity or to classify a users persona (i.e., terrorist, hacker, criminal, lawyer, etc.) or uses of the machine.

# CHAPTER 6:
# Conclusions

This paper served three valuable proposes. One, it tested the Poisson regression for rule based data correlation. Two, it demonstrated a current capability to combine human intuition of temporal events and the speed of computers. Finally, this thesis showed the utility of the RBI methodology and gave possible areas of future research.

The results of the methodology are extensible and replicatable to other forms of regression analysis, not just Poisson distributional data. Additionally, the methodology can be used as a framework to explore different datasets, rules, and temporal granularity.

THIS PAGE INTENTIONALLY LEFT BLANK

# REFERENCES


[1] Paul Gray. And bomb the anchovies. *Time*, August 1990.

[2] George Danezis and Richard Clayton. Introducing traffic analysis, 2007. Retrieved November 29 2009. http://www.cl.cam.ac.uk/~rnc1/TAIntro-book.pdf.

[3] Victor Zhang and Michael A. Lombardi. Time and frequency transfer activities at nist. In *40th Annual Precise Time and Time Interval (PTTI) Meeting*. National Institute of Standards and Technology (NIST), 2008.

[4] Ilan Guedj and Avi Mandelbaum. "anonymous bank" call-center data, February 2000. Retrieved October 19 2009. http://iew3.technion.ac.il/serveng/callcenterdata/documentation.pdf.

[5] James F. Allen. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11): 832–843, 1983. ISSN 0001-0782. http://portal.acm.org/ft_gateway.cfm?id=358434.

[6] I. Kant. *Critique of Pure Reason*. Doubleday, Garden City, New York, 1966.

[7] A. Einstein. *Relativity: The Special and General Theory*. Three Rivers Press, 1961.

[8] John Snow. *On the Mode of Communication of Cholera*. John Churchill, New Burlington Street England, 1855.

[9] E. Harrison I. R. Bartky. Standard and daylight-saving time. *Scientific American*, 240(5):46–53, 1979.

[10] Benjamin Franklin. An economical project. Letter to editor of The Journal of Paris, 1784.

[11] September 2007. Retrieved January 18, 2010. http://aa.usno.navy.mil/faq/docs/UT.

[12] Arie Segev and Arie Shoshani. Logical modeling of temporal data. In *SIGMOD '87: Proceedings of the 1987 ACM SIGMOD International Conference on Management of Data*, pp. 454–466. ACM, New York, NY, USA, 1987. http://portal.acm.org/ft_gateway.cfm?id=38760.

[13] E. Allen Emerson. Automated temporal reasoning about reactive systems. In *Proceedings of the VIII Banff Higher Order Workshop Conference on Logics for Concurrency: Structure Versus Automata*, pp. 41–101. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1996.

[14] N. Andrienko, G. Andrienko, and P. Gatalsky. Visualization of spatio-temporal information in the internet. In *DEXA '00: Proceedings of the 11th International Workshop on Database and Expert Systems Applications*, p. 577. IEEE Computer Society, Washington, DC, USA, 2000. http://portal.acm.org/ft_gateway.cfm?id=790409.

[15] Yingjiu Li, Peng Ning, X. Sean Wang, and Sushil Jajodia. Discovering calendar-based temporal association rules. *Data Knowl. Eng.*, 44(2):193–218, 2003. ISSN 0169-023X.

[16] James Bo Begole, John C. Tang, Randall B. Smith, and Nicole Yankelovich. Work rhythms: analyzing visualizations of awareness histories of distributed groups. In *CSCW '02: Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work*, pp. 334–343. ACM, New York, NY, USA, 2002. http://portal.acm.org/ft_gateway.cfm?id=587125.

[17] James Bo Begole, John C. Tang, and Rosco Hill. Rhythm modeling, visualizations and applications. In *UIST '03: Proceedings of the 16th annual ACM symposium on User interface software and technology*, pp. 11–20. ACM, New York, NY, USA, 2003. http://portal.acm.org/ft_gateway.cfm?id=964698.

[18] Jessica Lin, Eamonn Keogh, Stefano Lonardi, Jeffrey P. Lankford, and Donna M. Nystrom. Visually mining and monitoring massive time series. In *KDD '04: Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 460–469. ACM, New York, NY, USA, 2004. http://portal.acm.org/ft_gateway.cfm?id=1014104.

[19] Tanja Falkowski, Jorg Bartelheimer, and Myra Spiliopoulou. Mining and visualizing the evolution of subgroups in social networks. In *WI '06: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pp. 52–58. IEEE Computer Society, Washington, DC, USA, 2006. http://portal.acm.org/ft_gateway.cfm?id=1249048.

[20] Florence Duchêne, Catherine Garbay, and Vincent Rialle. Learning recurrent behaviors from heterogeneous multivariate time-series. *Artificial intelligence in medicine*, 39(1):25–47, 01 2007. http://linkinghub.elsevier.com/retrieve/pii/S0933365706001023.

[21] Joonghoon Lee. Exploring global terrorism data: a web-based visualization of temporal data. *Crossroads*, 15(2):7–14, 2008. ISSN 1528-4972. http://portal.acm.org/ft_gateway.cfm?id=1519393.

[22] Google news timeline, April 2006. Retrieved February 6, 2010. http://newstimeline.googlelabs.com/.

[23] Ramakrishnan Srikant and Rakesh Agrawal. Mining sequential patterns: Generalizations and performance improvements. In *EDBT '96: Proceedings of the 5th International Conference on Extending Database Technology*, pp. 3–17. Springer-Verlag, London, UK, 1996.

[24] Heikki Mannila, Hannu Toivonen, and A. Inkeri Verkamo. Discovery of frequent episodes in event sequences. *Data Min. Knowl. Discov.*, 1(3):259–289, 1997. ISSN 1384-5810. http://portal.acm.org/ft_gateway.cfm?id=593449.

[25] Claudio Bettini, X. Sean Wang, Sushil Jajodia, and Jia-Ling Lin. Discovering frequent event patterns with multiple granularities in time sequences. *IEEE Trans. on Knowl. and Data Eng.*, 10(2):222–237, 1998. ISSN 1041-4347. http://portal.acm.org/ft_gateway.cfm?id=627907.

[26] Jim Hunter and Neil McIntosh. Knowledge-based event detection in complex time series data. In *AIMDM '99: Proceedings of the Joint European Conference on Artificial Intelligence in Medicine and Medical Decision Making*, pp. 271–280. Springer-Verlag, London, UK, 1999.

[27] Behrouz Minaei-Bidgoli and Seyed Behzad Lajevardi. Correlation mining between time series stream and event stream. In *NCM '08: Proceedings of the 2008 Fourth International Conference on Networked Computing and Advanced Information Management*, pp. 333–338. IEEE Computer Society, Washington, DC, USA, 2008. http://portal.acm.org/ft_gateway.cfm?id=1444138.

[28] Michele Berlingerio, Fabio Pinelli, Mirco Nanni, and Fosca Giannotti. Temporal mining for interactive workflow data analysis. In *KDD '09: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 109–118. ACM, New York, NY, USA, 2009. http://portal.acm.org/ft_gateway.cfm?id=1557038.

[29] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *SIGMOD '93: Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pp. 207–216. ACM, New York, NY, USA, 1993. http://portal.acm.org/ft_gateway.cfm?id=170072.

[30] Eamonn Keogh, Selina Chu, David Hart, and Michael Pazzani. Segmenting time series: A survey and novel approach. In *In an Edited Volume, Data mining in Time Series Databases. Published by World Scientific*, pp. 1–22. Publishing Company, 1993.

[31] V. Paxson and S. Floyd. Wide area traffic: The failure of poisson modeling. In *IEEE/ACM Trans. on Networking*, volume 3, pp. 226–244. IEEE/ACM, IEEE, 1995.

[32] Yuval Shahar. A framework for knowledge-based temporal abstraction. *Artif. Intell.*, 90(1-2): 79–133, 1997. ISSN 0004-3702.

[33] Gautam Das, King ip Lin, Heikki Mannila, Gopal Renganathan, and Padhraic Smyth. Rule discovery from time series. In *American Association for Artifical Intelligence*, pp. 16–22. AAAI Press, 1998.

[34] Kin pong Chan and Ada Wai chee Fu. Efficient time series matching by wavelets. In *ICDE '99: Proceedings of the 15th International Conference on Data Engineering*, p. 126. IEEE Computer Society, Washington, DC, USA, 1999. http://portal.acm.org/ft_gateway.cfm?id=847201.

[35] Valery Guralnik and Jaideep Srivastava. Event detection from time series data. In *KDD '99: Proceedings of the fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 33–42. ACM, New York, NY, USA, 1999. http://portal.acm.org/ft_gateway.cfm?id=312190.

[36] Lucia Sacchi, Cristiana Larizza, Carlo Combi, and Riccardo Bellazzi. Data mining with temporal abstractions: learning rules from time series. *Data Mining and Knowledge Discovery*, 15 (2):217–247, 2007.

[37] Fabian Morchen. Unsupervised pattern mining from symbolic temporal data. *SIGKDD Explor. Newsl.*, 9(1):41–55, 2007. ISSN 1931-0145. http://portal.acm.org/ft_gateway.cfm?id=1294302.

[38] Tâm Huynh, Mario Fritz, and Bernt Schiele. Discovery of activity patterns using topic models. In *UbiComp '08: Proceedings of the 10th International Conference on Ubiquitous Computing*, pp. 10–19. ACM, New York, NY, USA, 2008. http://portal.acm.org/ft_gateway.cfm?id=1409638.

[39] Aurelie C. Lozano, Hongfei Li, Alexandru Niculescu-Mizil, Yan Liu, Claudia Perlich, Jonathan Hosking, and Naoki Abe. Spatial-temporal causal modeling for climate change attribution. In *KDD '09: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 587–596. ACM, New York, NY, USA, 2009. http://portal.acm.org/ft_gateway.cfm?id=1557086.

[40] Jungsoon Choi, Montserrat Fuentes, and Brian J. Reich. Spatial-temporal association between fine particulate matter and daily mortality. *Comput. Stat. Data Anal.*, 53(8):2989–3000, 2009. ISSN 0167-9473.

[41] Raymond H. Myers, Douglas C. Montgomery, and G. Geoffrey Vining. *Generalized Linear Models with Applications in Engineering and the Sciences*. John Wiley and Sons, Inc., New York, NY, USA, 2002.

[42] J. Neter M. H. Kutner, C. J. Nachtsheim. *Applied Linear Regression Models*. McGraw-hill Irwin, New York, NY, USA, 2004.

# APPENDIX A:
# Time Rules Code

Listing A.1: Time Rules Code

```
#!/usr/bin/python
"""time_rules.py:

This module contains a list of time rules.
Each rule is a function that takes a python time object and returns
True (in rule) or False (not in rule).

Design:

rule_list - an array that has all of the rules
apply_rules(tval) - applies all of the rules and returns an array of
True/False values apply_rules_to_csv_file(file) - reads a csv file in the
form 'timestamp,count' and returns an array of elements, each in the form:
[timestamp,count,r1,r2,r3...]
"""

import datetime
import time
import calendar
import csv
import os
import sys

# rule_list is the array that we will use to hold the instances of all the rules
# that are being analyzed
rule_list = []

class TimeRule:
    DESCRIPTION = "Abstract rule Class."
    def inRule(self,tval):
        """The default inRule is that it is never in the rule.
        tval is in local time."""
        return False
    def __str__(self):
        return "%s" % (self.__class__.__name__)

class Almanac(TimeRule):
    """ An almanac is a rule which returns true if the given day is part of a
    set. Days is a list of datetime.date objects."""
    def __init__(self,name,days,ignore_year=True):
        self.name = name
        self.days = days
```

```
        self.ignore_year = ignore_year
    def inRule(self,tval):
        for d in self.days:
            if tval.tm_mon  == d.month and tval.tm_mday == d.day:
                if self.ignore_year: return True
                if tval.tm_year == d.year: return True
        return False
    def __str__(self):
        return "Almanac(%s)" % (self.name)

from datetime import date

class Monday(TimeRule):
    """Is True if it is a Monday in the Gregorian Calendar"""
    def inRule(selfself,tval):
        return tval.tm_wday == 0
rule_list.append(Monday())

class Tuesday(TimeRule):
    """Is True if it is a Tuesday in the Gregorian Calendar"""
    def inRule(selfself,tval):
        return tval.tm_wday == 1
rule_list.append(Tuesday())

class Wednesday(TimeRule):
    """Is True if it is a Wednesday in the Gregorian Calendar"""
    def inRule(selfself,tval):
        return tval.tm_wday == 2
rule_list.append(Wednesday())

class Thursday(TimeRule):
    """Is True if it is a Thursday in the Gregorian Calendar"""
    def inRule(selfself,tval):
        return tval.tm_wday == 3
rule_list.append(Thursday())

class Friday(TimeRule):
    """Is True if it is a Friday in the Gregorian Calendar"""
    def inRule(selfself,tval):
        return tval.tm_wday == 4
rule_list.append(Friday())

class Saturday(TimeRule):
    """Is True if it is a MoSaturdaynday in the Gregorian Calendar"""
    def inRule(selfself,tval):
        return tval.tm_wday == 5
rule_list.append(Saturday())

class Sunday(TimeRule):
    """Is True if it is a Sunday in the Gregorian Calendar"""
    def inRule(selfself,tval):
```

```
        return tval.tm_wday == 6
rule_list.append(Sunday())

class January(TimeRule):
    """ Is True if it is January """
    def inRule(self,tval):
        return tval.tm_mon == 1
rule_list.append(January())

class February(TimeRule):
    """ Is True if it is February """
    def inRule(self,tval):
        return tval.tm_mon == 2
rule_list.append(February())

class March(TimeRule):
    """ Is True if it is March """
    def inRule(self,tval):
        return tval.tm_mon == 3
rule_list.append(March())

class April(TimeRule):
    """ Is True if it is April """
    def inRule(self,tval):
        return tval.tm_mon == 4
rule_list.append(April())

class May(TimeRule):
    """ Is True if it is May """
    def inRule(self,tval):
        return tval.tm_mon == 5
rule_list.append(May())

class June(TimeRule):
    """ Is True if it is June """
    def inRule(self,tval):
        return tval.tm_mon == 6
rule_list.append(June())

class July(TimeRule):
    """ Is True if it is July """
    def inRule(self,tval):
        return tval.tm_mon == 7
rule_list.append(July())

class August(TimeRule):
    """ Is True if it is August """
    def inRule(self,tval):
        return tval.tm_mon == 8
rule_list.append(August())
```

```
class September(TimeRule):
    """ Is True if it is September """
    def inRule(self,tval):
        return tval.tm_mon == 9
rule_list.append(September())

class October(TimeRule):
    """ Is True if it is October """
    def inRule(self,tval):
        return tval.tm_mon == 10
rule_list.append(October())

class November(TimeRule):
    """ Is True if it is November """
    def inRule(self,tval):
        return tval.tm_mon == 1
rule_list.append(November())

class December(TimeRule):
    """ Is True if it is December """
    def inRule(self,tval):
        return tval.tm_mon == 1
rule_list.append(December())

class WesternWeekday(TimeRule):
    """Westernworkdays are Monday through Friday"""
    def inRule(self,tval):
        return tval.tm_wday in [0,1,2,3,4]
rule_list.append(WesternWeekday())

class MuslimWeekday(TimeRule):
    """MuslimWeekdays are Sunday through Thursday"""
    def inRule(self,tval):
        return tval.tm_wday in [6,0,1,2,3]
rule_list.append(MuslimWeekday())

class IsraeliWeekday(TimeRule):
    """IsraeliWeekdays are Sunday through Thursday"""
    def inRule(self,tval):
        return tval.tm_wday in [6,0,1,2,3]
rule_list.append(IsraeliWeekday())

class US_NewYears(TimeRule):
    """NewYears occurs the first day of every year"""
    def inRule(self,tval):
        return tval.tm_yday == 1
rule_list.append(US_NewYears())

class US_MartinLuther(TimeRule):
    """ Martin Luther occurs on the third Monday in January """
    def inRule(self,tval):
```

```
        return ((tval.tm_mon == 1 and tval.tm_wday == 0 and tval.tm_yday in \
    range(15,21)))
rule_list.append(US_MartinLuther())

class US_WashingtonBDay(TimeRule):
    """ Washington's Birthday occurs on the third Monday in February """
    def inRule(self,tval):
        return ((tval.tm_mon == 2 and tval.tm_wday == 0 and tval.tm_mday in \
        range(15,21)))
rule_list.append(US_WashingtonBDay())

class US_MemorialDay(TimeRule):
    """ Memorial Day occures on the last Monday in May """
    def inRule(self,tval):
        return ((tval.tm_mon == 5 and tval.tm_wday == 0 and
            ((31 - tval.tm_mday)%7 == 0)))
rule_list.append(US_MemorialDay())

class US_IndependanceDayObserved(TimeRule):
    """ Independence Day occures on July 4 if falls on Sunday then observed on
    Monday if it falls on Saturday then it is observed on Friday """
    def inRule(self,tval):
        return ((tval.tm_mon == 7 and tval.tm_mday == 4 and tval.tm_wday < 5)
                or
                (tval.tm_mon == 7 and tval.tm_mday == 3 and tval.tm_wday == 4)
                or
                (tval.tm_mon == 7 and tval.tm_mday == 5 and tval.tm_wday == 0))
rule_list.append(US_IndependanceDayObserved())

class US_LaborDay(TimeRule):
    """ Labor Day occures on the first Monday in September """
    def inRule(self,tval):
        return ((tval.tm_mon == 9 and tval.tm_wday == 0 and tval.tm_mday in \
    range(1,7)))
rule_list.append(US_LaborDay())

class US_ColumbusDay(TimeRule):
    """ Columbus Day occures on the second Monday in October """
    def inRule(self,tval):
        return ((tval.tm_mon == 10 and tval.tm_wday == 0 and tval.tm_mday in \
    range(7,14)))
rule_list.append(US_ColumbusDay())

class US_VeteriansDayObserved(TimeRule):
    """ Veterans Day occures on November 11th """
    def inRule(self,tval):
        return ((tval.tm_mon == 11 and tval.tm_mday == 11 and tval.tm_wday < 5)
                or
                (tval.tm_mon == 11 and tval.tm_mday == 10 and tval.tm_wday == 4)
                or
                (tval.tm_mon == 11 and tval.tm_mday == 12 and tval.tm_wday == \
```

```
                0))
rule_list.append(US_VeteriansDayObserved())

class US_Thanksgiving(TimeRule):
    """ Thanksgiving occures on November 25th """
    def inRule(self,tval):
        return ((tval.tm_mon == 11 and tval.tm_mday == 25))
rule_list.append(US_Thanksgiving())

class US_ChristmasDayObserved(TimeRule):
    """ Christmas Day occures on December the 25th """
    def inRule(self,tval):
        return ((tval.tm_mon == 12 and tval.tm_mday == 11 and tval.tm_wday < 5)
                or
                (tval.tm_mon == 12 and tval.tm_mday == 10 and tval.tm_wday == 4)
                or
                (tval.tm_mon == 12 and tval.tm_mday == 12 and tval.tm_wday == \
                0))
rule_list.append(US_ChristmasDayObserved())

class US_WorkDays(TimeRule):
    def __init__(self):
        self.US_HOLIDAY_LIST = [US_NewYears(),US_MartinLuther(),
        US_WashingtonBDay(),US_MemorialDay(),US_IndependanceDayObserved(),
        US_LaborDay(),US_ColumbusDay(),US_VeteriansDayObserved(),
        US_Thanksgiving(),US_ChristmasDayObserved()]
        self.WEEKDAY_RULE = WesternWeekday()
    def inRule(self,tval):
        """ It is not a workday if it is a US holiday
        Go through the list of US holidays and see if any of the rules match.
        If so, return false"""
        for r in self.US_HOLIDAY_LIST:
            if r.inRule(tval)==True: return False
        # It is a Workday if the day is a weekday
        return self.WEEKDAY_RULE.inRule(tval)
rule_list.append(US_WorkDays())

class US_BimonthlyPay(TimeRule):
    """Pay is on the 1st and 15th of everymonth when 1st or 15th is on work
    day"""
    def inRule(self,tval):
        return ((tval.tm_yday == 1 and US_WorkDays())
            or
            (tval.tm_yday == 15 and US_WorkDays()))
rule_list.append(US_BimonthlyPay())

# Use the Almanc class to create a Jewish_Holidays_1999 object.
# That object will implement the rules!

Jewish_Holidays_1999 = Almanac("Jewish_Holidays_1999",
[date(1999,2,1),     # Tu_Bishvat
```

```
date(1999,3,2),      # Purim
date(1999,4,1),      # Pesach
date(1999,4,13),     # Yom_HaShoah
date(1999,4,21),     # Yom_HaAtzmaut
date(1999,5,4),      # Lag_BOmer
date(1999,5,21),     # Shavuot
date(1999,7,22),     # Tisha_BAv
date(1999,9,11),     # Rosh_HaShannah
date(1999,9,20),     # Yom_Kippur
date(1999,9,25),     # Sukkot
date(1999,10,2),     # Shemini_Atzeret
date(1999,10,3),     # Simhat_Torah
date(1999,12,4)      # Chanukah
],True)


class Israeli_WorkDays(TimeRule):
    """Return "True" if it is a day on which somebody works in Israel."""
    def __init__(self):
        self.Israel_Holiday_Rule = Jewish_Holidays_1999
        self.WEEKDAY_RULE = IsraeliWeekday()
    def inRule(self,tval):
        # A day is an israel workday if it is workday and not a holiday
        return self.WEEKDAY_RULE.inRule(tval) and \
        self.Israel_Holiday_Rule.inRule(tval)==False
rule_list.append(Israeli_WorkDays())

# Use the Almanc class to create a Muslim_Holidays_1999 object.
# That object will implement the rules!

Muslim_Holidays_1999 = Almanac("Muslim_Holidays_1999",
[date(1999,1,19),     # Id al-Fitr
date(1999,3,28),      # Id al-Adha
date(1999,4,17),      # New Years
date(1999,4,26),     # Ashura
date(1999,6,26),     # Mawlid 1
date(1999,12,9)      # Ramadan
],True)

class Muslim_WorkDays(TimeRule):
    """Return "True" if it is a day on which somebody works in Muslim country."""
    def __init__(self):
        self.Muslim_Holiday_Rule = Muslim_Holidays_1999
        self.WEEKDAY_RULE = MuslimWeekday()
    def inRule(self,tval):
        # A day is an Muslim workday if it is workday and not a holiday
        return self.WEEKDAY_RULE.inRule(tval) and \
        self.Muslim_Holiday_Rule.inRule(tval)==False
rule_list.append(Muslim_WorkDays())

import dstrules
```

```
class DSTRule(TimeRule):
    """This rule returns TRUE if there is a change to DST or ST for a given
    timezone on a given day."""
    def __init__(self,timezone):
        self.timezone = timezone
    def inRule(self,tval):
        return dstrules.is_change_day(self.timezone,tval.tm_year,tval.tm_mon,\
        tval.tm_mday)
    def __str__(self):
        return "DSTRule(%s)" % self.timezone


# Add all of the rules that we haven't seen before
seen_ids = set()
for TZ in dstrules.time_zones():
    rulestr = str(dstrules.timezone_id(TZ))
    if rulestr not in seen_ids:
        rule_list.append(DSTRule(TZ))
        seen_ids.add(rulestr)


# Our handy time-parser. Can parse any time! Really!
time_format_list = [
    "%Y-%m-%d %H:%M:%S",
    "%Y%m%dT%H%M%SZ",
    "%Y-%m-%d-%H:%M:%S",
    "%m/%d/%y"]

def parse_time(s):
    """Parse the string s and return a struct_time."""
    for f in time_format_list:
        try:
            return time.strptime(s,f)
        except ValueError:
            continue

def apply_rules(tval):
    return [r.inRule(tval) for r in rule_list]

if __name__=="__main__":
    from optparse import OptionParser
    global options

    parser = OptionParser()
    parser.usage = "usage: %prog [options] <inputfile>"
    parser.add_option("-l","--list",help="List rules",action="store_true")
    parser.add_option("-t","--test",help="Test rules with a time",\
    action="store_true")
    parser.add_option("-s","--show",\
    help="print the time rule information for a timezone")
    parser.add_option("--tex",help="output rules in LaTeX format",\
```

```
action="store_true")
(options,args) = parser.parse_args()

if options.show:
    print dstrules.timezone_id(options.show)
    exit(0)

if options.tex:
    print "\\def\\TotalRules{%d}\n" % len(rule_list)
    print "\\def\\AllMyRules{"
    for r in rule_list:
        print "\\myrule{%s} " % (str(r).replace("_","\\_"))
    print "}"
    exit(0)

if options.list:
    print "There are %d rules:" % (len(rule_list))
    for r in rule_list:
        print r
    print
    exit(0)
if options.test:
    tval = parse_time(options.test)
    print options.test,"=",tval
    result = apply_rules(tval)
    for i in range(len(result)):
        print rule_list[i],result[i]
```

THIS PAGE INTENTIONALLY LEFT BLANK

# APPENDIX B:
# Timeline Code

Listing B.1: Timeline Code

```
# To change this template, choose Tools | Templates
# and open the template in the editor.

# Please read http://seehuhn.de/pages/pdate

__author__="LCDR Kris Kearton"
__date__ ="$Dec 3, 2009 2:51:34 PM$"

from datetime import datetime,timedelta,date
import csv
import time
import os

def datetimeIterator(from_date=datetime.now(), to_date=None):
    while to_date is None or from_date < to_date:
        yield from_date
        from_date = from_date + timedelta(days = 1)
    return

def dict_to_count(d):
    keys = d.keys()
    keys.sort()
    ret = []
    for date in keys:
        ret.append((date,d[date]))
    return ret

def read_acts(fname):
    """Read the ACTS database from a file and return a dictionary
    where the key is the DATE and the value is the count."""

    ret = []
    for line in csv.reader(open(fname, "U")):
        when  = datetime.strptime(line[0], "%m/%d/%y")
        count = line[1]
        ret.append((when,count))
    return ret

def read_phonecenter(fname):
    headings = None
    tally = {}                          # tally by date
    for line in csv.reader(open(fname, "U"), delimiter="\t"):
```

```
        if not headings:
            headings = line             # get the headings
            continue
        try:
            # The 5th field is the date/time
            when = datetime.strptime(line[5], "%y%m%d") # just use the date
        except ValueError:
            # In one case, there were spaces instead of a tab, so grab the
            #previous field
            when = datetime.strptime(line[4], "%y%m%d") # just use the date

        try:
            tally[when] += 1          # increment the count for that date
        except KeyError:
            tally[when] = 1

    return dict_to_count(tally)

def read_gtd(fname):
    tally = {}                        # tally by date
    for line in csv.reader(open(fname, "U")):
        if line[3]=="":
            continue          # no day in database
        year  = int(line[1])
        month = int(line[2])
        day   = int(line[3])
        if month==0: month=int(line[0][4:6])
        if day==0: day=int(line[0][6:8])
        if year==0 or month==0 or day==0:
            continue
        try:
            when = datetime(year,month,day)
        except ValueError:
            print "bad line:",line
            continue
        try:
            tally[when] += 1          # increment the count for that date
        except KeyError:
            tally[when] = 1
    return dict_to_count(tally)

def make_zone_array():
    """Return a list of all the timezones"""
    print "make zone array"
    timezone_dir=[]

    tz = csv.reader(open("data/tz_zone_only.tab", "U"), delimiter="\t")
    for zone in tz:
        timezone_dir.append(zone[2])

    return timezone_dir
```

```
def compare_to_dst(dates_in, zone_array):
    print "entering compare_to_dist"
    flag = 0
    zone_dict = {}
    count = 0
    zonetemp = "Empty"

    for zone in zone_array:
        os.environ['TZ'] = zone
        time.tzset()

        for year in range(1999,2004):
            for month in range(1,13):
                for day in range(1,28): #left out 29, 30, and 31 as dates as std
                #happens in middle of month
                    for hour in range(0,24):
                        dst_flag = time.localtime(time.mktime((year, month,\
                        day, hour, 0,0,0,0,-1)))
                        temp = dst_flag[8]
                        if temp != flag:
                            t = time.strftime("%Y-%m-%d", dst_flag)
                            for idx in range(0, len(dates_in)):
                                if dates_in[idx]==t:
                                    count = count + 1 #adds only time zones with
                                    #5 or more hits
                                    if count > 4:
                                        zone_dict[zone] = count
                            flag = temp

        count = 0   #resets count after each zone
    print zone_dict
    zone = zonetemp

    return zone_dict

def eval_rule(rule,eventArray):
    """Given a rule, compute the number of events that match the rule and the
    number that don't."""
    rule_in = 0
    rule_out = 0
    for (day,count) in eventArray:
        if rule.inRule(day):
            rule_in += count
        else:
            rule_out += count
    print rule,"in:",rule_in,"out:",rule_out

if __name__ == "__main__":

    """starts here"""
```

```
"""
#vals = read_acts("data/acts_calls99_08.csv")
#vals = read_phonecenter("data/bank_phonecenter_99.txt")
vals = read_gtd("data/globalterrorismdb_0509dist.csv")
dict1999 = {}
eventArray1999 = filter(lambda x:x[0].year==1999,vals)
for (day,count) in eventArray1999:
    dict1999[day.date()] = count

import time_rules

arrayWithTimeTuples = []
poissonArray = []

timelinedata = csv.writer(open("gtd.csv", "wb"))

rows = []
row = ["Day","Count"]
for rule in time_rules.rule_list:
    row.append(str(rule))

rows.append(row)
# Generate a list of all the rows
for day in datetimeIterator(date(1999,1,1),date(2000,1,1)):
    count = dict1999.get(day,0)
    arrayWithTimeTuples.append((day.timetuple(),count))

    row = [day,count]

    for rule in time_rules.rule_list:
        if rule.inRule(day.timetuple()):
            row.append(1)
        else:
            row.append(0)

    rows.append(row)

# Remove duplicate columns
# This function turns any column into a string
def column2string(col_number):
    col = []
    for row in rows[1:]:
        col.append(str(row[col_number]))
    return "-".join(col)

# Now we are going to make clean_rows, which is all of the rows
# without duplicate columns
max_rows    = len(rows)
max_columns = len(rows[0])

new_rows = []
```

```
    for i in range(0,max_rows):
        new_rows.append([])

    seen_columns = set()
    for column_number in range(0,max_columns):
        column_string = column2string(column_number)
        if column_string not in seen_columns:
            # copy this column over
            for rownumber in range(0,len(rows)):
                new_rows[rownumber].append(rows[rownumber][column_number])
            seen_columns.add(column_string)

    # Now write out the cleaned table
    for row in new_rows:
        timelinedata.writerow(row)
    """
    """ends here"""

    """This sction interfaces with R and calculates the Poisson regression and
    builds the Latex table"""
    from rpy2.robjects import r

    r('p<-read.table("/Users/positiveforce1/Documents/Office\ Projects/Thesis/\
    Timeline2/timeline/gtd.csv", sep=",", header = TRUE)')
    r('attach(p)')
    names = r('names(p)')
    print names
    summaryArray=[]

    """ALL RULES ONE AT A TIME"""

    for i in range(2,len(names)):
        first = "r('m1<-glm(Count~"
        last = ", family=poisson)')"
        r('m1<-glm(Count~'+names[i]+', family=poisson)')
        r('library(xtable)')
        out = r('x<-xtable(m1)')
        summaryArray.append(str(r('x<-xtable(m1)')))
        print out
```

THIS PAGE INTENTIONALLY LEFT BLANK

# Referenced Authors

THIS PAGE INTENTIONALLY LEFT BLANK

# Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia
2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California
3. Naval Network Warfare Command
   Norfolk, Virginia
4. USCYBERCOM
   Fort George G Meade, Maryland
5. COMFLTCYBERCOM
   Fort George G Meade, Maryland