Estimating Covariance Models for Collaborative Integrity Monitoring

Liyan Yang

A thesis

in partial fulfillment of the requirements for

the degree of Master of Science

in

Mechanical Engineering

Tufts University

August 2016

Adviser: Dr.Jason Rife

Committee Member: Dr.Robert White

Cross-Department Committee Member: Dr.Usman Khan
Abstract

This thesis introduced a method for modeling non-stationary random errors to enable Collaboration-Enhanced Receiver Integrity Monitoring (CERIM). Collaboration-Enhanced Receiver Integrity Monitoring is a relative new method of GPS integrity monitoring which has multiple advantages compared to conventional Receiver Autonomous Integrity Monitoring (RAIM). The idea is to use information from multiple collaborative receivers to compute a statistic which can be used to monitor system health. CERIM performance depends strongly on the noise covariance model used for the collaborating receivers, but it is not trivial to estimate the noise covariance for receiver teams, particularly for teams operating in an urban environment. This thesis proposes and evaluates a method for estimating a noise covariance model without any external infrastructure or ground truth system. The method is designed to process data even for data sets where the dimension of the CERIM residual vector changes continually (e.g. due to satellites suddenly appearing or disappearing in an urban canyon).
Acknowledgement

This research was done together with my adviser Professor Jason Rife. Thank him for all the help and support he has given me throughout the research. He has taught me a lot during this research period. I really appreciate his guidance. This period of research will be one of my unforgettable memories.

Also, I want to thank Professor Pratap Misra. His course for introduction of GPS guided me into this field. He also gave many instructions and advice during the research.

I would like to thank Jonah Kadoko, Qingwen Liu, Okuary Osechas, and Frank Saunders, who were instrumental in collecting the data analyzed in this research.

I would like to thank the members of ASAR Lab Group in Tufts University. They have given me many advice during the research. They also gave me a lot of instructions for the thesis paper and also the defense presentation.

Finally, thank to my family and friends. My parents, Peijie Yang and Jiehong Feng, you have supported me for my study in Tufts University. The love you have given me is so great.
# Table of Contents

1 Introduction

1.1 GPS signals and errors ............................................. 11
1.2 GPS Positioning method ........................................... 14
1.3 Technologies for Integrity Monitoring .......................... 17
1.4 Function of an Integrity Monitor ............................... 19
1.5 Prior Art ......................................................... 23
1.6 Contribution ....................................................... 24
1.7 Thesis Overview .................................................. 25

2 Modeling and Testing .............................................. 26

2.1 Introduction ....................................................... 26
2.2 Motivation ......................................................... 27
2.3 CERIM Background ............................................... 32
2.4 Covariance Modeling Approach ............................... 41
2.5 Experimental Verification ..................................... 47
2.6 Results .......................................................... 54
2.7 Conclusion ...................................................... 58

3 Comparison with the Theoretical Prediction ................... 60
3.1 Problem Statement .............................................. 61
3.2 Method ......................................................... 62
3.3 Application to GPS data set ............................... 68
   3.3.1 Naive Algorithm .................................... 69
   3.3.2 Baseline Algorithm ................................ 72
   3.3.3 Common Residual Algorithm ....................... 72
3.4 Conclusion .................................................. 75

4 Summary ....................................................... 76
   4.1 Summary .................................................. 76
   4.2 Thesis Contributions .................................. 77
   4.3 Future Work ............................................. 78
List of Figures

1.1 Illustration of trilateration ........................................... 14

1.2 A simulation example of integrity monitoring. Monitor statistics are calculated with time (blue points). A threshold (red vertical line) is set to detect the fault-induced error (circle in the middle). ................ 20

1.3 Simulation example of missed detection. ............................... 21

1.4 Simulation example of false alarm ...................................... 21

1.5 An illustration of the previous work for this research. ............... 23

2.1 The simulated histogram of $m_{\text{CERIM}}/T$ for three cases. In each case the vector $\mathbf{x}$ is assumed to be a normal distributed random vector with zero mean and independent elements. The three cases include (a) all elements with unit variance, no weighting, (b) elements with non-unit variance, no weighting, (c) elements with non-unit variance and weighting matrix applied. .................................. 30
2.2 Locations of four receivers used in experimental data collection (in Medford, MA). Top plot (a) shows route of three car-mounted receivers, each separated by one minute of travel time. Middle plot (b) shows location of a single, stationary receiver. Bottom plot (c) zooms in on the urban canyon in the route, near the bottom right-hand corner of (a). Images based on map data, copyright 2016 Google.

2.3 Histogram of the 3rd element of the parity vector for one of the moving receiver, for the standalone case (left) and for the single-difference case (right).

2.4 Cost function $J$ plotted against two parameters ($\theta_c$ and $a_1$) for the third parameter fixed ($a_0$)

2.5 Histogram of the training data for all four elements of the parity vector $p$ (left column) and regularized vector $y$ (right column).

2.6 Histograms of normalized monitor statistic for each of the three CERIM algorithms. Theoretical predictions (derived from chi-square distributions) are shown as an orange curve superposed over each histogram.

3.1 Histogram of simulated monitor statistics and the theoretical distribution

3.2 CDF plot of simulated monitor statistics and theoretical prediction

3.3 probability-probability plot for simulated monitor statistics and theoretical prediction

3.4 Naive Algorithm’s result in P-P plots of the four different data.

3.5 Baseline Algorithm’s result in P-P plots of the four different data.
3.6 Common Residual Algorithm’s result in P-P plots of the four different data.
Nomenclature

*CERIM*  Collaboration Enhanced Receiver Integrity Monitoring

*DGPS*  Differential GPS

*DSRC*  Dedicated Short Range Communications

*GBAS*  Ground-Based Augmentation System

*GPS*  Global Positioning System

*L1*  Link1

*L2*  Link2

*LORAN*  long range navigation

*PRN*  pseudo-random code

*RAIM*  Receiver Autonomous Integrity Monitoring

*SBAS*  Space-Based Augmentation System
Chapter 1

Introduction

The Global Positioning System, also known as NAVSTAR, is a passive radio navigation system which provides users with position and time information [1]. Since the launch of the first GPS satellite in 1978, the system has been consistently developed and improved. Along with the advancement of the system’s performance, many fields use the system as an auxiliary tool to better achieve their own goals. Farmers now are using GPS combined technology to achieve precision farming which tremendously increases the agricultural efficiency and also enables all-day operation even in low visibility conditions [2]. Surveyors can now accomplish much more work with much less labor with the help of GPS-based data collection [3]. The collected data from the environment can then be displayed on maps or geographic information system (GIS) to help users to make better decisions. More and more areas are enjoying the benefits brought by GPS integrated applications.

One of the new applications of integrated GPS is in the autonomous vehicle field [4, 5]. Autonomous cars use various sensors like GPS or radar to acquire information from the surrounding environment. This information is then processed
by the system, and it is used to control the motion of the vehicle. Currently, vehicles can only use GPS position information to determine the route or achieve some basic autonomous functions. This is because that the current accuracy is only at meter level for a stand alone GPS. However, with the development in carrier phase positioning [6] and also DGPS [7], it is possible that the positioning can be much more accurate. With accuracy less than one meter, functions like lane-keeping may then be achieved directly by using GPS measurements [8].

However, a major problem for this application is the safety. A key feature of self-driving vehicles is that they blindly rely on the measurements from the sensors. Sensors’ errors make the estimated position deviate from the true position. If these errors are large, the deviation may also be large. This may lead the vehicle to make some wrong decisions which end in huge damage like a car crash. Therefore, it is critical to understand the GPS errors and create a way to assure probabilistic bounds on the reliability of the GPS measurements.

In the following sections, more details about the GPS signals, errors, and positioning mechanism are introduced. Then integrity-monitoring concepts will be introduced as methods to verify GPS reliability.

1.1 GPS signals and errors

The design of GPS was based on some similar radio-navigation systems such as LORAN [9] and Transit [10]. These systems use the basic principle of radio waves to transfer information. For GPS, there are currently 31 satellites in the orbit. Generally, a user can see 8 or more satellites at any time of the day (given an unobstructed view of the sky). Each of these satellites are continuously transmitting radio signals
to users. Two radio frequencies in the L-band, Link1(L1) and Link2(L2), are used. This L-band covers frequencies from 1 GHz to 2 GHz. The central frequency of L1 is 1575.42 MHz and the one for L2 is 1227.60 MHz.

The structure of the signal can be divided into three parts. The first part is the carrier signal. It is the radio frequency sinusoidal signal at $f_{L1}$ and $f_{L2}$. The second part is the ranging code, which is a binary code that approximates random noise. This pseudo-random code (PRN) has special properties that allow it to be sent from multiple satellites without interfering each other [11]. This special property can also reveal the time the signal spends in transmission [1]. The third part is the navigation data in binary-coded messages. These messages contain information like the satellite health status, position, or clock bias. This information provided from the PRN code and navigation data is critical for estimating the receivers’ position.

With the information mentioned above, a user receiver can compute the pseudo distance, which is called pseudorange, between the satellite and the receiver. If we subtract the signal’s sending time from the reception time, we can get the signal transmission time. This time can then be multiplied by the speed of light to get the approximate distance between the satellite and the receiver. This distance is not the real distance between the two objects. The reason the term is pseudorange rather than range is that there is a clock offset between GPS time and the user clock. This offset can be estimated if the user receiver has at least four satellites in view.

Positioning errors result from errors in the measured pseudorange. Generally pseudorange errors can be divided into noise and bias. Noise generally refers to a quickly varying error which can be averaged to zero in a short period of time. A bias is a systematic error that could in concept be estimated if sufficient additional
information were available. Biases are likely to remain correlated over long periods of time (many seconds or minutes). A pseudorange can be represented as follows:

\[ \rho = r + c \cdot [\delta t_u - \delta t_s] + I + T + M + \varepsilon \]  

(1.1)

\( \rho \): computed pseudorange  
\( r \): true range between the satellite and the receiver  
\( \delta t_u \): receiver clock bias  
\( \delta t_s \): satellite clock bias  
\( I \): Ionosphere bias (delay)  
\( T \): Troposphere bias (delay)  
\( M \): Multipath bias  
\( \varepsilon \): Thermal noise

The receiver clock bias is caused by the offset between the satellite and the receiver. The satellite’s clock bias is negligibly small. A major error is the propagation error which mainly indicates the Ionosphere and Troposphere error [12, 13]. These two layers of atmosphere change the velocity of signal propagation which creates biases in the measured range. Multipath refers to the phenomenon that the signal reaches the receiver through multiple paths [14]. The error multipath creates is extremely difficult to model and may be correlated over long periods of time. Generally multipath is treated as a source of random noise. The standard deviation of this noise is strongly geometry dependent (with the highest levels of multipath error occurring at the lowest satellite elevation angles). Thermal noise refers to noise in the electronics that is entirely random. Nominally, the mean is zero and the standard deviation is a function of the signal-to-noise ratio for the received PRN code. In this case, it is
possible to model it. The details will be discussed in the later chapters.

1.2 GPS Positioning method

The general idea of GPS positioning is to localize the user receiver with a method similar to trilateration, a classical method for obtaining position using distances from three known points. Fig1.1 sketches out the concept for this method. Suppose we know the position of P1, P2, and P3, just like we know the satellites’ position in GPS. We want to determine the position of an unknown point B. The distances between this unknown point B to the three point P1, P2, and P3 are r1, r2, and r3. We can then draw three circles around each of the known points (the pink, green, and blue circle in the plot) with the radius r1, r2, and r3. In this case, the position can be determined as the intersection of these three circles. The three ranging measurements can solve the three unknown position solution we need (x, y, and z). To find a mathematical way of showing this, suppose there are K satellites

![Figure 1.1: Illustration of trilateration](image)

(K = 3 for now) and suppose the clock offset is zero. Then we can measure the range from each satellite k to the receiver as $\rho_c^{(k)}$. The position of each of these K
satellites is the vector $\mathbf{x}^{(k)}$. To compute the position of the user receiver $\mathbf{x}$, we can first make a guess of the receiver’s position as the initial value $\mathbf{x}_0$. Then an initial pseudorange can be calculated as the following:

$$
\rho_0^{(k)} = ||\mathbf{x}^{(k)} - \mathbf{x}_{(0)}||
$$

(1.2)

Then we can take the difference between the measured and guessed pseudorange as follows:

$$
\delta \rho^{(k)} = \rho^{(k)}_c - \rho_0^{(k)}
$$

(1.3)

This pseudorange difference can then be related to the position difference $\delta \mathbf{x}$. By linearizing (1.1), it can be shown that the two differences are related by a geometry matrix $\mathbf{G}$, where each row is a pointing vector from the estimated position $\mathbf{x}_0$ to a satellite $\mathbf{x}^{(k)}$.

$$
\delta \rho = \mathbf{G} \cdot \delta \mathbf{x}
$$

(1.4)

Since we already know the ranges and satellites’ positions, we can calculate the position difference, as follows:

$$
\delta \mathbf{x} = \mathbf{G}^{-1} \cdot \delta \rho
$$

(1.5)

The procedure can be iterated multiple times until convergence.

A slight modification of this trilateration procedure is needed for GPS, in order to account for the clock offset. In this case, we need one more satellite measurement ($K = 4$). This extra measurement will add one more dimension in $\delta \mathbf{\rho}$ and compute one more solution in $\delta \mathbf{x}$. We will express this extra solution as $b$ which indicates the
receiver’s clock offset.

\[ \delta \mathbf{x} = [x, y, z, b] \]  
(1.6)

Further, we also have to consider the influence of errors. As we mentioned in the last section, there are different kinds of errors contained in the pseudorange. Adding an error term to (1.4) gives the following.

\[ \delta \rho = \mathbf{G} \cdot \delta \mathbf{x} + \varepsilon \]  
(1.7)

Since the value of this error vector is unknown, we can not subtract it from the vector \( \delta \rho \). So the position solution calculated from (1.5) contains errors. It is impossible to find a perfect solution without knowing the errors. However, we may be able to get a relatively best fit solution. If the receiver can see more than 4 satellites, then the geometry matrix \( \mathbf{G} \) in (1.7) will be over-determined. Though we can not directly take the inverse of this over-determined \( \mathbf{G} \) to get the position solution, we can instead use least square method to compute it, as follows:

\[ \delta \mathbf{x} = (\mathbf{G}^T \cdot \mathbf{G})^{-1} \cdot \mathbf{G}^T \cdot \delta \rho \]  
(1.8)

The equation may now be linearized about these new estimates of the user position and clock bias. The solution may be iterated until the change in the estimates is sufficiently small. Each iteration leads to new estimates of the pseudoranges and geometry matrix. This solution is not a perfect one but the one which fits the measurements best. Since it is an over-determined system of equations, \( \delta \rho \) will not be zero at the end of the iteration when \( \delta \mathbf{x} \) goes to zero. Here, we call the final \( \delta \rho \) as residual vector \( \mathbf{r} \). This residual vector is related to the pseudorange errors and it
can further be used in the error analysis.

1.3 Technologies for Integrity Monitoring

For many applications, such as automated driving, one of the most important criteria for application to integrate GPS is safety. The user needs to be informed when GPS information is unreliable due to the sudden onset of a fault condition. This brings up the idea of integrity monitoring.

GPS integrity refers to the ability of the system to provide timely warnings to users when the system should not be used for navigation. Specifically, a navigation system is required to send an alert of any malfunction to users within a given period of time. Loss of integrity can either mean that an unsafe situation is not detected or that it is not detected within the margin of time needed for the user to react. In this case, the automated system will act using an estimated position that contains large errors, which may result in disastrous consequences. To avoid the loss of integrity, we need to monitor the GPS measurements all the time to provide the integrity of the system.

In fact, there are many integrity monitoring methods. Most of them focus on checking whether or not measurement errors have exceeded a specified threshold. Generally, these methods can be divided to either using external or internal measurements.

External monitoring relies on messages provided by the ground stations in known locations [15]. Each received satellite’s pseudorange measurement is compared with the same satellite’s measurement of the ground station. Since the position of the ground station is already known, the errors of the receiver’s measurement
can then be determined through the comparison. If a measurement error exceeds a certain threshold, indicating that it is faulty, then an alert will be sent to the user.

External monitoring is a powerful approach since it can isolate all the errors related to a certain satellite, and also the errors generated in the path its signal travels. However, building and maintaining ground stations incurs a large cost.

Internal monitoring includes approaches like Receiver Autonomous Integrity Monitoring (RAIM) [15, 16]. RAIM establishes system integrity within the user receiver. It uses redundant information contained in the ranging measurements to check the consistency of these measurements. As was mentioned in the earlier section, the $\delta \rho$ in (1.8) does not equal zero at the end of the iteration if the equation system is over-determined. The residual vector $\mathbf{r}$ is defined to be equal to $\delta \rho$ after the last iteration of the position solution. The pseudorange errors project some of the redundant information into this residual vector $\mathbf{r}$. Here, we can take the square of this residual vector and sum the results to get a scalar monitor statistic $m$.

$$m = \mathbf{r}^T \cdot \mathbf{r}$$

This statistic contains error information related to all the satellites. So we can use the magnitude of this value to determine when a large anomalous error occurs. But since this method only uses the receiver’s own data, it can not be too accurate. Further, it also does not have the ability to distinguish errors specific to one user receiver from satellite errors, which would be common to all receivers operating in a local area.

When it is impractical to deploy a network of local reference receivers and when RAIM is not sufficiently sensitive to detect hazardous faults, an alternative in-
tegrity monitoring approach is needed. A recently proposed idea is to share residuals among multiple receivers to perform integrity monitoring. This technique is sometimes called Collaboration-Enhanced Receiver Integrity Monitoring (CERIM) [17]. CERIM shares GPS measurements among local vehicles to enhance integrity accuracy. Each vehicle exchanges GPS measurements with other local vehicles through vehicle-to-vehicle communications, like direct short range communications (DSRC) or a cellular network. By processing this large data set, users can detect errors with a better accuracy. The large data set provides more data with different sources which can also show the common and specific errors separately [18]. In some sense, CERIM is a generalization of RAIM. They both use redundant information to check the consistency of the measurement errors. However, by combining data from many receivers, CERIM provides integrity monitoring with potentially fast alerts, low cost, and high sensitivity – all with no fixed infrastructure required.

1.4 Function of an Integrity Monitor

This section describes how internal integrity monitors like RAIM and CERIM work. In both cases, a monitor statistic similar to (1.9) will be compared with a threshold value to determine whether a fault has happened or not. The general idea is illustrated in Fig1.2. The horizontal axis denotes the time. Ten thousand monitor statistics are computed as time goes by. The vertical axis is the value of these monitor statistics. When the system is operating normally, the monitor statistics will remain in a certain range. But when a fault happens, a large error will be generated. This large error will go into the monitor statistic. Then the statistic’s value will be much larger than the normal range. A threshold can be
used to separate the two situations. In this case, we can monitor the health of the measurements by comparing the magnitude of this monitor statistic with its normal values or the threshold.

Though there are obvious differences between the value of monitor statistics for nominal cases and faulty case in Fig1.2, the fault-induced error is sometimes more difficult to distinguish from nominal noise. Consider the case shown in Fig1.3, which shows two faults. Each fault is a small spike denoted by a circular marker. The fault on the right is only a little bit larger than the nominal values. If the threshold is the same value as it is in Fig1.2, then this faulty case may be missed. No warning will be sent to the user. This is the problem called missed detection. Minimizing missed detections implies that the monitor threshold should be decreased, but decreasing the threshold too much results in a new problem – the problem of false alarm. If the threshold is set too small, as in Fig1.4, then some of the nominal values will be larger than the threshold. A false alarm will be sent to the user although no fault happened.

Setting the threshold thus involves a balance of false alarms (when the thresh-
Figure 1.3: Simulation example of missed detection.

Figure 1.4: Simulation example of false alarm.
old is low) and missed detections (when the threshold is high). In practice, it is desirable to set the threshold as low as possible given a specification on the allowable probability of a false alarm. To do this correctly requires a good model for the statistics of the monitor noise distribution. This is the main focus of this thesis.

What makes estimating the properties of the noise particularly difficult is that the noise distribution changes in time. A random process with a time-varying noise distribution is called a nonstationary process. In the case of GPS, errors are a strong function of the external environment and the geometry of the satellite constellation. For instance, nominal noise is higher for satellites at low elevation angles. For this reason, it is important to regularize the monitor statistic by a time-varying error model that accounts for changes such as the changes in satellite elevation over time. To compensate for time-varying noise, it is useful to scale the monitor statistic (1.9) by the noise covariance matrix. The result is:

\[ m = r^T \cdot Q^{-1} \cdot r \]  

(1.10)

In this way, the elements of vector \( r \) can be normalized to have standard deviation equal to one in nominal cases. (In fact, it doesn’t have to be vector \( r \). Other vectors which can represent redundant information can all be used here. Details will be explained in the later chapter.) By extension, the monitor statistic follows a chi-square distribution in the nominal case. The chi-square distributed scalar helps us to determine a threshold with low false alarm rate.

Therefore, in conclusion, it is critical to find out a good way to model the covariance matrix. This model is essential for use in fault-detection schemes that, to normalize the monitor statistic and ensure that false alarm and missed detection
rates conform to specifications. Currently no clear method exists to estimate the required covariance matrix using residuals from a standard low-cost GPS receiver.

### 1.5 Prior Art

![Figure 1.5: An illustration of the previous work for this research.](image)

This research achieved a conservative estimation of the parity covariance. The research was based on works from many other researchers. Generally, the works can be categorized into those done by researchers from ASAR Lab of Tufts University and those done by other people. A general path is shown in the Fig1.5. The original idea was to find a method to compute the parity covariance for Collaborative Enhanced Receiver Integrity Monitoring (CERIM) (An integrity monitoring method brought by Jason Rife [18], marked as J.R in Fig1.5). To achieve this goal, results from previous research done by other labs were used in this research. The parity covariance was modeled indirectly in this research. This model was built based on the results in [19]. Some of the concepts like using the redundancy to monitor the
integrity or projecting the redundancy into the parity space were originally generated from Receiver Autonomous Integrity Monitoring (RAIM) [16]. The details can be found in [18].

There are also a lot of works done by local researchers (from ASAR lab in Tufts). The idea of overbounding the false-alarm probability of chi-square monitors to determine the threshold of integrity monitoring was originally brought up in [20]. The collaborative car-based data were collected by Jonah Kadoko (marked as J.K in Fig1.5) and other researchers from Tufts University. These GPS measurements were taken from three collaborative receivers from three moving vehicles. Detail descriptions will be introduced in the later chapters.

1.6 Contribution

The primary goal of this thesis is to develop a method to use a low-cost receiver’s data to model the non-stationary random errors’ covariance matrix to enable Collaborative Enhanced Receiver Integrity Monitoring. The low cost receiver means that no online information including Differential GPS (DGPS) are needed. The data does not contain ground truth. Non-stationary means that the value and the dimension of the residual vector is changing through the data collection. The performance of the model has been tested for different CERIM algorithms with real data. The covariance method developed in this dissertation is significant in that it could be deployed offline, for a moving vehicle, to estimate its covariance matrix in changing environments (urban, suburban, rural, etc.).
1.7 Thesis Overview

The remainder of the thesis develops and verifies a novel covariance-estimation method to describe nonstationary GPS ranging errors. The next chapter develops a detailed procedure for covariance modeling. The challenge, different approaches, and result of the covariance modeling have been illustrated. The modeled covariance is then tested in three variants of CERIM algorithms. The data used for testing are those with constant number of satellite-in-view.

Next, the third chapter shows the testing result of varying number of satellite-in-view. The computed monitor statistics are compared with the theoretical prediction in a graph I label the probability-probability graph. This new graph has the ability to show monitor statistic data when the underlying distribution is nonstationary.
Chapter 2

Modeling and Testing

2.1 Introduction

For safety-critical applications of GPS, a capability is needed to warn users when the GPS signal is not reliable [15, 21]. For example in automated vehicle applications, a warning of GPS reliability is needed to ensure that critical systems relying on GPS do not steer the vehicle on to a sidewalk or into the wrong lane of traffic [22, 4].

Currently, there are several integrity monitoring approaches for GNSS. For example, there is a widely used conventional approach called Receiver Autonomous Integrity Monitoring (RAIM) which only uses receivers’ own data for integrity monitoring [16]. There is also an approach called the Space-Based Augmentation System (SBAS) which uses a ground-based reference network and a satellite communication system to relay integrity data to users [23, 24]. An alternative approach called the

---

*The content of this chapter has been written into a paper and published in the ION conference.*
Ground-Based Augmentation System (GBAS) uses integrity and corrections from local reference stations [25]. Existing integrity systems like RAIM, SBAS and GBAS are not necessarily well suited for applications involving a large number of users interacting in an urban area with high-accuracy and short time-to-alert requirements, as in the automated driving applications, where GPS may be used as one of a suite of critical sensors. The disadvantage of RAIM is that the method is not really sensitive because it only uses the receiver’s own data. The disadvantage of SBAS is its long latency (with as much as 6 seconds needed to acquire integrity information). The disadvantage of GBAS is that it is a local system (rather than a regional system), and so deploying GBAS in a large number of cities would incur a very large expense. The disadvantages of these methods can potentially be addressed by CERIM [17]. Before CERIM can be implemented, however, a key challenge is to enable accurate estimation of the residual covariance matrix used in forming the CERIM monitor statistic. Defining a flexible strategy for estimating this covariance matrix is the key contribution of this research.

2.2 Motivation

This section introduces a model problem that motivates the importance of the covariance matrix for CERIM analysis.

The key characteristic of CERIM is that the statistic incorporates data from multiple mobile receivers. Most GPS integrity monitors, including CERIM, function by analyzing a monitor statistic to determine whether or not a fault has occurred. Under nominal conditions the monitor statistic is small, but under fault conditions the statistic becomes large. An alert is thus generated if the monitor statistic ever
exceeds a predetermined threshold indicating a fault is very likely to be present.

To date, three variant CERIM algorithms have been previously introduced [17, 18]. The details of these three algorithms will be reviewed in the next section. For the purpose of motivating the importance of the covariance matrix, it is sufficient to use a general form for the CERIM monitor statistic, one that describes all three algorithms. The general form of the monitor statistic is:

\[ m_{CERIM} = x^T \cdot Q^{-1} \cdot x \]  

(2.1)

Here, the random vector \( x \) describes the residuals for the GPS position solution for one or more receivers. The matrix \( Q \) is the covariance matrix for the random vector \( x \). In the monitor statistic described by (2.1), \( Q \) weights each element of the vector \( x \), effectively decorrelating the input noise to result in more predictable behavior of the monitor statistic \( m_{CERIM} \). If the input \( x \) is properly decorrelated by the inverse of \( Q \), then the monitor statistic is chi-square distributed [26, 20]. Accordingly, the threshold \( T \) can be calculated by the chi-square inverse:

\[ T = P_{X^2}^{-1}(1 - \alpha, DOF) \]  

(2.2)

where \( \alpha \) is the required false alarm probability and \( DOF \) is the degree of freedom for the vector \( x \) [17].

In general, it is not trivial to obtain a good model for the covariance matrix \( Q \). The simplest strategy might be to model \( Q \) with the same noise on all elements (i.e., model \( Q \) as an identity matrix multiplied by a scalar). For the most part in GPS, this is not an accurate model because the standard deviation of each vector...
element may vary based on the elevation angles of the corresponding satellites [27]. A lower-elevation satellite generates larger pseudorange random error and produces larger standard deviation in the corresponding element of vector $\mathbf{x}$.

Weighting by the covariance matrix is important for achieving predictable performance from the monitor. This point is illustrated by an example. Consider a model problem involving a random vector $\mathbf{x} \in \mathbb{R}^4$. For this vector the covariance matrix is $\mathbf{Q} \in \mathbb{R}^{4\times4}$. Let’s consider the simplest case when the vector $\mathbf{x}$ comprises four independent, unit variance elements, all Gaussian distributed. This case can easily be analyzed using a Monte Carlo simulation. Fig2.1(a) illustrates a Monte Carlo simulation in which the monitor statistic is computed $10^7$ times without using a weighting matrix. In other words, the covariance $\mathbf{Q}$ is set to the identity matrix (which happens by coincidence to be the correct covariance in this case). The plot in Fig2.1(a) shows a histogram of the number of times the monitor statistic $m_{\text{CERIM}}$ falls within each bin. In the plot, monitor statistic values are normalized by the threshold $T$, which was obtained by solving (2.2) for an alarm probability $\alpha = 10^{-5}$ and 4 degrees of freedom. The red vertical line indicates the monitor statistic value that equals the threshold ($m_{\text{CERIM}} = T$). As noted in Fig2.1(a), the number of false alarm cases ($m_{\text{CERIM}} > T$) obtained from the Monte Carlo simulation ($0.93 \cdot 10^{-5}$) is very close to the theoretical prediction ($\alpha = 1 \cdot 10^{-5}$).

Now consider an alternate situation where the covariance is still excluded from computing the monitor statistic ($\mathbf{Q}$ is assumed to be $\mathbf{I}$), but where the elements of $\mathbf{x}$ are assumed to have different variances ($\mathbf{Q}$ is not actually equal $\mathbf{I}$). Specifically
Figure 2.1: The simulated histogram of $m_{CERIM}/T$ for three cases. In each case the vector $\mathbf{x}$ is assumed to be a normal distributed random vector with zero mean and independent elements. The three cases include (a) all elements with unit variance, no weighting, (b) elements with non-unit variance, no weighting, (c) elements with non-unit variance and weighting matrix applied.
the actual covariance matrix in this second example is assumed to be $Q_b$, where,

$$Q_b = \begin{bmatrix}
3 & 0 & 0 & 0 \\
0 & 1/3 & 0 & 0 \\
0 & 0 & 1/3 & 0 \\
0 & 0 & 0 & 1/3 \\
\end{bmatrix} \quad (2.3)$$

A histogram of the resulting monitor statistic is shown in Fig2.1(b). The data points are more spread. The false alarm rate $2.48 \cdot 10^{-3}$, which is much higher compared to case (a), where the false alarm rate was only $0.93 \cdot 10^{-5}$ for the same threshold. The problem is that the threshold equation (2.2) is no longer useful, because the unweighted monitor statistic does not follow a chi-square distribution as assumed by (2.2).

The problem is fixed in a third scenario. The third scenario (c) considers a covariance matrix matching that described for scenario (b). However, in contrast with scenario (b), scenario (c) explicitly includes a model of the covariance in computing the monitor statistic through (2.2). As a result of proper decorrelation using the inverse covariance matrix $Q$, the distribution for part (c) is essentially identical to that for part (a).

This comparison of three examples illustrates the great significance of proper decorrelation, so that the process of obtaining a threshold produces a meaningful prediction of false alarm risk.
2.3 CERIM Background

This section provides a more thorough description of three CERIM algorithms, in order to provide a clearer understanding for how the monitor statistic (2.1) might be implemented in practice. The three CERIM algorithms are called the Naive Algorithm, the Baseline Algorithm, and the Common-Residual Algorithm. In many ways, all three algorithms operate similarly. For instance, in all three algorithms, each collaborating user shares GNSS measurements with other users through vehicle-to-vehicle communications. Received data and local data are combined to detect potential faults. A monitor statistic is calculated and compared with a threshold value to decide whether a fault has occurred or not. The primary distinction among the three algorithms is the particular way that the received and local measurement sets are converted into a monitor statistic. These differences are detailed below.

Naive Algorithm

In the Naive Algorithm, a position solution is calculated for each received or local measurement set, and the residual vectors associated with each solution are averaged. Suppose there are $N$ broadcasting receivers (receivers indexed $n = 1$ through $N$) whose measurement sets are communicated to another user (receiver $n=0$). For pseudorange vector $\rho_n$, a position solution can be calculated in the standard fashion, inverting the associated geometry matrix $G_n$ and using Newton-Raphson iterations to solve for the position $x_n$ as a function of $\rho_n$ [1]. A residual vector $r_n$ is obtained when the position solution is converged (as long as there are more than four satellites in view, which makes the geometry matrix over-determined). This residual vector indicates the differences between the measured pseudorange vector $\rho_n$ and
The modeled pseudorange vector $\hat{\rho}_n$.

$$\mathbf{r}_n = \rho_n - \hat{\rho}_n$$  \hspace{1cm} (2.4)

These differences are usually related to measurement errors but can also result from fault-induced biases.

Since the residuals are constrained to be orthogonal to the four columns of $G_n$, the elements in $\mathbf{r}_n$ are not independent [16]. The redundancy in $\mathbf{r}_n$ results in a degenerate (less than full rank) covariance matrix, such that the covariance is not invertible. In order to decorrelate the elements of the residual vector in forming a monitor statistic, as in (2.1), it is thus useful to transform the residuals into an alternate form that eliminates the redundancy. This alternative form of the residual vector is typically called the parity vector $\mathbf{p}_n$. The parity vector is computed from the residual vector by the following equation [28].

$$\mathbf{p}_n = N^T_n \cdot \mathbf{r}_n$$  \hspace{1cm} (2.5)

Here $N_n$ is the null space matrix of the geometry matrix $G_n$ ($N^T_n \cdot G_n = 0$). In a traditional RAIM implementation, the monitor statistic might be defined from the parity vector using a form analogous to (2.1).

$$m_{RAIM} = \mathbf{p}^T \cdot Q^{-1} \cdot \mathbf{p}$$  \hspace{1cm} (2.6)

In CERIM, data are available from multiple receivers, all of which presumably observe the same systematic bias should a satellite fault occur. Thus monitor sensi-
tivity can be increased if an average parity vector is computed, as the averaging operation should (in concept) filter out random noise to provide a more accurate estimate of any systematic bias in the parity vector caused by a fault. Assuming a set of common satellites can be identified across all users (and that the position solution is computed using only those common satellites), then the averaging operation results in $\bar{p}$.

$$\bar{p} = \frac{1}{N+1} \sum_{n=0}^{N} p_n$$  \hspace{1cm} (2.7)

The averaged parity vector can in turn be converted into a monitor statistic $m_{naive}$.

$$m_{naive} = \bar{p}^T \cdot \bar{Q}^{-1} \cdot \bar{p}$$  \hspace{1cm} (2.8)

Here $\bar{Q}$ is the covariance matrix for $\bar{p}$. Given that the noise for each individual pseudorange vector $\rho_n$ is Gaussian with covariance $Q_n$, it is straightforward to show that the covariance for $\bar{p}$ is $Q$.

$$\bar{Q} = \sum_{n=0}^{N} \frac{Q_n}{(N+1)^2}$$  \hspace{1cm} (2.9)

The $(N+1)^2$ term in the denominator results from the averaging operation applied to $\bar{p}$ in (2.7). To detect possible faults using $m_{naive}$, a threshold $T$ is calculated from (2.2). Although the Naive Algorithm is easy to implement, this algorithm is not very practical. The problem is that the collaborating receivers will not generally see the same set of satellites. Especially in urban environments, loss of satellite tracking may happen frequently due to buildings, trees, and terrain features blocking line-of-sight between satellites and individual receivers. As a result, the set of common satellites may be much smaller than the set of satellites seen by any given receiver. Since
the size of the common set tends to shrink as the number of collaborators grows, and since the Naive Algorithm does not monitor integrity for satellites missing from the common set, the algorithm offers little utility for large teams operating in challenging environments.

Baseline Algorithm

In the Baseline Algorithm, each collaborating receiver can contribute a residual vector of arbitrary length, and so no common set of satellites needs to be identified. This is accomplished by creating a concatenated parity vector, which stacks together all of the parity vectors \( p_n \) for each measurement set \( n \) into one large vector \( p_c \).

\[
\begin{align*}
p_c^T &= \begin{bmatrix} p_0^T & p_1^T & \cdots & p_N^T \end{bmatrix} \quad (2.10)
\end{align*}
\]

The monitor statistic \( m_{\text{base}} \) is computed from this concatenated parity vector.

\[
\begin{align*}
m_{\text{base}} &= p_c^T \cdot Q_c^{-1} \cdot p_c \quad (2.11)
\end{align*}
\]

The matrix \( Q_c \) is the covariance of the vector \( p_c \). The matrix \( Q_c \) can be constructed in blocks, where each block is the expected value of a pair of measurement sets, \( m \) and \( n \).

\[
\begin{align*}
Q_c[m,n] &= E[p_m \cdot p_n^T] \quad (2.12)
\end{align*}
\]
If the errors across receivers are independent, then $Q_c$ can be constructed as a block diagonal matrix from the covariance $Q_n$ for each measurement set $n$.

$$Q_c[m, n] = \begin{cases} 
0 & m \neq n \\
Q_n & m = n 
\end{cases}$$  
(2.13)

The zero matrices on the off diagonal are rectangular if the number of parity elements is not the same for receivers $m$ and $n$. In the special case when the errors are independent across measurement sets (and hence across parity vectors), then it is more efficient to rewrite (2.11) as a summation.

$$m_{\text{base}} = \sum_{n=0}^{N} p_n^T \cdot Q_n^{-1} \cdot p_n$$  
(2.14)

The statistic $m_{\text{base}}$ allows the collaborating receivers to check all satellites seen by the collaborating team, regardless of whether or not those satellites are common across the collaborating receivers. This characteristic allows large groups of collaborators to work together effectively, even in urban environments where not all users see the same satellite set. In practice, the Baseline Algorithm has a slight deficiency, however, in that the fault-detection performance does not improve monotonically as the collaborating team grows [17].

**Common Residual Algorithm**

In the Common Residual Algorithm, the monitor statistic is constructed so that the monitor performance always improves when the collaborating team grows. This addresses the deficiency of the Baseline Algorithm, where the signal-to-noise ratio
for any given satellite may grow worse for a satellite not represented in a particular receiver’s measurement set when that measurement set is included in $m_{\text{base}}$, resulting in added noise with no added signal.

The Common Residual Algorithm computes the monitor statistic in the following way [18]:

$$m_{cr} = \hat{c}^T \cdot Q_{\hat{c}}^{-1} \cdot \hat{c} \quad \text{(2.15)}$$

The vector $\hat{c}$ is an all-in-view common residual vector. The matrix $Q_{\hat{c}}$ is its covariance matrix. The common residual vector $\hat{c}$ represents the parity information for a superset of satellites that includes any satellites seen by any of the collaborating receivers.

The key to converting the parity vector for each measurement set $n$ into the all-in-view set is to define a mapping between two null spaces. As seen in (2.5), the null space for each individual measurement set is $N_n$. An all-in-view null space $N_{av}$ can be constructed from a geometry matrix $G_{av}$ that contains a pointing vector to all satellites visible by any of the collaborating receivers. In general, the rows of $N_n$ may be less than the rows of $N_{av}$, because the latter always includes at least as many satellites as the former. The mapping between the basis vectors of the null space for receiver $n$ and the all-in-view null space can then be written as follows.

$$A_n = N_n^T \cdot P_n \cdot N_{av} \quad \text{(2.16)}$$

To help match dimensions, a projection matrix $P$ can be defined where the $(i, j)$
element matches the $i$-th row of $\mathbf{N}_n$ to the $j$-th row of $\mathbf{N}_{av}$.

\[
P = \begin{cases} 
0, & i \text{ and } j \text{ refer to different satellites} \\
1, & i \text{ and } j \text{ refer to same satellite}
\end{cases} \quad (2.17)
\]

The common-mode residual $\mathbf{c}$ can then be related to residual $\mathbf{p}_n$ for any receiver $n$.

\[
\mathbf{p}_n = \mathbf{A}_n \cdot \mathbf{c} + \mathbf{s}_n \quad (2.18)
\]

While the common-mode residual $\mathbf{c}$ is a component of the residual that is the same for all collaborating receivers, the specific residual $\mathbf{s}_n$ includes multipath and other effects that are observed only by receiver $n$. An analogous equation can be defined for the concatenated residual vector $\mathbf{p}_c$, as defined in (2.10).

\[
\mathbf{p}_c = \mathbf{A}_c \cdot \mathbf{c} + \mathbf{s}_c \quad (2.19)
\]

Here the concatenated $\mathbf{A}_c$ matrix is:

\[
\mathbf{A}_c^T = \begin{bmatrix} 
\mathbf{A}_0^T & \mathbf{A}_1^T & \cdots & \mathbf{A}_N^T
\end{bmatrix} \quad (2.20)
\]

and the concatenated $\mathbf{s}_c$ matrix is

\[
\mathbf{s}_c^T = \begin{bmatrix} 
\mathbf{s}_0^T & \mathbf{s}_1^T & \cdots & \mathbf{s}_N^T
\end{bmatrix} \quad (2.21)
\]

It is now possible to estimate the common residual vector $\mathbf{c}$, treating the specific residuals $\mathbf{s}_c$ as noise. The common residual estimate $\hat{\mathbf{c}}$ can be computed as a weighted
least squares solution.

\[ \hat{c} = A^+ p_c \]  \hspace{1cm} (2.22)

The pseudoinverse \( A^+ \) is computed using weights based on the inverse of the covariance matrix \( Q_s \).

\[ A^+ = (A_c^T Q_s^{-1} A_c)^{-1} A_c^T Q_s^{-1} \]  \hspace{1cm} (2.23)

Here the covariance matrix \( Q_s \) describes the noise in the specific residuals (i.e. \( Q_s = \text{E}[s_c \cdot s_c^T] \)).

The specific-residual covariance \( Q_s \) is related (but not identical) to \( Q_c \) as defined in (2.12). An inspection of (2.19) indicates that the concatenated parity vector \( p_c \) is related to a sum of the common mode residual \( c \) and the specific residual \( s_c \). Because \( Q_c \) describes the covariance of the concatenated parity vector, its value depends both on \( Q_s \), the specific-residual covariance, and on \( Q_{av} \), the covariance of the all-in-view vector \( c \).

\[ Q_c = A_n \cdot Q_{av} \cdot A_n^T + Q_s \]  \hspace{1cm} (2.24)

\( Q_{av} \) describes the random noise in common mode error due primarily to troposphere and ionosphere error. Ideally the all-in-view covariance \( Q_{av} \) would be zero if atmospheric errors are removed (e.g. using a real-time, web-based model for atmospheric error). In this thesis, it is reasonable to assume \( Q_{av} \) is the zero matrix, such that \( Q_c \approx Q_s \).

In order to construct the monitor statistic \( m_{cr} \) using (2.15), yet another covariance matrix is needed: \( Q_{\hat{c}} \), which is the the covariance of the estimate \( \hat{c} \).

\[ Q_{\hat{c}} = \text{E}[\hat{c} \cdot \hat{c}^T] \]  \hspace{1cm} (2.25)
The matrix $Q_{\hat{c}}$ is slightly larger than $Q_{av}$ due to estimation error. The idea is that the “natural” (e.g. atmospheric) variation in the common-mode bias given by $Q_{av}$ is augmented by additional covariance $Q_{\epsilon\hat{c}}$, which models noise caused by the estimation process.

$$Q_{\hat{c}} = Q_{av} + Q_{\epsilon\hat{c}}$$

(2.26)

Using the standard methodology for determining the output covariance for a weighted least-squares estimate [29], the following expression can be obtained.

$$Q_{\epsilon\hat{c}} = (A_{\epsilon}^{T} \cdot Q_{s} \cdot A_{\epsilon})^{-1}$$

(2.27)

Modeling the specific residuals $s_{n}$ for each receiver $n$ to be independent from the residuals for other receivers, this equation can be computed more efficiently as follows [17].

$$Q_{\epsilon\hat{c}} = \left( \sum_{n=0}^{N} A_{n}^{T} Q_{s,n}^{-1} A_{n} \right)^{-1}$$

(2.28)

In the case where a web-based model can remove the atmospheric error (such that $Q_{av}$ is negligible), then the monitor statistic of (2.15) can be computed with $Q_{\hat{c}} \approx Q_{\epsilon\hat{c}}$.

By operating on the common-mode residual, $m_{cr}$ avoids the pitfall of $m_{base}$. In other words, sensitivity is increased because noise is not added to $m_{cr}$ unless signal is also added, such that the sensitivity always improves when the number of collaborators $N$ increases.
2.4 Covariance Modeling Approach

Challenge of Covariance Modeling

This section describes why it is not easy to obtain a good model for the parity-vector covariance matrix $Q_n$, as needed to form the monitor statistic for any of the CERIM variants. A data-driven model is desired, especially because the algorithm is intended for operation in a wide variety of conditions, from nearly wide-open suburban boulevards to dense urban canyons, where existing measurement noise models are not well developed.

In concept, a first approach to obtaining a data-driven model is simply to compute the covariance matrix using traditional statistical techniques. Given a set of $L$ random parity vector samples $p_l$, with $l \in [1, L]$, an estimated covariance matrix $\hat{Q}_x$ can be obtained as

$$\hat{Q}_x = \frac{1}{L} \cdot \sum_{l=1}^{L} (p_l - \hat{\mu})(p_l - \hat{\mu})^T$$  \hspace{1cm} (2.29)

given that the estimated mean $\hat{\mu}$ has already been obtained. For parity-vector modeling, this approach is not very useful, since the structure of the parity vector (and its distribution) are constantly changing. To appreciate this, consider the origins of the parity vector. The parity vector results from Newton-Raphson iterations in the GPS solution that solve for small perturbations in the position solution $\delta x$ given the residual error $\delta \rho$ between the pseudorange measurement and pseudorange model.
The perturbations are described by the following equation.

\[
\delta \rho = \begin{bmatrix} G & N \end{bmatrix} \begin{bmatrix} \delta x \\ p \end{bmatrix}
\]  

(2.30)

Here \( N \) is the null space matrix associated with the geometry matrix \( G \). Though the null space matrix needs not be included in the GPS solution, it is critical for defining the parity vector \( p \), because together \( G \) and \( N \) form a complete basis that spans the vector space of possible values \( \delta \rho \). It should further be noted that the null space matrix is not unique, since the null space may be any matrix with columns that are linearly independent from each other and from the columns of \( G \) (that is \( N^T \cdot G = 0 \) and \( N^T \cdot N = I \)).

Newton-Raphson iterations of (2.30) stop when the vector \( \delta x \) converges to the zero vector. Once fully converged, the pseudorange measurement errors translate directly into position (and clock) errors through \( \delta x \) and parity values through \( p \). In other words, if the pseudorange error vector is \( \nu \), then the position/clock error \( \varepsilon \) and the parity vector \( p \) are given by the following equations.

\[
\varepsilon = G^+ \cdot \nu
\]  

(2.31)

\[
p = N^T \cdot \nu
\]  

(2.32)

Here the “+” superscripts after \( G \) in equation (2.31) indicates the matrix pseudoinverse.

The structure of (2.30)-(2.32) reveals the problems with the statistical approach for calculating the covariance of the parity vector. An obvious issue involves the
length of the parity vector. As the number of satellites in view changes (e.g. due
to a blockage when driving past a tall building), then the length of the pseudorange
vector $\delta \rho$ is reduced, and the length of the parity vector, also. Additionally, motion
of satellites through the sky will change the geometry matrix $G$; the nullspace matrix
$N$ and the probability distribution of the parity vector $p$ will change accordingly.
As a final confounding factor, the statistics of the measurement error vector $\nu$ also
change with the position of the satellite in the sky. Taking all of these factors into
account, it is clear that statistics for $Q$ can only be generated using (2.29) over
relatively short periods of time and only if the set of visible satellites is consistent
over that time.

A second approach for obtaining the covariance matrix $Q$ is to derive it by
estimating the measurement-error covariance matrix $R$.

$$R = \mathbb{E}[\nu \cdot \nu^T]$$  \hspace{1cm} (2.33)

The parity-vector covariance matrix $Q_n$ for a receiver $n$ can be constructed as a
function of the associated pseudorange-error covariance $R_n$ using equation (2.32)
from the previous section. Inserting (2.32) into (2.32) gives the following relationship
between $Q_n$ and $R_n$.

$$Q_n = N_n^T \cdot R_n \cdot N_n$$  \hspace{1cm} (2.34)

The approach of estimating $Q_n$ indirectly by first estimating $R_n$ has the advan-
tage that the satellite errors can be estimated independently for the most part (with
the exception of atmospheric errors, which may be correlated). Thus, in concept,$R_n$ can be modeled as diagonal and statistics for each satellite can be compiled
independently. Blockage events do not change the values of $\mathbf{R}_n$; rather, such events simply project $\mathbf{R}_n$ to a lower dimension (where the row and column for the blocked satellite are omitted). This characteristic means that estimation of $\mathbf{R}_n$ is robust to loss-of-lock events (unlike the case in which $\mathbf{Q}_n$ is estimated directly).

One challenge remains in estimating $\mathbf{R}_n$. The problem is that $\mathbf{R}_n$ cannot be obtained unless the full error vector $\mathbf{\nu}$ is known, which would require a ground-truth system. The reason that a ground-truth system is needed is that the error vector $\mathbf{\nu}$ is divided into observable and unobservable components that are linearly independent. The unobservable dimensions of $\mathbf{\nu}$ are the four dimensions that map into the position and clock error $\mathbf{\varepsilon}$. The remaining (observable) dimensions are those that map in $\mathbf{p}$. A separate positioning (and timing) system would be needed to measure the GPS position error $\mathbf{\varepsilon}$ in order to observe all of the dimensions of $\mathbf{\nu}$.

Although it might be possible to obtain ground-truth for a specialized system, a typical low-cost CERIM unit would not have access to ground truth. Thus, if a typical CERIM unit needs to estimate $\mathbf{R}_n$ on-the-fly, new methods are needed that combine available statistical samples of $\mathbf{p}$ with models in order to obtain a reasonable estimate of $\mathbf{R}_n$ using only a low-cost GPS receiver.

Residual-Based Modeling of Measurement Covariance

This section introduces a new procedure for acquiring the covariance matrix by using a combination of modeling and statistics.

As a starting point, assume the measurement error is independent between satellites. Further, for satellite $m$, assume that the measurement error is Gaussian distributed with standard deviation $\sigma_m$. Model the standard deviation to be a
function of satellite elevation \( \theta_m \) and of three user-specified parameters \((a_0, a_1 \text{ and } \theta_c)\) [19].

\[
\sigma_m = a_0 + a_1 \cdot e^{-\theta_m/\theta_c}
\]  

(2.35)

The three parameters are primarily intended to model thermal noise and multipath, and data must be used to tune the specific values of these parameters to model the user environment and hardware. The \(\sigma_m\) models for all satellites \(m\) can be compiled into a measurement-error covariance matrix \(R_n\).

\[
R_n = \begin{bmatrix}
\sigma_1^2(\theta_1) & 0 & \ldots & 0 \\
0 & \sigma_2^2(\theta_1) & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & \sigma_M^2(\theta_M)
\end{bmatrix}
\]  

(2.36)

The diagonal structure of (2.36) implies independence. Note the assumption of independent errors between satellites is only valid if atmospheric errors can be removed (e.g. with SBAS or web-based models).

Equation (2.36) automatically changes in time as satellites move through the sky. Whenever a satellite disappears, whether due to a brief blockage or setting over the horizon, the associated row and column of \(R_n\) can be removed to compute (2.34) for the particular satellite set in view of any individual receiver \(n\).

Data processing is needed to obtain the model parameters \((a_0, a_1, \text{ and } \theta_c)\). I do the following process to obtain a best fit model. The process begins by guessing parameter values and obtaining the associated \(R\) matrix using (2.36). Next the \(Q_n\) matrix is computed for each individual receiver \(n\) using (2.34). The matrix \(Q_n\) is then used to convert the parity vector \(p_n\) into a nominally decorrelated vector \(y_n\).
This mapping is described by the following equation.

\[ y_n = L_n^{-1} \cdot p_n \]  \hspace{1cm} (2.37)

The matrix \( L_n \) is the Cholesky decomposition of matrix \( Q_n \). (It can be computed in Matlab with the command chol.) If the matrix \( L_n \) used in (2.37) correctly models the data, then the covariance of \( y \) will be the identity matrix.

\[ \mathbb{E}[y_n y_n^T] = \mathbb{E}[L_n^{-1} \cdot Q_n \cdot L_n^{-T}] = \mathbf{I} \]  \hspace{1cm} (2.38)

Thus the elements of the nominally decorrelated vector \( y_n \) should all have a variance of 1. In fact, if the \( R_n \) model are correct, then the elements of the vectors \( y_{n,k} \) for all receivers \( n \), for all time steps \( k \), should have unit variance.

With this concept in mind, I will define an optimization problem to obtain modeling parameters. In general, one might define the model \( R_n \) for each vehicle separately by minimizing a cost function \( J(R_n) \). In this thesis, I will assume that the covariance model for all receivers is the same, and so the pseudorange-error model will be identified as \( R \) rather than \( R_n \). This assumption helps makes the most of available data and is reasonable for our data (all receivers identical and all cars operating in similar environments). The cost function is related directly to \( R \), and indirectly to the three modeling parameters in (2.35), as follows.

\[ J(R) = (\hat{\sigma}^2 - 1)^2 \]  \hspace{1cm} (2.39)

In the above equation, the parameter \( \hat{\sigma}^2 \) is the variance of all of the elements of the
vectors $\mathbf{y}_{n,k}$.

$$\hat{\sigma}^2 = \frac{1}{K} \sum_{k=1}^{K} \left( \frac{1}{N+1} \sum_{n=0}^{N} \frac{1}{I_n} \mathbf{y}_n^T \cdot \mathbf{y}_n \right)$$  \hspace{1cm} (2.40)

Here $K$ is the number of time steps in the training set, $I_n$ is the number of elements in the $\mathbf{y}$ vector for receiver $n$, and $N$ is the collaborating receivers excluding receiver zero (the local receiver). Equation (2.40) is implemented as a noncentral second-moment, assuming that the mean is zero. This zero-mean assumption must be verified separately.

By minimizing $J$ over a space of possible parameters $a_0$, $a_1$, and $\theta_c$, the best candidate model can be identified. Although the optimization process could be applied online, this thesis focuses more on verifying the concept and thus the estimation of the sensor covariance was achieved offline in post-processing.

### 2.5 Experimental Verification

This section describes an experimental setup that was used to acquire data to train the $a_0$, $a_1$, and $\theta_c$ parameters for a representative urban setting with low-cost receivers (GlobalSat Technology Corporation BU-353 “puck” style receivers equipped with SiRF Star III chipsets). In all, data were taken on four receivers. Three of them were placed on the top of three separate moving vehicles driving in an urban area of Medford, MA on the campus of Tufts University. The other one was fixed at a nearby grandstand with a clear view of the sky. Each receiver took data once per second. The route of each vehicle was the same with approximately one minute of spacing between each pair of vehicles. The route and the location of the stationary receiver are shown in Fig2.2.
Figure 2.2: Locations of four receivers used in experimental data collection (in Medford, MA). Top plot (a) shows route of three car-mounted receivers, each separated by one minute of travel time. Middle plot (b) shows location of a single, stationary receiver. Bottom plot (c) zooms in on the urban canyon in the route, near the bottom right-hand corner of (a). Images based on map data, copyright 2016 Google.
The data analyzed in this thesis are drawn from a trial lasting 277 seconds. The collected data mainly include the position of the satellites, the speed of the satellites, clock biases, pseudoranges, and other auxiliary parameters. Each receiver saw different numbers of satellites during the entire data collection process. Thus, when the position solution was computed (in post-processing) the length of the parity vector was variable in time. One section of the route, shown in Fig2.2(c), is an urban canyon, where the roadway is surrounded on either side by buildings of three or more stories. The data collected in the urban canyon include many samples affected by severe multipath.

As mentioned above, the CERIM algorithm is best implemented when an online database can be used to provide models for ionosphere and troposphere delay. In the experiments I used a surrogate for this database. Specifically, one of the mobile receivers was used as a reference, and differential pseudorange values for the other three receivers were computed relative to the reference. Though this procedure is problematic for a practical implementation (since differential corrections would likely subtract out the very errors that CERIM is meant to detect), the procedure did provide an ability to analyze operational data representative of the intended use case. Because the first mobile receiver was used as a reference, only three measurement sets were available for CERIM processing.

Importantly, the parity vector elements obtained through the differential correction were very close to zero mean. Note that atmospheric errors were relatively constant over the duration of the trial, and so they would have caused parity elements to exhibit a nonzero mean if not for the use of single differencing as described above. This removal of the mean is clearly shown in Fig2.3, which shows a histogram
of values for the 3rd parity element (for one of the moving receivers, viewing consistent satellites over the first one third of the trial). The figure shows a reduction in mean from 1.94 m to 0.13 m when the single difference is introduced. This shift is representative of other elements, as well.

![Histogram of the 3rd element of the parity vector for one of the moving receiver, for the standalone case (left) and for the single-difference case (right).]

In this thesis, I have chosen to analyze data in which all of the receivers see the same number of satellites. Although the training method described in the prior section does not impose any requirement on the number of satellites seen by each collaborating receiver, the testing process in which models are verified by comparing to data is simplified in this case (see next section).

In order to ensure that receivers viewed a consistent set of satellites in both the training and testing data sets, I divide the trial into three segments. The first
65 seconds were used as a training data set to obtain model parameters. The middle 91 seconds were excluded because tracking of some satellites was interrupted as collaborating receivers passed through the urban canyon region (illustrated in Fig2.2). By comparison, some measurements were lost in the final 121 seconds of the trial, so these data were used to verify the model parameters. The fact that the urban canyon data were excluded clearly has an impact on the values of the model parameters; methods for developing and testing models for urban canyons will be explored in future work.

In this thesis, a direct evaluation of \( J \) was used to search for the best set of parameters. A grid of parameter values \((a_0, a_1 \text{ and } \theta_c)\) was explored and the cost function \((2.39)\) was evaluated at each grid point. The cost function values are illustrated in Fig2.4. Values of \( a_1 \) from 1 to 40m with step length of 1m, and \( \theta_c \) from 1 to 45 degrees with step length 1 degree were tested. The lowest \( J \) value occurred at the point \((a_0 = 1m, a_1 = 12m, \theta_c = 18 \text{ degree})\). These parameters were used to construct the model for the measurement covariance matrix \( R \), which was in turn used to compute the parity covariance matrix and monitor statistic for each of the three CERIM algorithms.

To provide additional insight into the training process, it is useful to compare the elements of the parity vector \( p \) and the regularized vector \( y \). If the model is perfect, then the regularized vector \( y \), as defined by \((2.38)\), would be expected to have an identity covariance matrix, indicating all of the elements of \( y \) should be independent with unit variance. An analysis of model results shows that the regularization process loosely approaches this desired result. Fig2.5 shows the data for the training period, where a histogram of \( p \) values for each of four parity elements

51
Figure 2.4: Cost function $J$ plotted against two parameters ($\theta_c$ and $a_1$) for the third parameter fixed ($a_0$)
(corresponding to eight satellites in view) is shown in the left column. Histograms of \( y \) values for each of the four regularized vector elements are shown in the right column. Note that histograms compile single-difference data from three receivers (with a fourth receiver used as a reference). Standard deviations for each histogram are listed as an inset. The standard deviations of the elements of the \( y \) vector (0.90, 1.48, 0.39, and 0.40 unitless) are close but not identical to the desired value of 1.0. Although the deviations from 1.0 do indicate some deficiency in the model fit, the match is close enough to justify the use of the associated covariance modeling parameters \( a_0, a_1 \) and \( \theta_c \).

Figure 2.5: Histogram of the training data for all four elements of the parity vector \( p \) (left column) and regularized vector \( y \) (right column).
2.6 Results

In order to verify the utility of the proposed process of modeling $Q$, this section presents results of implementing the covariance model in the three CERIM algorithms mentioned earlier. Whereas the data were trained on one set of points (first 65 seconds of the data set) the data were tested on a second set of points (last 121 seconds of the data set).

The process for verifying the distributions was to generate histograms of the monitor statistic from the testing data and to compare those histograms to a theoretical prediction. To make this qualitative comparison, it is first necessary to compute the monitor statistic for each of the three CERIM algorithms at each epoch in the testing data set.

This process of comparing data-derived and theoretical distributions is useful and intuitive; however, the process implies that there is a stationary distribution that describes the monitor statistic. In the real data set, this is not true, since the number of satellites seen by each receiver changes constantly while moving along urban roads. Changes in the number of visible satellites translate to changes in the number of degrees of freedom for the theoretical chi-square distribution and changes in the distribution shape.

In the next chapter, we will seek more flexible verification methods to handle non-stationary distributions, the verification strategy presented here forces us to exclude data from the navigation solution to ensure that the number of satellites seen by each receiver in the test set is consistent. In particular, in analyzing the testing data, I only used single-difference data for two of the three mobile receivers and only for eight of the satellites in view. The majority of epochs in the training data
conformed to these requirements. Mobile receiver 1 was available more consistently than mobile receivers 2 and 3, so in cases when at least 8 satellites from all three receivers were available, either mobile receiver 2 or 3 was excluded at random (to ensure only two measurement sets were used in statistical processing).

**Naive Algorithm**

For each epoch, the monitor statistic for the naive algorithm was computed. The process was to use the \( R \) model obtained in training to compute the parity-vector covariance \( Q_n \) using (2.34). Then from \( Q_n \), the matrix \( \bar{Q}_n \) was computed with (2.9) and the average parity vector with (2.7). Finally, the monitor statistic \( m_{\text{naive}} \) was obtained from (2.8). Since all the averaged parity vectors have length of 4, the degree of freedom for the threshold, calculated by (2.2) for \( \alpha = 1 \cdot 10^{-5} \), is also 4. The histogram of \( m_{\text{base}}/T \) is displayed in Fig2.6(top plot). The testing set includes 53 data points, as shown in the histogram.

The orange curve represents the prediction of the histogram using theory. This prediction is obtained by multiplying the probability density function (PDF) by the number of data points and histogram bin width, so that the integrated area under the curve is equivalent for both the histogram and the theoretical prediction. In the case of the Naive Algorithm the theoretical prediction is generated from a chi-square distribution with 4 degrees of freedom.

The match between the histogram and the theoretical prediction is reasonable given the limited number of points in the testing data set. The good quality of the match means that the \( Q \) model obtained from our training procedure is good enough to assure reasonable performance for CERIM. Certainly the shape of the
modeled $Q$ matrix is sufficiently accurate to avoid the long-tail mis-modeling effects shown in Fig2.1(b).

It should be noted that there is one outlier point in the histogram, one which is quite near the threshold (above $m_{\text{naive}}/T$ of 0.8). This point, likely due to severe multipath, represents an elevated risk of false alarm for the monitor. In order to protect against false-alarms due to multipath, adaptations to all three CRERIM algorithm may be required in the future.

**Baseline Algorithm**

The data used for the Baseline Algorithm are the same as those used in the Naive Algorithm. The monitor statistic for this algorithm is calculated by (2.11). The $p_c$ and $Q_c$ in this equation are computed by (2.10), (2.12), and (2.13). The $Q_n$ in (2.13) is created through the same process as it is in the Naive Algorithm using the elevation angle and the model. The threshold is computed by (2.2) with the same $1 \cdot 10^{-5}$ risk possibility. The parity vector for this algorithm is a concatenated one which makes the degree of freedom equal the summation of all receivers’ degrees of freedom. Thus the threshold and theoretically predicted distribution are constructed using 8 degrees of freedom (as compared to 4 for the Naive Algorithm).

The distribution of $m_{\text{base}}/T$ and the theoretical prediction are shown in the middle plot of Fig2.6. The distribution and the theoretical prediction compare very well. Again, the suggestion is that the modeling process works well for this data set, though some work will be needed in the future to address outliers, like the point that appears at $m_{\text{base}}/T = 0.6$. 
Common Residual Algorithm

The data used in Common Residual Algorithm is the same as the above two algorithms. The common residual vector $\mathbf{c}$ is calculated by (2.22), which depends on the $\mathbf{A}^+$ matrix computed by (2.23). The $\mathbf{A}^+$ matrix in turn depends on an estimate of $\mathbf{Q}_s$. For my analysis, it was assumed that the single difference removed atmospheric effects, such that $\mathbf{Q}_{av}$ is negligible and so $\mathbf{Q}_s \approx \mathbf{Q}_c$, where $\mathbf{Q}_c$ is the same covariance matrix used in the Baseline algorithm. In computing the monitor statistic $m_{cr}$ using (2.15), $\mathbf{Q}_{av}$ was again modeled to be negligible, such that $\mathbf{Q}_{c} \approx \mathbf{Q}_{ce}$, as computed by (2.28).

For the purposes of assessing a threshold and creating a theoretical prediction, the distribution of the monitor statistic was modeled as chi-square with 4 degrees of freedom, since the estimated common residual vector $\hat{\mathbf{c}}$ has 4 elements. In estimating the threshold, a false-alarm probability of $\alpha$ was assumed consistent with prior analyses ($\alpha = 1 \cdot 10^{-5}$). A histogram of monitor statistic values normalized by threshold is shown in Fig2.6 (bottom). The histogram compares relatively well to the theoretical prediction.

In fact, it appears that the results for the Common-Residual Estimation Algorithm are perfectly identical to those of the Naive Algorithm. This suggests a result not previously reported in [17], where the Common-Residual Algorithm was first presented. It is quite possible that the Naive Algorithm is the limiting case of the more general Common-Residual Algorithm, and that the algorithms might be mathematically equivalent when all receivers see the same common set of satellites. This is an interesting mathematical relationship that may be explored further in future work.
The fact that the monitor statistic values $m_{naive}$ and $m_{cr}$ are the same in this case means that the interpretation of the histograms is also the same.

![Figure 2.6: Histograms of normalized monitor statistic for each of the three CERIM algorithms. Theoretical predictions (derived from chi-square distributions) are shown as an orange curve superposed over each histogram.](image)

2.7 Conclusion

This chapter resolves an important challenge in implementing Collaboration Enhanced Receiver Integrity Monitoring. The monitor statistic’s calculation needs
to be weighted so that noise from each element of the input vector can be decorrelated. A method is introduced which solves the issue by estimating covariance using only parity vector data from a low-cost receiver (and no additional ground truth information). The model obtains the parity vector covariance indirectly from a model of the pseudorange noise, which is a function of satellite elevation and three tunable parameters. These parameters are selected by minimizing a cost function designated to ensure consistency between the model and a training data set. The method was implemented on read data collected in an urban driving scenario. A segment of the data were used to train the model and a separate segment were used for verification.
Chapter 3

Comparison with the Theoretical Prediction

As mentioned in the last chapter, the verification methods I used force me to exclude some of the collected data from the navigation solution to make sure that the number of satellites in view is constant. However, this is not going to happen in the real case especially in the urban area. Existing CERIM algorithms have been designed to deal with a variable number of satellites because the number of satellites seen by each user can vary in time. Satellites may be blocked by tall buildings but appear again after few minutes due to the satellites’ motion. Moving receivers might also view different satellites as they move. Because CERIM algorithms are designed to handle such variations, it is important to find a verification method that analyzes all the data collected. This chapter mainly introduces a new way of comparing the data-derived distribution with theoretical prediction when the number of satellites in view for each user is changing. This new method has been used to analyze the nonstationary data collected in this research.
3.1 Problem Statement

In the last chapter, a restriction was set to the data. If there were fewer than eight satellites available for a receiver at an epoch, then the measurement for this receiver at this epoch should not be used. For the last chapter, I did not provide a good technique to compare experiment to theory when the distribution is nonstationary. So the reason to add this restriction was to remove nonstationarity effects from the distribution, so that I could plot experimental data against a reference histogram. The reason the number is 8 is because there are 8 satellites available in most of the epochs. This restriction forced us to abandon many data points which are good to be analyzed. It made the data set to be smaller. Additionally, since the nonstationarity was removed, the result might have become less representative.

Also, because it was rare that all three receivers saw all 8 satellites in view, I only analyzed two receivers at a time in the last chapter. This restriction produced more data points (because it was more likely to find two receivers than three tracking all 8 satellites). In the prior chapter’s analysis, data from receiver 1 were always used. Additional data alternated back and forth between receivers 2 and 3 (depending on measurement availability).

The restrictions on seeing at least 8 satellites on at least two receivers removed the nonstationarity from the data and allowed the monitor statistic histogram to be compared directly to a prediction of that histogram. However, to evaluate the modeled covariance in a more general way, I need to test all epochs. The nonstationary cases should not be avoided.

In summary, it is necessary to represent the monitor statistic and the theoretical prediction in a new form that allows a comparison to be made for a nonstationary
random process. Therefore, in this section I introduce an alternative verification technique that works both in the stationary and nonstationary case. It allows the number of available satellites to vary, which means that all the collected data can be used in the research. This also makes using 3 receivers possible. The more provided data points give more accurate results in the end.

### 3.2 Method

Currently, for a nonstationary data set, the problem is that random samples are each associated with a different probability distribution from a different moment in time. In the case of CERIM, each monitor statistic sample may be associated with a different chi-square distribution (featuring a different number of degrees of freedom). So to solve the problem, I propose to analyze the data in a different way. Specifically, I will suppress the value of the monitor statistic for each sample (as given on the horizontal axis of the histogram of Fig2.6 or of Fig3.1, below). Instead, I propose to plot an empirical cumulative distribution function directly against the theoretical cumulative distribution function, such that both the horizontal and vertical axes of the comparison plot have units of probability. I will label this a probability-probability plot (P-P plot). When the data are nonstationary (and when a model is available for the probability distribution at each moment in time), the proposed process can be applied to each sample in a population individually, making it possible to evaluate the overall quality of the theoretical model for the nonstationary process.

In order to understand this type of plot, I will first describe its application to a stationary data set, generated by simulation. The simulation has 10 thousand points. Each point has a vector $\mathbf{x}$ with 4 elements. Each element is standard normal
distributed. A procedure like (2.1) has been done to generate a scalar monitor statistic from the random vector samples. Since the elements of the vector \( \mathbf{x} \) are already standard normal distributed, the inverse covariance matrix can simply be identity matrix. In this case, the computed monitor statistic is chi-square distributed with degree-of-freedom equal to 4. However, we may want to simulate a circumstance which can be closer to the result in Fig2.6. Look back into the result section in Chapter 2, it seems that the distributions of \( m \) are more pressed to the left compared to the theoretical plot. This occurred because an outlier, perhaps caused by multipath, caused an overestimation of the variance. When the monitor statistic was computed (divided by variance), it came out slightly low for typical samples. In other words, the model is adequate for CERIM applications, but nonetheless contains a clear systematic error.

This effect related to mis-estimation of variance can be modeled in my simple Matlab simulation by scaling the true standard deviation (to \( \sigma = 0.6 \)) while leaving the assumed theoretical distribution with a higher standard deviation (to \( \sigma = 1.0 \)). Thus, the theory-based weighting matrix is still the identity matrix. The histogram is shown in Fig3.1. Here, the simulated distribution locates more on the left comparing to the theoretical probability density function (PDF) plot, just like what is shown in Fig2.6.

To convert the two variables, the monitor statistic and the degree of freedom, into the variable of probability, I need to first make a Cumulative Distribution Function (CDF) plot for the monitor statistics. Let’s first make an assumption that the degree-of-freedom is constant for every sample. Then the CDF can show the probability of a computed monitor statistic to have its value below a certain
Figure 3.1: Histogram of simulated monitor statistics and the theoretical distribution

amount. The plot is shown in Fig3.2. The procedure of getting the empirical CDF is to first sort the monitor statistics to make them in an order from small to large. A probability of $1/N$, where $N$ is the number of samples in the data set, is assigned to each measured statistic. The empirical CDF plots accumulated probability for each value of the sorted monitor statistic, where the accumulated probability increments from $1/N$ (for the first sample) up to 1 (for the final sample). Here, the horizontal axis is the value of the monitor statistic. The vertical axis is the CDF probability corresponding to the statistic value. The blue curve is the empirical CDF of the simulated data points. The empirical CDF is an alternate representation of the histogram shown in Fig3.1. To be precise, it is the integral of the histogram from Fig3.1 (where the bin size is allowed to be arbitrarily narrow, such that each histogram bin holds only one sample). The red curve is the theoretical CDF, which is the integral of the PDF model used to plot the red theoretical curve in Fig3.1.

One advantage of the plot comparing the empirical CDF to the theoretical
CDF is that the difference between the two plots is amplified. The scaling difference between the empirical data and the theoretical curve is now seen as a clear offset between CDFs. Tracing a vertical line through the plot, it is easy to notice that the probabilities correspond to the statistic values are different for the two curves. This plot helps us to get a better comparison between the data results and the theoretical ones.

For the CDF plot in the Fig3.2, one of the most important drawbacks is that the plot still cannot compare samples drawn from a time-varying probability distribution. A large monitor statistic may have a low CDF probability if the standard deviation is large and vice versa. If a theoretical model of the nonstationary process is available, then one way to assess the quality of the model is to generalize Fig3.2, by using the nonstationary model to assess the likelihood of each individual sample. This is the basis for the probability-probability plot proposed in this chapter. Specifically, the CDF at each epoch is inverted to convert the sample value into a
probability that the sample occurred. Those probabilities are then sorted in order to form a theoretical CDF associated with the sample population (effectively normalizing out nonstationary effects). An empirical CDF can be created by assigning each sample a probability of $1/N$ and accumulating those probabilities. Plotting the empirical CDF against the theoretical CDF results in the final probability-probability plot for the nonstationary process.

For the simulated data (and also for the GPS data in our dataset), the nonstationary model is chi-square, where the distribution’s degrees-of-freedom parameter varies with the number of satellites seen by each collaborating receiver. To convert each monitor statistic sample to a theoretical probability, I used the chi-square CDF below.

$$p = 1 - \frac{1}{\Gamma(k/2)} \Gamma\left(\frac{k}{2}, \frac{x}{2}\right)$$

(3.1)

This equation is the integral of the Probability Density Function. The variable $k$ denotes the degree-of-freedom. $x$ is the monitor statistic. $\Gamma$ is the gamma function. In practice, this CDF was computed using the chi2cdf command in Matlab.

A P-P plot of the simulated data is shown in Fig3.3. Here, the horizontal axis is the sorted theoretical CDF of the monitor statistic. The vertical axis provides the empirical CDF values. Because the simulated process is stationary, with a consistent probability distribution, the new plot is equivalent to plotting the value of the empirical CDF from Fig3.2 against the value of the theoretical CDF (for the same monitor statistic). The new plot suppresses monitor statistic information, which was seen on the horizontal axis of Fig3.2. The resulting probability-probability curve is shown in blue in Fig3.3. The red curve is the reference line. The reference line is what one would expect to see if the theoretical and empirical CDFs were identical.
In other words, for a good model, the blue curve would be expected to match the red curve, with $y = x$. For the simulated data, since the computed monitor statistics are not normalized correctly, the blue curve should show some deviation. Since the data are smaller than predicted, the blue curve is generally above the reference line, indicating that the theoretical probabilities for each error are conservatively over-predicted. If the blue curve dipped below the red reference line, it would be indicative that the theoretical model had under-predicted the likelihood of larger errors.

Comparing with the former plots, this plot successfully solved the problem of visualizing a nonstationarity of the random process. It also makes the comparison between the simulated and theoretical data to be clearer.
3.3 Application to GPS data set

As I talked about in the last section, here I use probability-probability (P-P) plots to analyze the performance of the covariance modeled for our multi-receiver GPS data set. Just like the result section in the last chapter, plots need to be drawn for each of the three algorithms. First of all, it is necessary to have the P-P plot for the same condition of last chapter. So it means that this plot is drawn under the condition that only using the data from two receivers (always using receiver 1 and switching between 2 and 3) and the epochs with exact 8 satellites. The reason the first plot follows the same restriction of the last chapter is that a control group is needed. With the control group, the consequence of the restriction can be known so that we can evaluate the value of the last chapter’s result. Again, this plot also connects the P-P plot with the histogram in Fig2.6 which helps us to study the histogram better.

In the remainder of this section, I will generate four plots for each of the CERIM algorithms considered in this thesis. The first of the four plots will analyze samples from only the testing data, considering two receivers per epoch, each with exactly eight satellites in view. The second of the four plots will analyze samples from the testing data, considering two receivers per epoch, each seeing an arbitrary number of satellites in view. The third will analyze samples from a combined data set with data from both the training and testing trials, considering the two receivers per epoch, each seeing an arbitrary number of satellites. The fourth of the plots will analyze samples from the combined data set, considering three receivers per epoch, each seeing an arbitrary number of satellites.

For all of the plots, the same model parameters ($a_0, a_1, \theta_c$) were used to obtain
the theoretical CDF. The values of these parameters were identical to those obtained in Chapter 2. It should be noted that training data were used (in addition to testing data) for the verification in subplots (c) and (d). Although training data would not normally be used in model testing, the sample population for the experiment was very limited in size. For this reason, the latter plots combined the training and testing data to reduce statistical variation effects associated with small population size.

### 3.3.1 Naive Algorithm

For the nonstationary case of Naive Algorithm, the averaging process of (2.7) or (2.9) becomes more complicated. This process requires parity vectors or covariances to have the same dimension among the receivers. Otherwise they can not be added up. Thus I used the intersection set of the satellites. This intersection set contains satellites seen by all the receivers. Satellites not seen by all receivers are not included in the intersection set. In this way, the dimension of the parity vector $p_n$ in (2.7) is all the same for each receiver. The error information contains in each $p_n$ element is projected from the same satellites’ measurements. These $p_n$ can now be added. The same idea goes for covariance $Q_n$.

Four plots were created in order to analyze the Naive Algorithm. They are shown in Fig3.4. The horizontal axis denotes the theoretical CDF for the nonstationary data (based on a time-varying chi-square model). The vertical axis shows the empirical CDF. Subplot (a) is the P-P plot for 2 receiver’s data of the testing trial. Restriction of only using 8 satellites’ measurement is on for this plot. Subplot (b) is the P-P plot for 2 receivers’ data of the testing trial. There is no restriction
on number of satellites seen by each receiver. Subplot (c) is the 2 receivers’ data for the combined set of training trial and testing trial. Subplot (d) shows the result of 3 receivers’ data for the combined set of training trial and testing trial.

Figure 3.4: Naive Algorithm’s result in P-P plots of the four different data.

The blue probability-probability curves in both plot (a) and (b) are located more on the upper-left relative to the reference line. This shows that the monitor statistics for these two cases are smaller than the theoretical prediction. If we compare between these two plots, we may conclude that the restriction seems do not affect the result very much. Both of the curves are off in some degree. However,
the curve with no restriction is generally closer to the prediction. This might result from the larger data set of (b). More epochs are allowed in (b). The user saw less satellites during these epochs which may indicate higher levels of noise due to multipath. Noisier data produce larger monitor statistics. This might be the reason that the curve in (b) is closer to the prediction. The plot (a) has eliminated some of the nominal but noisy points which makes the curve deviates.

If compare the plot (b) and (c), we can clearly see the improvement. The only difference between (b) and (c) is the size of the data set. A larger data set makes the result more stable. This might be the reason why the curve gets much closer to the prediction. Another explanation may be that the model is a better fit for the training data than the testing data (perhaps because the model parameters are changing in time in a way that is not modeled simply by elevation effects). Because the training data are included in the sample population, it is perhaps not surprising that the model more closely matches the data. Further experimentation will be needed in the future to distinguish between effects caused by the small size of the sample population and effects caused by mixing training data into the testing data set.

Plot (d) is even closer to the prediction compared to (c). The same question persists as to whether the better match is due to reduced sensitivity to the size of the sample population or whether the better match is due to mixing even more of the training samples into the testing data set.
3.3.2 Baseline Algorithm

Generally, the 4 plots of Fig3.5 in this algorithm deviates pretty much from the exact chi-square result. Plot (a) and (b) differ the most.

The deviation is indicated by the fact that the middle section of the P-P curve moves into the upper-left corner of the plot, far away from the reference line. The plot (c) and (d) are little bit better but still show notable deviations between model and sample data. This may due to the characteristic of the Baseline Algorithm. It is notable that the nature of the Baseline Algorithm introduces many more degrees of freedom than are introduced for the Naive and Common Residual algorithms. It is possible that the higher degree-of-freedom parameter for the Baseline Algorithm causes an exaggeration of the experiment-to-theory mismatch shown in Fig3.5. Examining the effect of the chi-square degree-of-freedom parameter on the shape of the probability-probability plot remains an open topic for future research.

3.3.3 Common Residual Algorithm

The comparison of model and experiment for the Common Residual Algorithm has characteristics similar to those observed for the Naive Algorithm (Section 3.3.1). For the common-residual algorithm, it seems that removing the restriction does not improve much of the curves. There is no clear change from plot (a) to (b). This might be caused by the characteristic of this algorithm. The common residual vector contains information of all the in view satellites. For the current data, the 8 satellites used in the restricted case represent the maximal set of satellites in view during the experiment.

If comparing (b) and (c), we may find that (c) is closer to the prediction. The
Figure 3.5: Baseline Algorithm’s result in P-P plots of the four different data.
The only difference between (c) and (b) is that (c) also uses data from the training trial. The blue curve in (d) is almost around the prediction line.

Again, it is difficult to conclude whether the improved matching in subplots (c) and (d) is the result of the increased size of the sample population or the result of mixing the training data with the testing data, thereby possibly biasing the outcome toward a better match.

Figure 3.6: Common Residual Algorithm’s result in P-P plots of the four different data.
3.4 Conclusion

The primary outcome of this section is less an analysis of specific CERIM algorithms than a demonstration of the potential of the probability-probability plot as a tool for analysis of nonstationary random processes. The simulation example as well as the CERIM data-analysis indicate that probability-probability plots can be used to determine how well a model matches a given data set. The probability-probability plot will be explored more in the future (with larger data sets) to learn how to define more formalized methods to verify probabilistic models used in safety-of-life navigation systems.
Chapter 4

Summary

4.1 Summary

This thesis introduced a way of modeling the covariance matrix of Collaboration Enhanced Receiver Integrity Monitoring (CERIM). Integrity monitoring is a very important function for safety-critical GPS applications. CERIM is a new emerging method of integrity monitoring. One of the key parts of realizing accurate CERIM is to weight each element of the residual vector during the computation of monitor statistics.

In this thesis, a method is introduced which solves the problem by modeling the covariance and using the inverse of the covariance matrix to achieve the weighting. The covariance was built on a model of pseudorange noise which is related to the satellite elevation. The parameters of the model were selected by minimizing a cost function which assures the consistency between the model and a training data set.

The new covariance-estimation method was tested with three CERIM algorithms. In Chapter 2, the method was applied to a testing data set in a stationary
condition which fixed the number of satellites used in the data set. The performance of the estimated covariance was tested by analyzing the distribution of the monitor statistic as compared to theory. In Chapter 3, the method was applied to a testing data set in a nonstationary condition where the number of satellites and receivers was variable at each time step. In this chapter, results were compared to theoretical predictions using probability-probability plots. My analysis indicates that the covariance-estimation model was relative accurate, likely good enough for CERIM applications. However, some ambiguity remains in determining the sensitivity of modeling errors to the size of the sample population used in model verification.

4.2 Thesis Contributions

CERIM is a new integrity monitoring method which has many advantages. A key part of realizing accurate CERIM is to weight each element of the parity residual vector. A weighting matrix which can normalize the differences between each element is needed so that a chi-square monitor statistic can be computed. This research created a model which can model the pseudorange noise based on satellite elevation. An indirect method was then introduced to compute a parity residual covariance based on the pseudorange noise model. The inverse of this parity residual covariance can be used as the weighting matrix during the monitor statistic computation.

This model was tuned and tested on the real data collected from a low-cost receiver. This means the tuning and testing methods I used for this model do not require ground truth and online information. The testing results showed that the modeled covariance has a good performance both in the stationary and nonstationary condition.
ary condition.

In concept, the covariance-method introduced in this dissertation could be applied in real-time driving applications, to estimate covariance on the fly. This would very much improve the performance of the CERIM algorithm for urban driving scenarios.

4.3 Future Work

This thesis leaves a number of interesting topic open for future research. They are summarized here.

In the Chapter two, I mentioned that the trial was divided into 3 parts. The middle part contains data collected from the urban canyon region. These data are left unused right now. This data was excluded because it was known that noise levels would be much higher in this high multipath region. New covariance models are needed with additional dependencies (not just elevation) to reflect the differences between performance inside and outside of urban canyons.

Also, the result in Chapter two, Fig2.6, there is an outlier point at 0.8 of the horizontal axis in Naive Algorithm and Common Residual Algorithm. This point may be caused by multipath. It has the risk of triggering the false alarm since its value is really close to the threshold. To avoid the false alarm, it is necessary to analyze the outlier point in the future. The real reason of causing it needs to be studied. In fact, we may need to find out a way to better model the multipath in the future. The multipath effect created many errors in the data which are hard to model. These errors make the current estimated covariance inaccurate to a certain degree.
In the Fig2.6, the plot of Naive Algorithm is almost the same as the plot of the Common Residual Algorithm. This may indicate that the Naive Algorithm is a special case of the Common Residual Algorithm. It will be interesting to investigate this similarity in the future.

In Chapter 3, I talked that the fitness between the data-derived result and the theoretical prediction in Naive and Common-Residual Algorithms improves after adding in the training data set and more receivers’ data. The reason of this improvement is still unclear. Whether it is because the increase size of the available data set or result of mixing with the training data needs to be studied in the future. It will also be interesting to investigate the bad matching in the Baseline Algorithm. The effect of the chi-square degree-of-freedom on the shape of the probability-probability plot can be analyzed in the future.
Bibliography


URL: http://www.gps.gov/applications/survey/.


