Real-Time Bayesian Analysis of Ground Motion Envelopes for Earthquake Early Warning

Thesis by

Gokcan Karakus

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

2016

(Defended January 21, 2016)

Acknowledgements

I would like to thank my advisor, Prof. Thomas H. Heaton, for his extremely helpful advice and limitless patience towards me everyday, without exception, during our collaboration. I feel tremendous honor to be his student.

I would like to thank Prof. James L. Beck for his very important help on the statistical portion of my research. I could not have achieved it without you; thank you.

Starting with Prof. Jean Paul Ampuero, I would like to thank my PhD examining committee. It has been my honor to have him, Prof. Hiroo Kanamori and Prof. Domniki Asimaki on my PhD examining committee.

Starting with Dr. In Ho Cho, I would like to thank the students of Civil Engineering option at Caltech for their company and help during my stay at Caltech.

Starting with Dr. Arthur Lipstein, I would like to thank my friends at Caltech who were associated with options other than Civil Engineering.

Starting with Carolina Oseguera, I would like to thank the staff of the Civil Engineering option at Caltech for creating a positive work environment for me.

Starting with Dr. Yannik Daniel Behr, Dr. Men-Andrin Meier and Dr. Maren Boese, I would like to thank my colleagues at institutions different from Caltech for their help and support.

Starting with the International Student Program (ISP), I would like to thank all of the offices and staff at Caltech for doing their job so well so that I could focus on my research instead of tedious paper work.

Starting with Prof. Cem Yalcin, I would like to thank the civil engineering faculty in my undergraduate institution, Bogazici University, who helped me prepare for a journey of lifelong learning.

I would like to especially thank Dr. Georgia B. Cua for her constant help, and for starting an envelope based earthquake early warning method.

And, finally, I would like to thank my family in Turkey for their emotional support. You deserve the best of everything!

Abstract

Current earthquake early warning systems usually make magnitude and location predictions and send out a warning to the users based on those predictions. We describe an algorithm that assesses the validity of the predictions in real-time. Our algorithm monitors the envelopes of horizontal and vertical acceleration, velocity, and displacement. We compare the observed envelopes with the ones predicted by Cua & Heaton's envelope ground motion prediction equations (Cua 2005). We define a "test function" as the logarithm of the ratio between observed and predicted envelopes at every second in real-time. Once the envelopes deviate beyond an acceptable threshold, we declare a misfit. Kurtosis and skewness of a time evolving test function are used to rapidly identify a misfit. Real-time kurtosis and skewness calculations are also inputs to both probabilistic (Logistic Regression and Bayesian Logistic Regression) and nonprobabilistic (Least Squares and Linear Discriminant Analysis) models that ultimately decide if there is an unacceptable level of misfit. This algorithm is designed to work at a wide range of amplitude scales. When tested with synthetic and actual seismic signals from past events, it works for both small and large events.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recently, the earthquake early warning (EEW) systems are being developed in many parts of the world. These systems do not predict when an earthquake is going to happen; rather they predict the eventual characteristics of earthquakes, such as their magnitude and location, by using the first few seconds of the earthquakes' ground motion data that are being recorded by seismic stations in real-time. Unfortunately, the seismic stations records do not only contain earthquake data, but also any activity that "shakes" the ground, such as sonic booms, quarry blasts, and even heavy traffic noise. These types of activities may confuse a warning system that relies on ground motion amplitudes recorded by seismograms being larger than certain thresholds. In addition to non-earthquake activity, temporal distribution of actual earthquakes, such as an earthquake swarm instead of an isolated earthquake being preceded and followed by quiet ground motion periods, may affect the accuracy of the alert messages sent to the EEW system subscribers.

The current EEW system being tested in California does not have a mechanism that checks the accuracy of the messages sent to the system subscribers. However, we desire to have a sophisticated real-time checking mechanism that will oversee and verify the system predictions. To develop a robust alert confirmation mechanism, we examined more than 500 synthetically created false and missed EEW alert scenarios for earthquakes ranging from M2.5 to M6.5.

We used earthquake waveform envelope data predicted by Cua & Heaton's envelope ground motion prediction equations (Cua 2005) to verify the predictions made by the EEW system. During the comparison calculations between observed and predicted ground motions, we measured the disagreement using higher order statistics: kurtosis and skewness. We developed a new algorithm called reality check (Figure 1) that classifies the state of an EEW system alert based on the kurtosis and skewness measures of the misfit between observed and predicted waveform envelope data.

This thesis presents the step-by-step development process of the reality check algorithm and shows how this technique helps protect the credibility of an EEW system in an innovative manner.

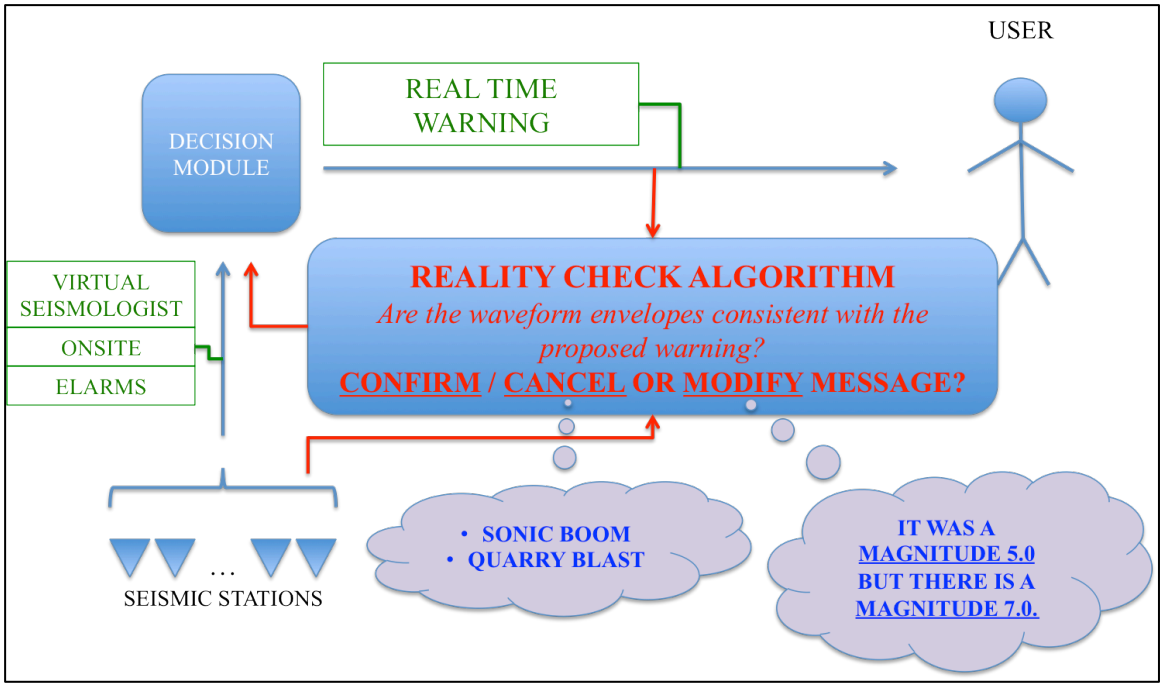

Figure 1.1: Schematic illustration of the reality check algorithm. The recorded ground motion data are sent to a main computer called Decision Module. Decision Module makes a prediction using several algorithms (Virtual Seismologist, Onsite, and Elarms) and sends an alert to the users if necessary. The Reality check algorithm receives the sent alert messages, and then compares them with the real-time recorded data by seismic stations and reports the accuracy of the messages back to the decision module. We continuously check self consistency of data and warning.

Chapter 2

Data Processing

2.1 Data Processing

This section describes the choice of input used regarding the ground motion data.

The seismic stations in California can record ground motion in from 80 samples per second (broadband seismometers) to 100 samples per second (strong motion seismometers). Our algorithm does not use these recordings directly. Instead, we use envelopes of the waveform data. We run a 1 second long window throughout the continuous records in real-time, and take the absolute maximum value within the window and make it our envelope amplitude at the corresponding second in real-time (Figure 2.1). Then, we slide the window to the next second and so on. This process gives us the "envelope of the observed data" whether there is an event or just noise at a site.

2.2 Prelude: Virtual Seismologist

The Virtual Seismologist (VS) (Cua 2005) is briefly explained in this section, because it is a prerequisite for our algorithm.

Our work uses envelopes of earthquake waveform data rather than the seismogram recordings directly as explained in the previous section. In addition to the observed earthquake envelopes, we make use of predicted envelopes that are created by the ground motion prediction equations provided by the Virtual Seismologist. The Virtual Seismologist (VS) (Cua 2005) was developed at Caltech. It is a Bayesian Inference Framework based on waveform envelopes and prior information, e.g., foreshocks, network topology. VS ground motion prediction equations (GMPE's) predict P and S

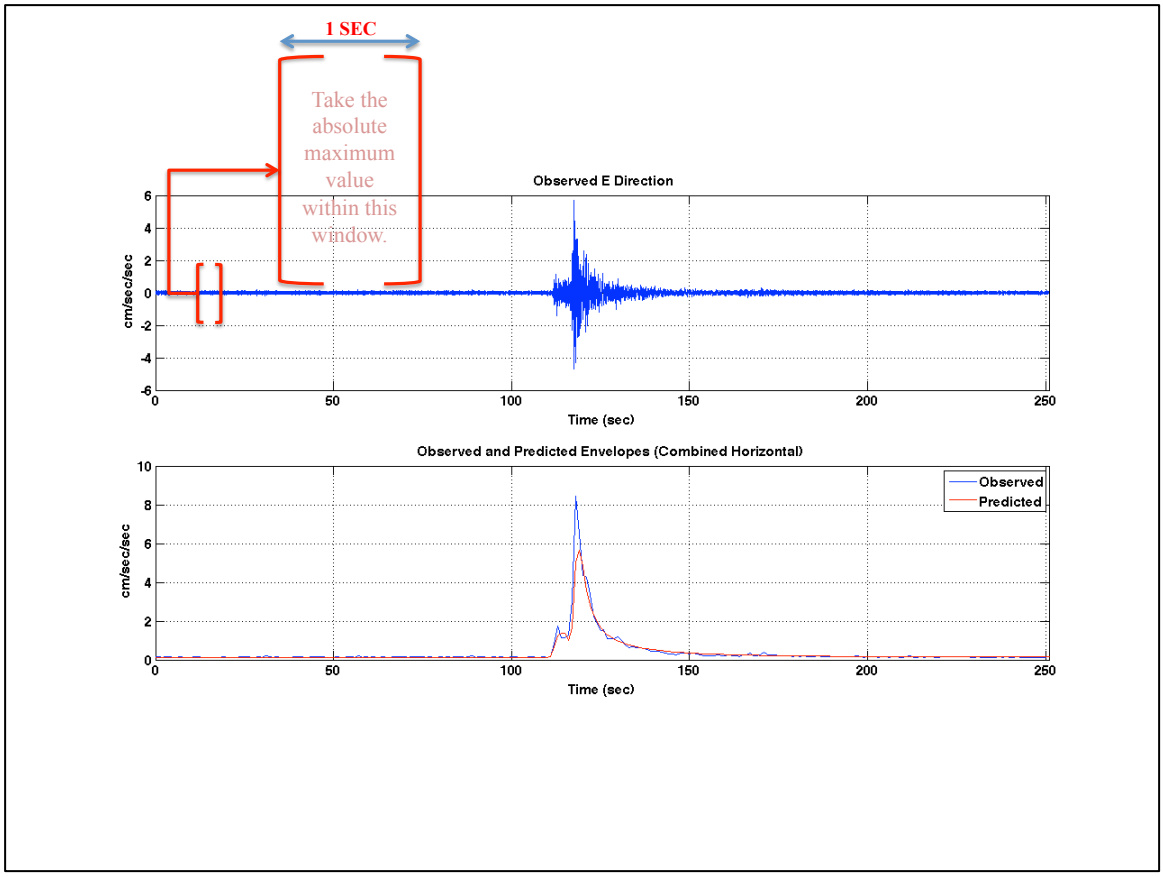

Figure 2.1: Example of an observed and a predicted envelope. Plot on top shows a broadband seismogram recorded at 80 samples per second rate at a station in E-W direction. We run a 1 second long window throughout the continuous records in real-time, and take the absolute maximum value within the window and make it our envelope amplitude at the corresponding second in real-time. Note that the envelope shown in blue in the bottom plot is a combination on E-W and N-S horizontal envelopes. Note that the bottom plot also shows the corresponding predicted envelope created using the Virtual Seismologist method.

waveform envelopes as a function of amplitude and distance. We need to provide the magnitude of the event and the distance between its epicenter and a particular station to create the estimated envelopes at that station. In addition to the P- and S-wave envelopes, VS predicted envelopes also contain a constant noise envelope that represents the noise level at a particular station. These three envelopes (P-wave, S-wave, and constant noise) are combined in a square root of sum of squares fashion to produce the VS predicted envelopes at a given station for a given earthquake. Examples of VS predicted envelopes are shown in Figure 2.2.

$$\text{VS Envelope} = \sqrt{\left(\text{P-wave Envelope}\right)^2 + \left(\text{S-wave Envelope}\right)^2 + \left(\text{Noise Envelope}\right)^2} \quad (2.1)$$

The data for this work are collected from Southern California Seismic Network (SCSN) and Next Generation Attenuation (NGA) strong motion data. Figure 2.2 shows examples of predicted envelopes generated using the Virtual Seismologist method. VS equations for predicting envelopes are given in Appendix A.

2.3 Test Functions

This section explains what we do right after we obtain both the observed and predicted envelopes. Because the seismic stations continuously record ground motions and we assume there will always be some noise being recorded at a station, the predicted envelope (2.1) only consists of constant noise when the decision module is not publishing any earthquake data. That means we have both observed and predicted envelope data at any given time. Therefore, the following steps are performed at every second in real-time.

Once we have both predicted and observed envelope values, we calculate the misfit between them using the following equation:

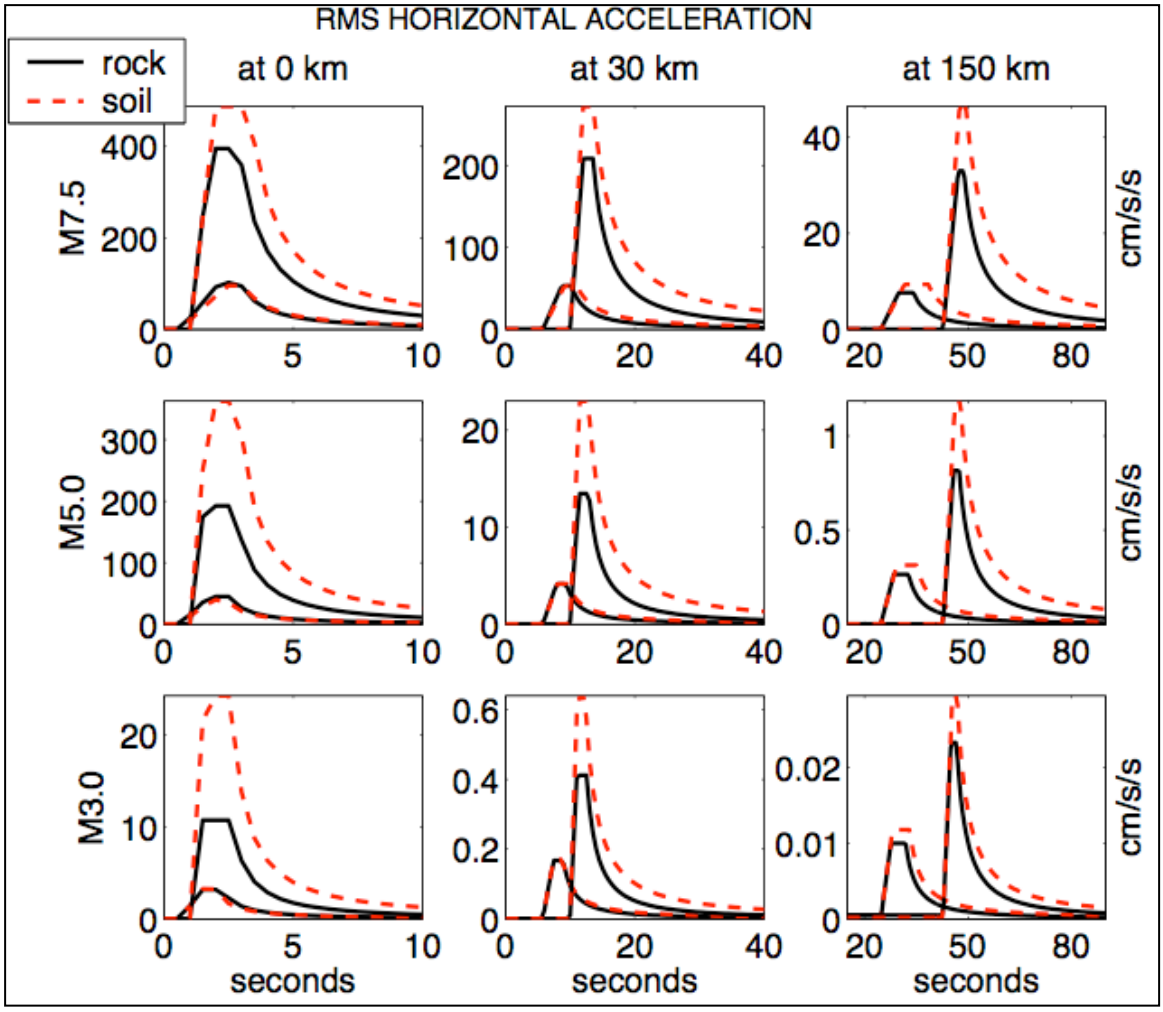

Figure 2.2: Examples of predicted envelopes generated using the Virtual Seismologist method. Note that there is a binary classification as soil/rock at station sites. A different set of ground motion prediction equations are provided for each site class by VS. (Source: (Cua 2005))

$$\phi\left(n\Delta t\right)=B_{HP}*\log\left(\frac{\text{Observed Envelope}\left(n\Delta t\right)}{\text{Predicted Envelope}\left(n\Delta t\right)}\right) \tag{2.2}$$

where

n = 1,2,…

Δt : envelope sampling rate (1 second)

$*$: convolution symbol

B_{HP} : Butterworth high-pass filter, 2nd order, 600 seconds, see below for justification.

ϕ : test function

Test functions, denoted ϕ, are second by second misfit computations between observed and predicted envelopes. Figure 2.3 shows that a test function represents the time evolution of the misfit between observed and predicted ground motion values.

In some of our calculations, we noticed that the misfit in the S-wave coda is usually different from the rest of the misfit, especially for the displacement misfit, if we do not apply high-pass filtering to the ratio of logarithm of envelope values (Figure 2.3). In order not to confuse our algorithm, which indirectly depends on the amplitude of the misfit, we decided to high-pass filter the ratios between observed and predicted ground motion values. We apply a recursive high-pass Butterworth filter to the raw misfit.

2.4 Virtual Seismologist Assumption Regarding Misfit

The data fitting process in creating the predicted envelope GMPE's was done by modeling the difference between predicted and observed ground motion envelopes as a Gaussian distribution with a mean of zero in logarithm space. This model was chosen by the VS creators and we should check how well it does. The histogram of the test function results from the previous example is shown in Figure 2.4.

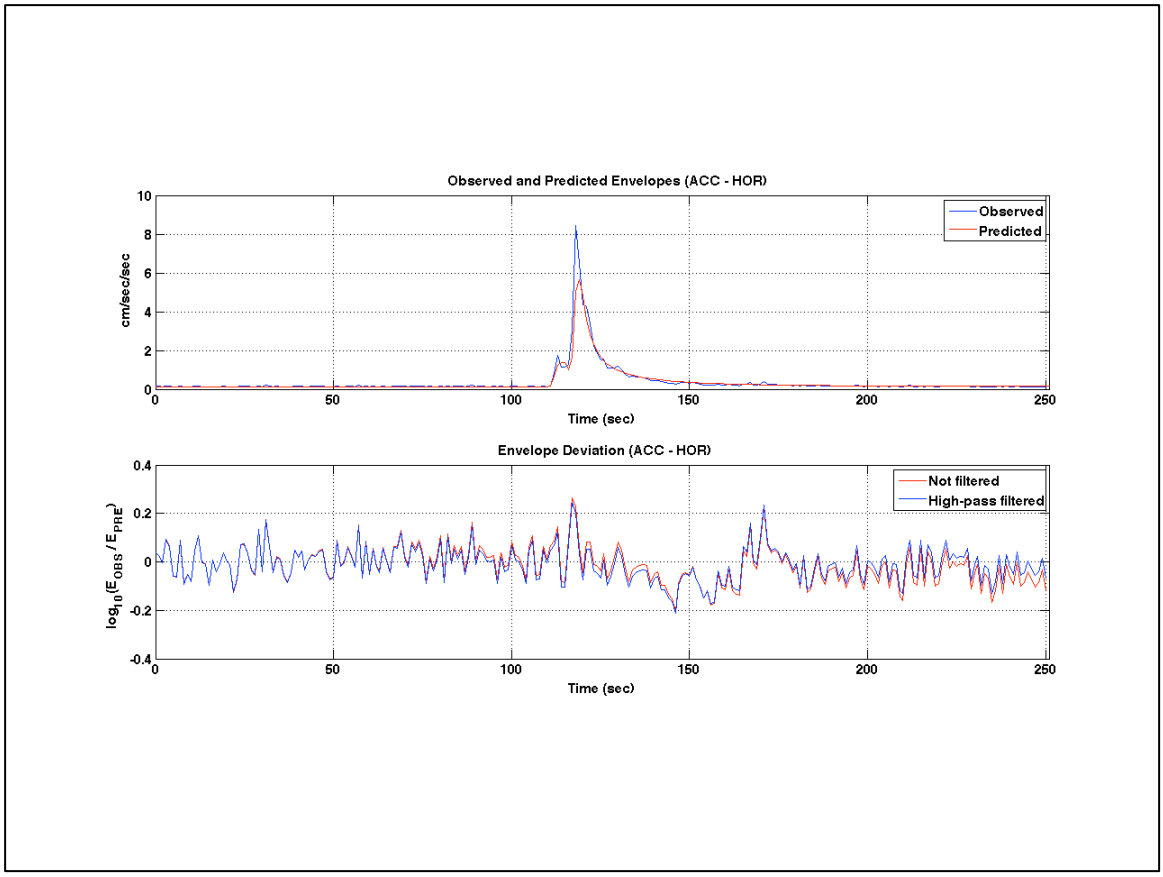

Figure 2.3: Envelopes and their test function. (On top) The observed and predicted envelopes shown in Figure 2.1. (On bottom) Test function computed using (2.2) both with and without high-pass filtering.

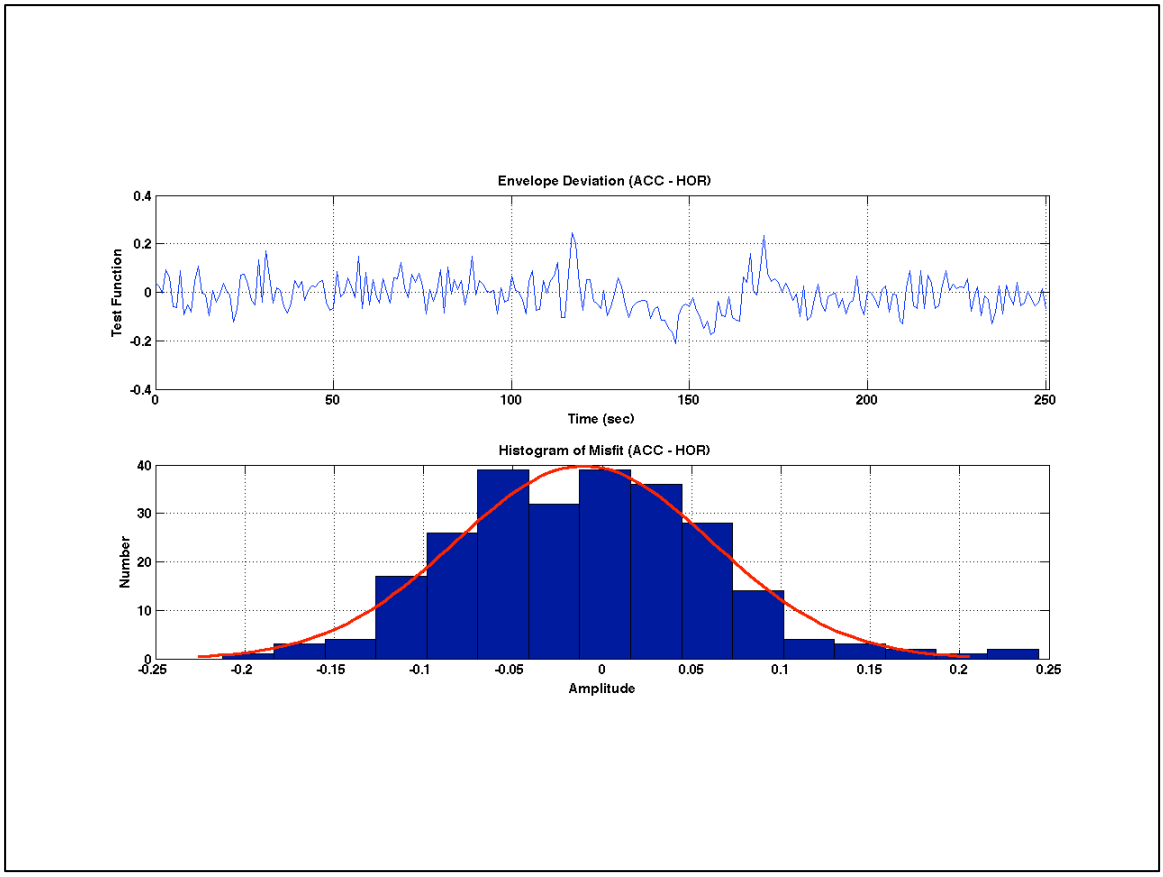

Figure 2.4: A test function result and its histogram.

Each one of the bins shown at the bottom of Figure 2.4 represents an amplitude value in the graph above. All of the values of the test function are distributed about the mean as shown in the histogram, and a Gaussian ($\mu = -0.0104, \sigma = 0.0718$) distribution is adequate to explain it. The VS regression analysis included more than 30,000 seismograms, so this much departure from normality in one example is within acceptable limits (this is not based on an analysis, it is just a qualitative judgment). The assumption is observed to be valid.

It is relatively easy for a human to visually notice if a distribution departs from a Gaussian one, but how can we make a computer detect any unacceptable departure automatically in real-time? One answer is to use higher order statistics: kurtosis and skewness.

2.5 Higher Order Statistics

We use higher order statistics to detect departure from normality. Using higher order statistics makes these techniques more robust than just using the mean and the standard deviation.

2.5.1 Higher Order Statistics – Kurtosis

Kurtosis is a normalized fourth moment of a distribution about its mean where the normalization is done using the square of the variance (Langet, Maggi et al. 2014), so it does not have a unit. It is mainly used to detect outliers based on a normal distribution. Kurtosis for a normal distribution is the value three (3), so outliers in the tails of the distribution make the kurtosis value bigger than three.

$$Kurtosis = \frac{E\left[\left(X-\mu\right)^4\right]}{\left(E\left[\left(X-\mu\right)^2\right]\right)^2} \tag{2.3}$$

where

E : expected value

X : set of values in a distribution

μ : mean of the set of values in X

2.5.2 Higher Order Statistics – Skewness

Skewness, on the other hand, is the third moment about the mean of a distribution, and just like kurtosis, it is normalized using the appropriate power of the variance, so skewness too is dimensionless. Skewness is used to detect departure from symmetry. Its sign depends on the direction of the skew as shown in Figure 2.5. A normal distribution is perfectly symmetric about its mean and so its skewness is zero.

$$Skewness = \frac{E\left[\left(X-\mu\right)^3\right]}{\left(E\left[\left(X-\mu\right)^2\right]\right)^{\frac{3}{2}}} \tag{2.4}$$

Note that higher order statistics equations, i.e., (2.3) and (2.4) are for theoretical moments and they must be replaced by sample moments when applied to data. Matlab functions ‘kurtosis’ and ‘skewness’ are used on the data, however, ‘bias correction’, as explained in the function definition in Matlab, was not applied.

2.6 An Example – Missed Event

We now take a look at an example and see the algorithm in action. The following example aims to simulate an 'under-prediction' case where the system misses an event. In order to emphasize an under-prediction case, the following figure also includes a case where the system successfully detects an earthquake, i.e., the first one (Figure 2.6). The top of Figure 2.6 shows a synthetic seismogram. We basically put two identical events

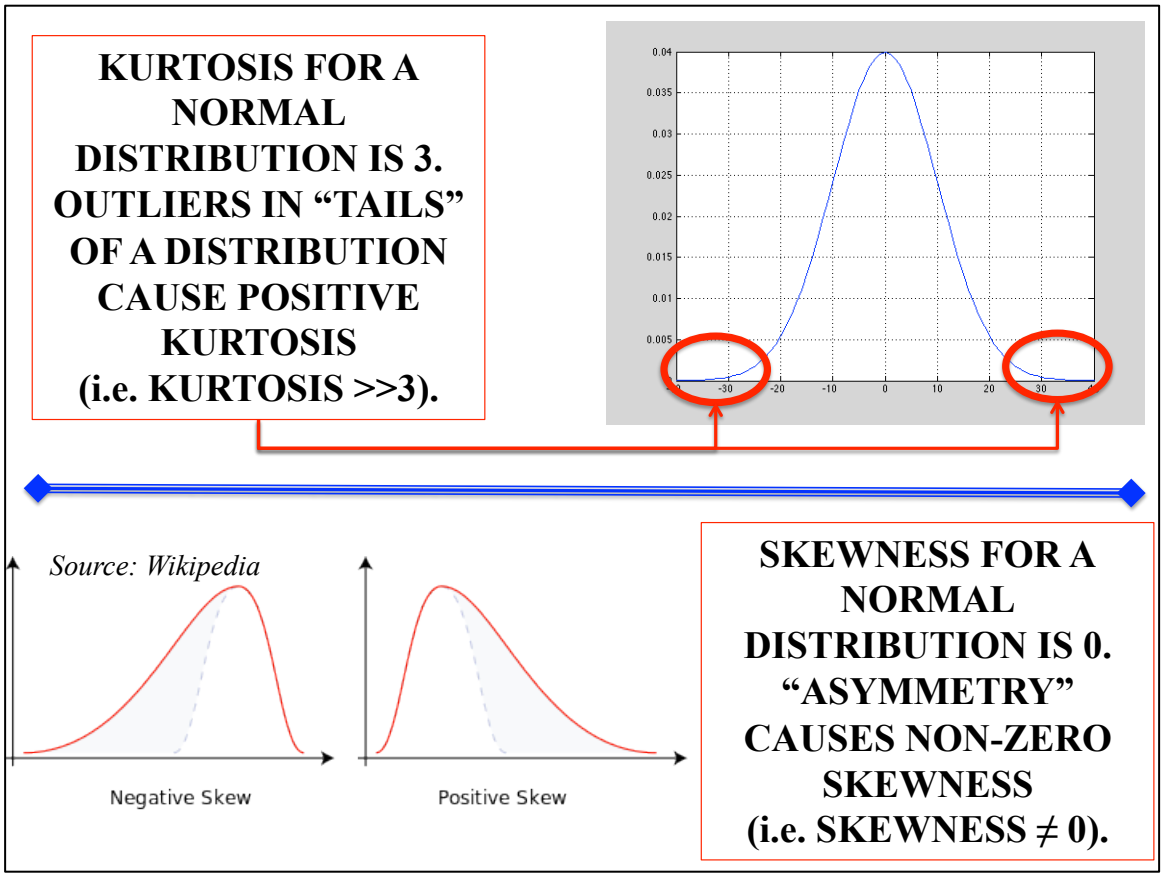

Figure 2.5: Graphical interpretation of higher order statistics.

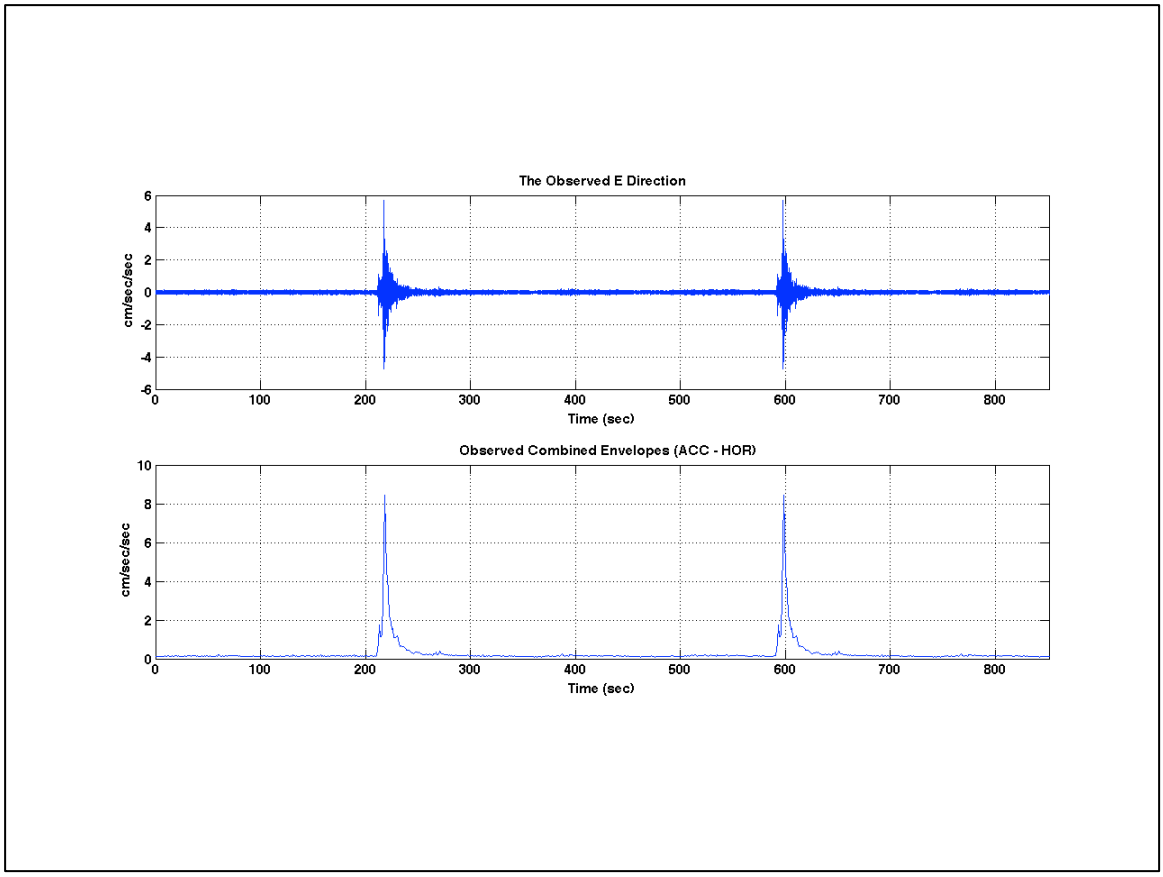

Figure 2.6: Example of a double event seismogram and its envelope. (On top) seismogram, (on bottom) envelope. Envelopes on the bottom are two horizontal envelopes in perpendicular directions combined in a square-root-sum-of-squares fashion.

(~M4.6) back to back in time. On the bottom of Figure 2.6, we see the envelopes created using the technique demonstrated earlier. From now on, we will not show the seismogram itself, and we will work with the envelopes instead.

In this scenario (Figure 2.7), incoming data from the first event stimulated the system to the point where an alarm is issued. The magnitude and the location of the event related to the first alarm are used to create the VS envelopes and they are overlaid on top of the observed ones using the origin time predicted by the decision module. However, there happens to be a second event and the system does not catch it. We calculate our test function in real-time as more data are recorded. Then, time derivatives of kurtosis and skewness, which are computed using a multiple window approach (see Appendix B) on the test function values, are computed in real-time as well.

Everything seems fine until the second event arrives at the station. In other words, both derivative of excess kurtosis and derivative of skewness values indicate normality; therefore, there is agreement between the decision module alert and what is actually observed. But, once the second event arrives, the test function shows values that are considered as outliers and this produces significantly greater values of the derivatives. Moreover, because the symmetry is lost, derivative of skewness values indicate positive skew.

2.7 Another Example – False Event

With an example of a false alarm case, we now show some justification for using two higher order statistics functions instead of one. Kurtosis would give us a very large number if we had a false alarm. That is, if the corresponding test function value was a large negative number, relative to the rest of its values within a given window and

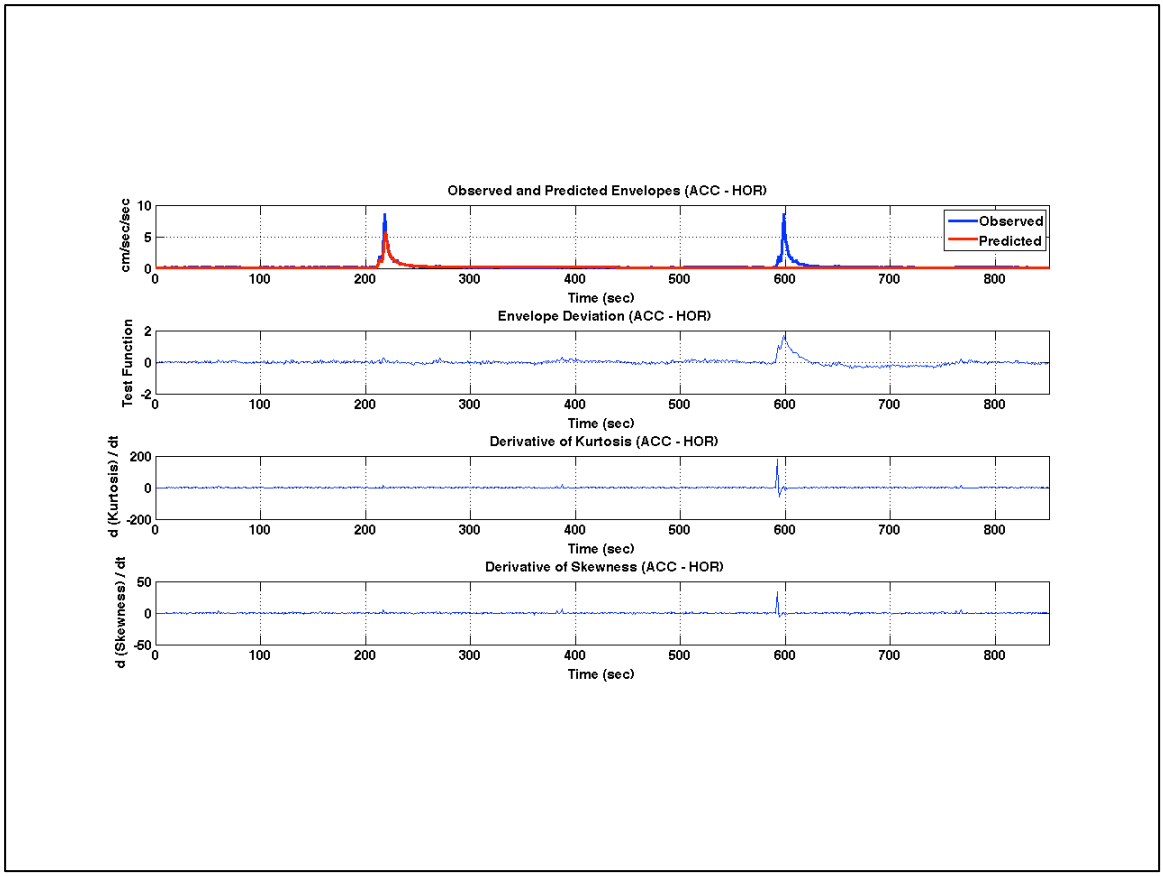

Figure 2.7: Under-prediction, i.e., missed event example.

computed by using equation (2.2), it would be considered as an outlier, similar to the case of missed alarm example. In other words, as far as kurtosis is concerned, there is no difference between the outliers on the left or on the right tail of the mean. However, skewness would detect a negative skew in that scenario (Figure 2.8), and we would know the "nature" of the departure from normality. That is why we use both kurtosis and skewness in our algorithm.

In the scenario shown in Figure 2.8, the decision module predicts an earthquake, and issues the parameters that are needed to create the VS predicted envelopes. All is well. After the earthquake, however, the station that recorded this seismogram experiences some non-earthquake shaking in the form of a short lasting burst of energy, for example, traffic noise nearby. This blip seen in the top row tricks the decision module to issue another alarm, then our algorithm creates the corresponding VS envelope and starts measuring the agreement between the observed and predicted envelopes. As soon as the unacceptable mismatch is detected with the help of derivative of higher order statistics, our algorithm indicates an over-prediction.

2.8 Summary of Test Function States

To sum up, we declare that there are three states that a test function can be in (Figure 2.9):

State 1: The ideal case where the decision module reports agreement with the observed data, i.e., okay-prediction.

State 2: There is a second event when we thought there was only one, i.e., under-prediction, or missed event/alarm.

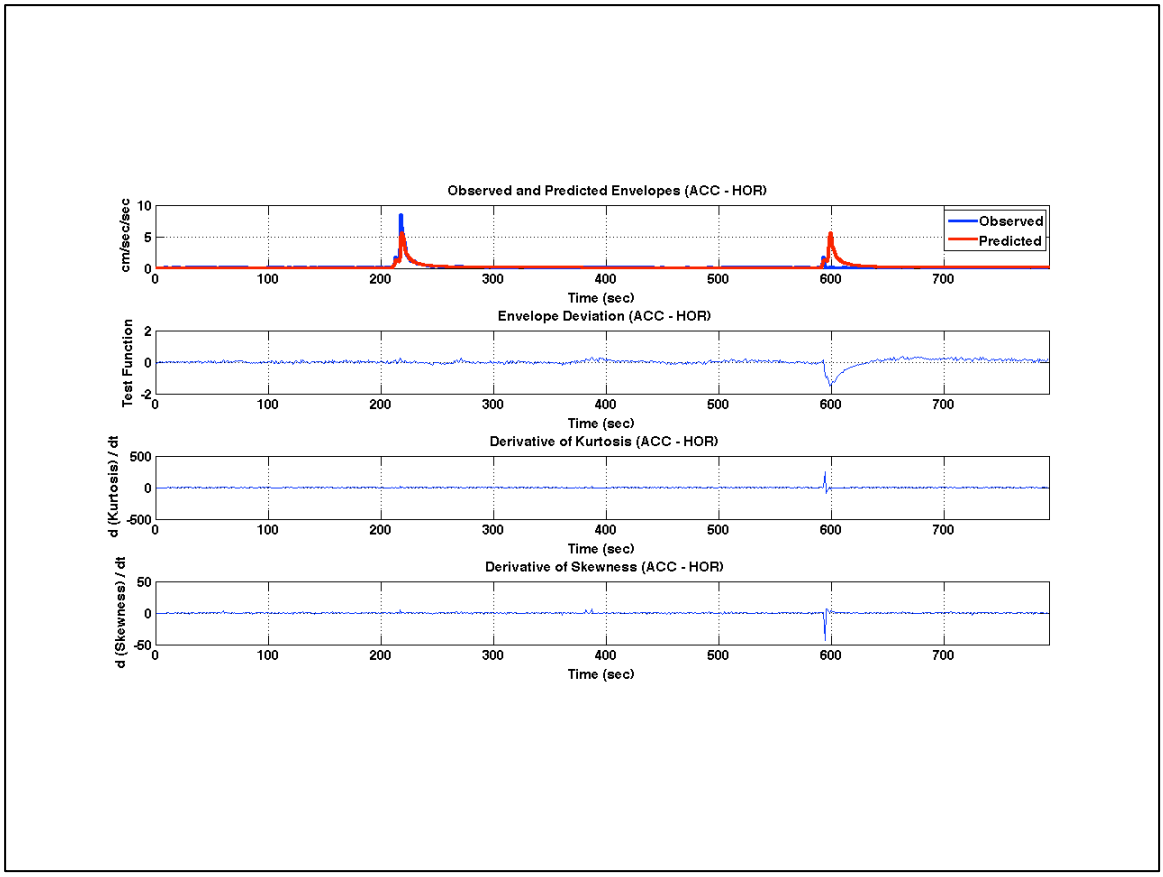

Figure 2.8: Over-prediction, i.e., false alarm example.

State 3: A non-earthquake vibration such as noise at a station tricks the system into believing that there is an earthquake, i.e., false event/alarm. This one is especially tricky because one might lose confidence in the warning system.

We perform separate analyses for different ground motion parameters influenced by different frequency contents; acceleration, velocity, and displacement are separately used as inputs to our computations. We obtained our training data by synthetically creating over- and under- prediction scenarios. The VS predicted envelopes were created using the cataloged magnitude and location. Our training data included 1000 okay-prediction, 250 over-prediction, and 250 under-prediction input values in both horizontal and vertical acceleration, velocity, and displacement.

The peak values of derivatives of kurtosis and skewness for both horizontal and vertical acceleration, velocity, and displacement, at the moment of unacceptable mismatch between observed and predicted envelopes as shown in the figures above, are considered as inputs that belong to under- and over- prediction classes: under-prediction if observed envelope value is significantly larger, and over-prediction if predicted envelope value is significantly larger. Maximum and minimum values of the derivatives preceding the peak values mentioned above are considered as inputs of the okay-predictions because they represent the boundaries within which the state of the prediction is acceptable. We use these values as inputs for various classification models explained in the following sections. Note that the coefficient values presented in this thesis are truncated versions although they are used, in computations, as the computer programs produced, i.e. with higher precision. We used our judgment regarding meaningful precision presentation.

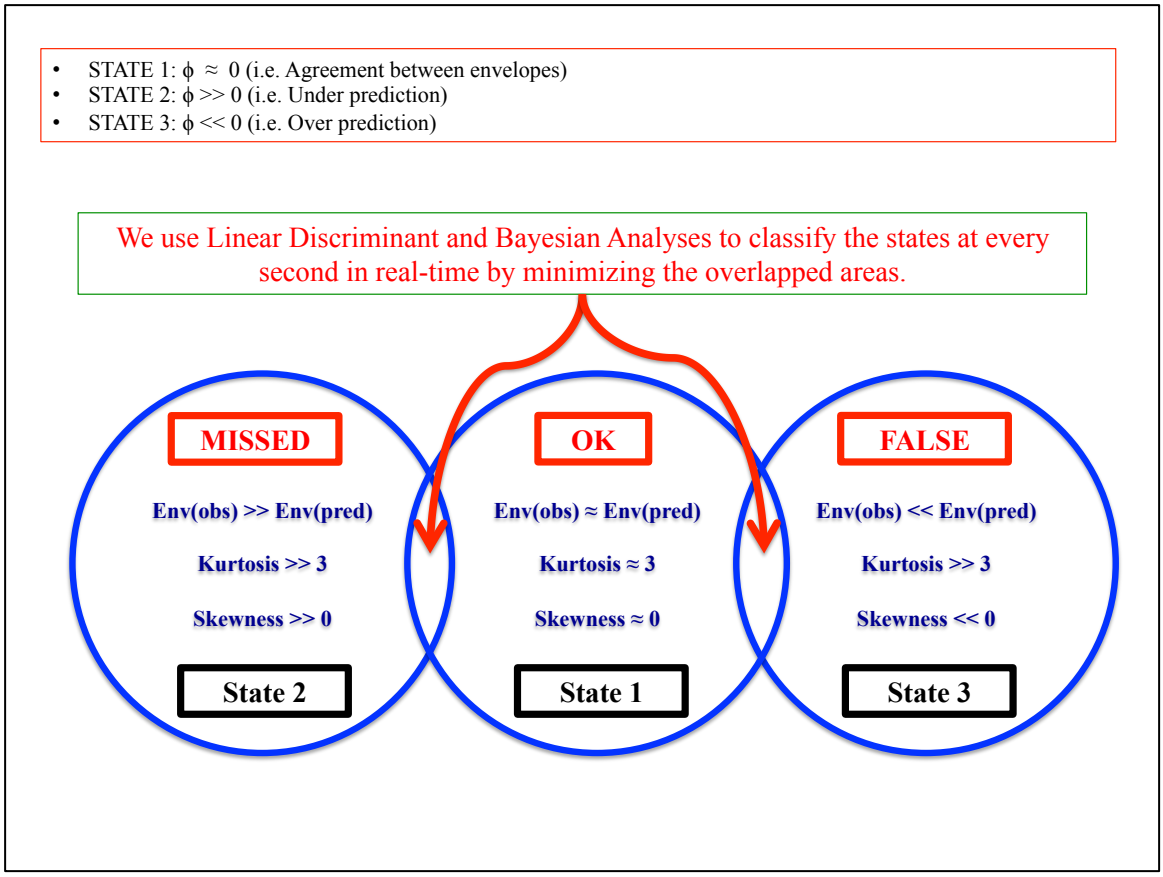

Figure 2.9: Three states of the test function.

Chapter 3

Probabilistic Classifications

Mathematical Notation:

Throughout this work, unless stated otherwise, we use lowercase bold Roman letters, such as $\mathbf{x}$, to denote vectors. All vectors are assumed to be column vectors. Therefore $\mathbf{x}=\left[x_1,x_2,\cdots,x_D\right]^T$ is a D-dimensional vector. We use uppercase bold Roman letters, such as $\mathbf{W}$, to denote matrices.

Classifications:

In the following sections, we present different ways of applying classification. Two probabilistic classification methods are described in this chapter (towards the end of it): Method I which uses Ockham's razor on a given set of models, and Method II which uses Sparse Bayesian Learning (SBL), i.e., models with Automatic Relevance Determination (ARD) prior. Note that both of these methods are using Bayesian Ockham razor by maximizing evidence (or posterior probability); with ARD prior, it is a continuum of model classes, each defined by specifying hyperparameters (prior variances), instead of a discrete set of model classes *(personal communication with Prof. James L Beck)*. Two non-probabilistic classification methods were also examined: least squares and linear discriminant analysis. The theory and results for these non-probabilistic methods are presented in Appendix C. Appendix C is included for two main purposes: 1) to provide classification methods that do not require as much run time as the

probabilistic methods, 2) to help some interested readers understand probabilistic classification models if classification is a new concept to them.

We aim to have the least amount of misclassification rate when we take an input vector $\mathbf{x}$ and assign it to a class among K discrete classes. Unless it is specified otherwise, we will use linear models where the decision boundaries that separate classes are linear functions of the input. The number of classes in our work is three, i.e., $K = 3$ (Figure 2.9). We will use a $1-of-K$ coding scheme (as described in Bishop, 2006) in which the target vector $\mathbf{t}$ is of length $K = 3$ such that if the class for an input is C_j, then all elements t_k of $\mathbf{t}$ are zero except element t_j, which takes the value 1 (Bishop 2006). Our classifications will generally be done using a discriminant function. The parameters of the discriminant functions are obtained with several different techniques in the following sections.

3.1 Classifications – Probabilistic Generative Model

The following theory is based on the concepts described in Chapter 4, Linear Models for Classification in Bishop, 2006.

In this chapter, we work with Bayesian probabilistic classifications so when a new input is observed, we compute probability (the degree of plausibility) that it belongs to any class.

We start with generative models. Generative models use a probability model for the inputs and outputs; that is, we can create synthetic input data by sampling from the generative probabilistic model. In this section, we aim to compute posterior probabilities,

$p\left(C_k|\mathbf{x}\right)$, with the help of Bayes' theorem. In order to do that, we model both class-conditional densities $p\left(\mathbf{x}|C_k\right)$ and class priors $p\left(C_k\right)$.

Our classification problem has three classes. In practice, the problems with more than two classes, i.e., $K>2$ are considered "multiclass" classification problems. It is common to use a softmax function in a multiclass classification problem. So, to better understand what a softmax function represents, let us start with a two-class classification problem set up ($K=2$) and then try to generalize our computations to a multiclass classification problem. If we had only two classes, i.e., C_1 and C_2 we could write the posterior probability for class C_1 as

$$p\left(C_1|\mathbf{x}\right)=\frac{p\left(\mathbf{x}|C_1\right)p\left(C_1\right)}{p\left(\mathbf{x}|C_1\right)p\left(C_1\right)+p\left(\mathbf{x}|C_2\right)p\left(C_2\right)} \tag{3.1}$$

We can write (3.1) as

$$p\left(C_1|\mathbf{x}\right)=\sigma\left(a\right)=\frac{1}{1+\exp\left(-a\right)} \tag{3.2}$$

where a is implicitly defined as

$$a=\ln\frac{p\left(\mathbf{x}|C_1\right)p\left(C_1\right)}{p\left(\mathbf{x}|C_2\right)p\left(C_2\right)} \tag{3.3}$$

and $\sigma\left(a\right)$ defined by (3.2) is called the logistic sigmoid function. For a multiclass case (where $K>2$), we consider a to be a linear function of the input $\mathbf{x}$. From Bayes Theorem:

$$p\left(C_k\middle|\mathbf{x}\right)=\frac{p\left(\mathbf{x}\middle|C_k\right)p\left(C_k\right)}{\sum_j p\left(\mathbf{x}\middle|C_j\right)p\left(C_j\right)}=\frac{\exp\left(a_k\right)}{\sum_j \exp\left(a_j\right)} \tag{3.4}$$

Expression (3.4) is known as the softmax function (or the normalized exponential). It is regarded as a generalization of the logistic sigmoid to a multiclass problem. In this case, the quantities a_k are defined as

$$a_k=\ln\left(p\left(\mathbf{x}\middle|C_k\right)p\left(C_k\right)\right) \tag{3.5}$$

Let us now see what happens if we choose a Gaussian form for the class-conditional densities $p\left(\mathbf{x}\middle|C_k\right)$. For simplicity, let us assume that all classes have the same covariance matrix Σ but each class has a distinct mean $\mathbf{m}_k$. Then, we can write the class-conditional density for class C_k as

$$p\left(\mathbf{x}\middle|C_k\right)=\frac{1}{\left(2\pi\right)^{D/2}}\frac{1}{\left|\Sigma\right|^{1/2}}\exp\left\{-\frac{1}{2}\left(\mathbf{x}^T-\mathbf{m}_k\right)^T\Sigma^{-1}\left(\mathbf{x}^T-\mathbf{m}_k\right)\right\} \tag{3.6}$$

where D is the dimensionality of $\mathbf{x}$. Using (3.4) and (3.5), we can write

$$a_k\left(\mathbf{x}\right)=\mathbf{w}^T\mathbf{x}+w_{k0} \tag{3.7}$$

where we used the following definitions

$$\begin{aligned}\mathbf{w}_k&=\Sigma^{-1}\mathbf{m}_k\\ w_{k0}&=-\frac{1}{2}\mathbf{m}_k{}^T\Sigma^{-1}\mathbf{m}_k+\ln p\left(C_k\right)\end{aligned} \tag{3.8}$$

Thanks to our assumption that all classes have the same covariance matrix, the quadratic terms cancel and the $a_k\left(\mathbf{x}\right)$ become linear functions of $\mathbf{x}$. This results in decision

boundaries being defined as linear functions of $\mathbf{x}$, as opposed to the case where we do not make the same covariance assumption. In that case (where each class has its own covariance matrix), the quadratic terms would not cancel and we would no longer have a linear discriminant.

In the next step, we will use maximum likelihood (maximum a posteriori for uniform priors) to calculate the parameters, $\mathbf{w}_k$ and w_{k0}, and the prior class probabilities, $p(C_k)$. We have chosen a Gaussian form for class-conditional densities, $p(\mathbf{x}|C_k)$, as we mentioned above. We will use the data set described in Chapter 2; the training data set is given as $\{\mathbf{x}_n, \mathbf{t}_n\}$ with $n = 1, \cdots, N$. We construct $\mathbf{X}$ with its n^{th} row $\mathbf{x}_n^T$ given by an instance of

$$\mathbf{x} = \begin{bmatrix} \frac{d}{dt}(\text{Kurtosis of Horizontal Acceleration}) \\ \frac{d}{dt}(\text{Kurtosis of Vertical Acceleration}) \\ \frac{d}{dt}(\text{Skewness of Horizontal Acceleration}) \\ \frac{d}{dt}(\text{Skewness of Vertical Acceleration}) \end{bmatrix} \tag{3.9}$$

The target vector $\mathbf{t}_n$ is a binary vector of length $K = 3$. It uses the $1-of-K$ coding scheme; it has components $t_{nj} = \delta_{jk}$ (Kronecker delta) if input n is from class C_k. We denote the prior class probabilities as

$$p(C_k) = \pi_k \tag{3.10}$$

We chose a probability model where the predictions of the features for class C_k are independent, so the likelihood function is given by

$$p\left(\mathbf{T},\mathbf{X}\middle|\pi_k,\mathbf{m}_k,\Sigma\right)=\prod_{n=1}^{N}\prod_{k=1}^{K}\left\{p\left(\mathbf{x}_n\middle|C_k\right)\pi_k\right\}^{t_{nk}} \tag{3.11}$$

After taking the logarithm of (3.11), we get

$$\ln p\left(\mathbf{T},\mathbf{X}\middle|\pi_k,\mathbf{m}_k,\Sigma\right)=\sum_{n=1}^{N}\sum_{k=1}^{K}t_{nk}\left\{\ln p\left(\mathbf{x}_n\middle|C_k\right)+\ln\pi_k\right\} \tag{3.12}$$

To determine the prior probability for C_k, we take the derivative of (3.12) with respect to π_k and equate it to zero, and note that $\sum_k \pi_k = 1$. Then we obtain

$$\pi_k=\frac{N_k}{N} \tag{3.13}$$

where N_k represents the number of data points assigned to class C_k.

Let us next determine $\mathbf{m}_k$. We will take the derivative of (3.12) with respect to $\mathbf{m}_k$ and set the result to zero. Using (3.6) as the functional form of class-conditional densities, we obtain

$$\mathbf{m}_k=\frac{1}{N_k}\sum_{n=1}^{N}t_{nk}\mathbf{x}_n \tag{3.14}$$

which is simply the mean of all of the input vectors assigned to class C_k.

The only parameter left to determine is the shared covariance, Σ. Once again, taking the derivative with respect to Σ and setting it to zero gives

$$\Sigma=\sum_{k=1}^{K}\frac{N_k}{N}\mathbf{S}_k=\sum_{k=1}^{K}\pi_k\mathbf{S}_k \tag{3.15}$$

where

$$\mathbf{S}_k = \frac{1}{N_k}\sum_{n=1}^{N} t_{nk}\left(\mathbf{x}_n - \mathbf{m}_k\right)\left(\mathbf{x}_n - \mathbf{m}_k\right)^T \tag{3.16}$$

Expressions (3.15) and (3.16) show that we can compute Σ by weighting the covariances of the data of each class by their prior probabilities and then averaging the result.

For the following confusion matrices (where the rows are the total percentage of a labeled class, and hence the values in a row add up to 100, and the columns are the fraction of the total number of cases for that row that is classified as the indicated class on the top row of that column), we choose the maximum probability value for a given input. Although the parameter values, $\mathbf{w}_k$ and w_{k0}, are computed using the entire data set for training, we provide confusion matrices for cross validation results for different ground motion quantities. The confusion matrix is useful to assess the algorithm's performance. We divided our data set into two halves: training and validation sets. We first trained our algorithm using one set (training set) and then calculated the result using the other half (validation set). Then, we swapped the sets, that is, we used the validation set from the previous step as the training set and the training set as the validation set. Then, we computed the average performance of these two validations. As a final step, we used the entire data set as both training and validation data sets.

Note that $k=1$ represents okay-prediction, $k=2$ represents over-prediction, and $k=3$ represents under-prediction classes in this thesis.

The parameter values and the shared covariance matrix computed using the entire acceleration data set for training are

$$\mathbf{w}_{1_{acceleration}} = \begin{bmatrix} 0.0001 \\ 0.003 \\ 0.022 \\ -0.020 \end{bmatrix} \qquad (3.17)$$

$$w_{1_{acceleration_0}} = -0.450$$

$$\mathbf{w}_{2_{acceleration}} = \begin{bmatrix} 0.016 \\ 0.019 \\ -0.113 \\ -0.120 \end{bmatrix} \qquad (3.18)$$

$$w_{2_{acceleration_0}} = -8.812$$

$$\mathbf{w}_{3_{acceleration}} = \begin{bmatrix} 0.011 \\ 0.019 \\ 0.118 \\ 0.038 \end{bmatrix} \qquad (3.19)$$

$$w_{3_{acceleration_0}} = -7.936$$

$$\Sigma_{acceleration} = \begin{bmatrix} 6926.366 & 5284.517 & 258.698 & 241.974 \\ 5284.517 & 7306.191 & 134.662 & 316.073 \\ 258.698 & 134.662 & 201.493 & 164.134 \\ 241.974 & 316.073 & 164.134 & 217.947 \end{bmatrix} \qquad (3.20)$$

Table 3.1: Confusion matrix for probabilistic generative classification using acceleration data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	97.4	0.5	2.1
	Over	33.2	66.8	0
	Under	24.4	0	75.6
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	93.2	0.6	6.2
	Over	13.6	86.4	0
	Under	12.8	0	87.2
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	98.4	0.2	1.4
	Over	59.2	40.8	0
	Under	42.4	0	57.6
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	95.8	0.4	3.8
	Over	36.4	63.6	0
	Under	27.6	0	72.4

The parameter values and the shared covariance matrix computed using the entire velocity data set for training are

$$\mathbf{w}_{1_{velocity}} = \begin{bmatrix} 0.001 \\ 0.002 \\ 0.015 \\ -0.011 \end{bmatrix} \tag{3.21}$$

$$w_{1_{velocity0}} = -0.437$$

$$\mathbf{w}_{2_{velocity}} = \begin{bmatrix} 0.031 \\ 0.007 \\ -0.222 \\ -0.030 \end{bmatrix} \tag{3.22}$$

$$w_{2_{velocity_0}} = -7.235$$

$$\mathbf{w}_{3_{velocity}} = \begin{bmatrix} 0.015 \\ 0.015 \\ 0.122 \\ 0.048 \end{bmatrix} \tag{3.23}$$

$$w_{3_{velocity_0}} = -8.034$$

$$\Sigma_{velocity} = \begin{bmatrix} 5682.072 & 4229.134 & 384.569 & 313.847 \\ 4229.134 & 5958.392 & 254.687 & 310.505 \\ 384.569 & 254.687 & 165.553 & 132.205 \\ 313.847 & 310.505 & 132.205 & 186.711 \end{bmatrix} \tag{3.24}$$

Table 3.2: Confusion matrix for probabilistic generative classification using velocity data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	96.8	0.7	2.5
	Over	37.6	62.4	0
	Under	26.8	0	73.2
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	91	1	8
	Over	18.4	81.6	0
	Under	10.4	0	89.6
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	97	0.6	2.4
	Over	71.2	28.8	0
	Under	48	0	52
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	94	0.8	5.2
	Over	44.8	55.2	0
	Under	29.2	0	70.8

The parameter values and the shared covariance matrix computed using the entire displacement data set for training are

$$\mathbf{w}_{1_{displacement}} = \begin{bmatrix} 0.0005 \\ 0.005 \\ 0.008 \\ -0.023 \end{bmatrix} \qquad (3.25)$$

$$w_{1_{displacement_0}} = -0.431$$

$$\mathbf{w}_{2_{displacement}} = \begin{bmatrix} 0.052 \\ 0.009 \\ -0.324 \\ -0.045 \end{bmatrix} \qquad (3.26)$$

$$w_{2_{displacement_0}} = -5.490$$

$$\mathbf{w}_{3_{displacement}} = \begin{bmatrix} 0.019 \\ 0.007 \\ 0.169 \\ 0.066 \end{bmatrix} \qquad (3.27)$$

$$w_{3_{displacement_0}} = -7.186$$

$$\Sigma_{displacement} = \begin{bmatrix} 3188.044 & 2203.698 & 386.171 & 311.242 \\ 2203.698 & 2678.795 & 281.599 & 332.173 \\ 386.171 & 281.599 & 100.443 & 72.613 \\ 311.242 & 332.173 & 72.613 & 96.763 \end{bmatrix} \qquad (3.28)$$

Table 3.3: Confusion matrix for probabilistic generative classification using displacement data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	97.4	1.4	1.2
	Over	44	56	0
	Under	38.4	0	61.6
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	95	1.4	3.6
	Over	32	68	0
	Under	17.6	0	82.4
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	97.8	1.4	0.8
	Over	67.2	32.8	0
	Under	63.2	0	36.8
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	96.4	1.4	2.2
	Over	49.6	50.4	0
	Under	40.4	0	59.6

3.2 Discussion of Results for Tables 3.1 to 3.3

The confusion matrices show a consistent decrease in prediction performances with decreasing frequencies (acceleration is dominated by high-frequencies, velocity by mid-frequencies, and displacement by low-frequencies) although this decrease is not significant. Cross validation using the first half of the data set for training and the rest for validation produces reasonable prediction performances (at least more than 65 percent accurate predictions). On the other hand, using the second half for training and the first half for validation gives not so desirable cross validation results. This trend is seen in all types of input: acceleration, velocity, and displacement. The simplest explanation is that the models could be suffering from overfitting because we did not use a prior distribution to provide regularization. We will propose a solution that is robust to overfitting by using a full Bayesian treatment in the next sections.

3.3 Classification – Probabilistic Discriminative Model with Maximum Likelihood Estimate (MLE)

The following theory is based on the concepts described in Chapter 4, Linear Models for Classification in Bishop, 2006.

In the previous section, we determined the parameters of the class-conditional densities, $p\left(\mathbf{x}\middle|C_k\right)$, along with the class priors, $p\left(C_k\right)$. Then, we used Bayes' theorem to compute the posterior class probabilities, $p\left(C_k\middle|\mathbf{x}\right)$. The posterior class probabilities are given by a softmax function of a linear function of the input feature vector $\mathbf{x}$. The coefficients (parameters $\mathbf{w}_k$ and w_{k0}) of the linear function of $\mathbf{x}$ are computed using the parameters of the class-conditional densities, i.e., $\mathbf{m}_k$ and Σ and the class priors, $p\left(C_k\right)$.

The advantage of the probabilistic generative model is that we can create (generate) synthetic input values, $\mathbf{x}$, by sampling from the marginal distribution $p(\mathbf{x})$. However, the predictive performance may decrease, especially when the Gaussian form, which we used to model the class-conditional densities, does not give a good representation. In this section, we compute the parameter values in a more direct approach by maximization of the likelihood function or the posterior probability density function (PDF). By not modeling the class-conditional densities explicitly, we will have less number of parameters to determine, and this may lead to an increase in predictive performance. Directly determining the parameters is an example of a *probabilistic discriminative approach*.

The likelihood function we want to maximize to determine the parameters consists of the conditional distributions introduced earlier: $p(C_k|\mathbf{x})$. We start with a relabeling of the variables first, and then we simplify as much as possible to avoid clutter in our mathematical expressions. In the previous section, we obtained the functional form of the posterior class probability, conditional on an input vector, as (3.4). Using this definition, let us define

$$y_k(\tilde{\mathbf{x}}) = p(C_k|\tilde{\mathbf{x}}, \tilde{\mathbf{w}}_k) = \frac{\exp\left(a_k\left(\tilde{\mathbf{x}}|\tilde{\mathbf{w}}_k\right)\right)}{\sum_j \exp\left(a_j\left(\tilde{\mathbf{x}}|\tilde{\mathbf{w}}_j\right)\right)} \tag{3.29}$$

where a_k are called *activations* and are given by

$$a_k\left(\tilde{\mathbf{x}}|\tilde{\mathbf{w}}_k\right) = \tilde{\mathbf{w}}_k{}^T\tilde{\mathbf{x}} \tag{3.30}$$

Let us now clarify the terms that involve "~":

In the expression above, $\tilde{\mathbf{w}}_k = \left(w_{k0}, \mathbf{w}_k{}^T\right)^T$ and $\tilde{\mathbf{x}} = \left(1, \mathbf{x}^T\right)^T$, that is, we augment the input vector with a dummy input $x_0 = 1$, similar to what we did in the least squares classification in Appendix C. In order to decrease the clutter in the mathematical notation, let us redefine the parameters ($\tilde{\mathbf{w}} \to \mathbf{w}$ and $\tilde{\mathbf{x}} \to \mathbf{x}$) such that $\mathbf{w}_k = \left(w_{k0}, w_{k1}, \cdots, w_{kD}\right)^T$ and $\mathbf{x} = \left(1, x_1, x_2, \cdots, x_D\right)^T$. Then, we consider maximization of the likelihood function to determine the parameters $\left\{\mathbf{w}_k\right\}$ directly.

Now, we need the likelihood function. As we mentioned above, it consists of the posterior class probabilities $p\left(C_k \middle| \mathbf{x}\right)$ if the prior on the C_k's is uniform. We will follow the same $1-of-K$ coding scheme as we did above for the target vectors: the target vector $\mathbf{t}_n$ associated with the input vector $\mathbf{x}_n$, which is assigned to class C_k, will be a unit vector of dimension $K = 3$ with each of its elements being zero unless it is the k^{th} element, which is one. Then, we obtain the likelihood function as

$$p\left(\mathbf{T} \middle| \mathbf{X}, \mathbf{W}\right) = \prod_{n=1}^{N} \prod_{k=1}^{K} p\left(C_k \middle| \mathbf{x}_n, \mathbf{w}_k\right)^{t_{nk}} = \prod_{n=1}^{N} \prod_{k=1}^{K} y_{nk}{}^{t_{nk}} \tag{3.31}$$

where the elements t_{nk} form the matrix $\mathbf{T}$ whose dimension is $N \times K$ with N as the number of data points and K as the number of classes, and $\mathbf{W}$ is formed by $D+1$-dimensional vector $\mathbf{w}_k$ as its k^{th} column, and $\mathbf{X}$ is formed by $D+1$-dimensional vector $\mathbf{x}_n{}^T$ as its n^{th} row. So, $\mathbf{W}$ and $\mathbf{X}$ are matrices with dimensions $\left(D+1\right) \times K$ and $N \times \left(D+1\right)$ respectively. We also have

$$y_{nk} = y_k(\mathbf{x}_n) = p(C_k|\mathbf{x}_n, \mathbf{w}_k) = \frac{\exp(\mathbf{w}_k{}^T \mathbf{x}_n)}{\sum_j \exp(\mathbf{w}_j{}^T \mathbf{x}_n)} \in [0,1] \quad (3.32)$$

Before we start evaluating the probabilistic discriminative model from a Bayesian perspective, let us use the maximum likelihood method to find $\mathbf{W}_{MLE}$ by maximizing the likelihood function given by (3.31). Note that the value we will find is in fact the Bayesian *maximum a posteriori* (MAP) value but we are taking a flat (non-informative) prior for $\mathbf{W}$, so $\text{MAP} \equiv \text{MLE}$. I solved this optimization problem by an algorithm provided by Matlab.

We maximized (3.31) with respect to $\mathbf{W}$ separately for acceleration, velocity, and displacement input and obtained the following confusion matrices by assigning an input vector $\mathbf{x}$ to class C_k, where $p(C_k|\mathbf{x}, \mathbf{w}_k)$ is a maximum over $k = 1,2,3$. Similar to the previous confusion matrices, we show the predictive performance of our models by cross validation; we divide the data sets into two: training data set and validation data set. Then we swap these data sets and average the predictive performances in the form of confusion matrices.

Let us start with acceleration results. The maximum likelihood estimate (MLE) of the parameter matrix, $\mathbf{W}_{MLE_{acceleration}}$, computed using the entire acceleration data set for training, is given by

$$\mathbf{W}_{MLE_{acceleration}} = \begin{bmatrix} 2.899 & -1.263 & -1.606 \\ -0.003 & 0.018 & 0.003 \\ 0.002 & 0.013 & -0.005 \\ -0.002 & -0.118 & 0.075 \\ -0.008 & -0.067 & 0.117 \end{bmatrix} \quad (3.33)$$

Table 3.4: Confusion matrix for probabilistic discriminative classification with MLE using acceleration data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	95.2	2	2.8
	Over	16.8	83.2	0
	Under	21.2	0	78.8
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	88.4	1.4	10.2
	Over	12	88	0
	Under	9.6	0	90.4
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	97.2	1.8	1
	Over	37.6	62.4	0
	Under	39.2	0	60.8
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	92.8	1.6	5.6
	Over	24.8	75.2	0
	Under	24.4	0	75.6

$\mathbf{W}_{MLE_{velocity}}$ computed using the entire velocity data set for training is given by

$$\mathbf{W}_{MLE_{velocity}} = \begin{bmatrix} 2.743 & -1.150 & -1.563 \\ -0.001 & 0.028 & 0.003 \\ 0.001 & 0.011 & -0.006 \\ 0.022 & -0.184 & 0.114 \\ -0.024 & -0.039 & 0.091 \end{bmatrix} \tag{3.34}$$

Table 3.5: Confusion matrix for probabilistic discriminative classification with MLE using velocity data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	95.3	1.9	2.8
	Over	19.2	80.8	0
	Under	22.8	0	77.2
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	87.6	1.8	10.6
	Over	14.4	85.6	0
	Under	9.6	0	90.4
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	96	1.6	2.4
	Over	46.4	53.6	0
	Under	44	0	56
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	91.8	1.7	6.5
	Over	30.4	69.6	0
	Under	26.8	0	73.2

$\mathbf{W}_{MLE_{displacement}}$ computed using the entire displacement data set for training is given by

$$\mathbf{W}_{MLE_{displacement}} = \begin{bmatrix} 2.680 & -0.910 & -1.757 \\ 0.0002 & 0.052 & -0.015 \\ 0.0003 & 0.012 & -0.020 \\ 0.024 & -0.294 & 0.264 \\ -0.056 & -0.059 & 0.130 \end{bmatrix} \tag{3.35}$$

Table 3.6: Confusion matrix for probabilistic discriminative classification with MLE using displacement data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	94.3	2.8	2.9
	Over	30.8	69.2	0
	Under	22.4	0	77.6
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	91.6	2.6	5.8
	Over	24.8	75.2	0
	Under	10.4	0.8	88.8
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	94.4	2.2	3.4
	Over	50.4	49.6	0
	Under	49.6	0	50.4
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	93	2.4	4.6
	Over	37.6	62.4	0
	Under	30	0.4	69.6

3.4 Discussion of Results in Table 3.4 to 3.6

The worst predictive performance is observed when we use only the displacement input for training. The decrease in performance is not only clear from the overall reduction in values of the diagonal cells, but also from the fact that some under-prediction input vectors are assigned to over-prediction class. Although the observed misclassification value, i.e., 0.8 % is not significant, this is so far the only model that classified an under-prediction input as an over-prediction. Until now, the okay-prediction acted as a "buffer" class between over- and under-prediction classes.

On the other hand, the predictive performances observed with acceleration and velocity show that those models can be used in the real-time applications without too much error in the form of misclassifications.

3.5 Classification – Probabilistic Discriminative Model with Maximum A Posteriori (MAP) Value

The following theory is based on the concepts described in Chapter 4, Linear Models for Classification in Bishop, 2006.

We now choose an informative prior and carry out a full Bayesian treatment of the probabilistic discriminative model. Because it is practically too difficult to evaluate analytically the posterior distribution which consists of multiplication of several softmax functions and a prior distribution, we make use of the Laplace approximation for the problem of learning the Bayesian probabilistic model. The Laplace approximation approximates a function by a Gaussian distribution whose center is located at a local maximum of that function. Because we need a covariance matrix to describe a Gaussian distribution, we will have to evaluate the Hessian matrix as well. Since we want to

approximate the posterior distribution by a Gaussian form, let us choose a Gaussian prior over the parameters. A general Gaussian prior distribution over the set of parameter is given by

$$\begin{aligned} p(\mathbf{w}) &= \mathrm{N}(\mathbf{w}|\mathbf{m}_0,\mathbf{S}_0) \\ &= \frac{1}{(2\pi)^{(D+1)/2}} \frac{1}{|\mathbf{S}_0|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{w}-\mathbf{m}_0)^T \mathbf{S}_0^{-1}(\mathbf{w}-\mathbf{m}_0)\right\} \end{aligned} \tag{3.36}$$

where $\mathbf{m}_0$ and $\mathbf{S}_0$ are general, but fixed, hyperparameters.

We chose the prior of each model parameter of each class to be a Gaussian PDF with zero mean and standard deviation $\sigma = 100$ to cover a wide range of the parameter space. Note that in the following sections σ will not be fixed, and we will let σ adapt to the training data. Then, the prior is

$$\begin{aligned} p(\mathbf{w}_k) &= \frac{1}{(2\pi\sigma^2)^{(D+1)/2}} \exp\left\{-\frac{1}{2}\mathbf{w}_k{}^T\mathbf{w}_k\right\} \\ &= \mathrm{N}(\mathbf{w}_k|\mathbf{0},\sigma^2\mathbf{I}) \end{aligned} \tag{3.37}$$

We ask ourselves the following question:

"If we knew the values of the parameters $\mathbf{w}_k$ of one class, would that affect our level of knowledge about the rest of the parameters?"

The answer is "no". Therefore, we can state that the prior distributions of parameters of different classes are independent and obtain the following

$$p(\mathbf{W}) = p(\mathbf{w}_1,\cdots,\mathbf{w}_K) = \prod_{k=1}^{K} p(\mathbf{w}_k) \tag{3.38}$$

Using Bayes' theorem, the posterior distribution over the parameters is given by

$$p\left(\mathbf{W}\middle|\mathbf{T},\mathbf{X}\right)\propto p\left(\mathbf{T}\middle|\mathbf{X},\mathbf{W}\right)\times p\left(\mathbf{W}\right)$$
$$posterior\propto likelihood\times prior \tag{3.39}$$

where the likelihood is given by (3.31), and the prior is given by (3.38).

We can now determine $\mathbf{W}$ by finding the most probable value of $\mathbf{W}$ given the data, which means we will determine $\mathbf{W}$ by maximizing the posterior distribution. This produces what is called the *maximum a posteriori (MAP)* value. We solved this problem by an algorithm provided by Matlab. Substituting (3.31) and (3.38) into (3.39), our posterior distribution over the parameters (without the normalizing factor) is given by

$$\begin{aligned} p\left(\mathbf{W}\middle|\mathbf{T},\mathbf{X}\right) &\propto p\left(\mathbf{T}\middle|\mathbf{X},\mathbf{W}\right)\times p\left(\mathbf{W}\right) \\ &\propto \prod_{n=1}^{N}\prod_{k=1}^{K} p\left(C_k\middle|\mathbf{x}_n,\mathbf{w}_k\right)^{t_{nk}}\times\prod_{k=1}^{K} p\left(\mathbf{w}_k\right) \end{aligned} \tag{3.40}$$

Note that in order to find the values $\mathbf{W}$ that maximize (3.40), we do not need the normalizing constant. Note also that maximizing (3.40) with respect to $\mathbf{W}$ is equivalent to minimizing the negative logarithm of (3.40) with respect to $\mathbf{W}$. Taking the logarithm of (3.40) gives

$$\ln p\left(\mathbf{W}\middle|\mathbf{T},\mathbf{X}\right)=\ln p\left(\mathbf{T}\middle|\mathbf{X},\mathbf{W}\right)+\ln p\left(\mathbf{W}\right)+const \tag{3.41}$$

where

$$\begin{aligned} \ln p\left(\mathbf{T}\middle|\mathbf{X},\mathbf{W}\right) &= \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\ln p\left(C_k\middle|\mathbf{x}_n,\mathbf{w}_k\right) \\ &= \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\ln y_{nk} \\ &= \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\ln\left(\frac{\exp\left(\mathbf{w}_k{}^T\mathbf{x}_n\right)}{\sum_j \exp\left(\mathbf{w}_j{}^T\mathbf{x}_n\right)}\right) \end{aligned} \tag{3.42}$$

and

$$\begin{aligned}\ln p(\mathbf{W}) &= \ln p(\mathbf{w}_1, \mathbf{w}_2 \cdots, \mathbf{w}_K) \\ &= \ln\left(\prod_{k=1}^{K} p(\mathbf{w}_k)\right) \\ &= \ln\left(p(\mathbf{w}_1)p(\mathbf{w}_2)\cdots p(\mathbf{w}_K)\right) \\ &= \sum_{k=1}^{K} \ln p(\mathbf{w}_k)\end{aligned} \tag{3.43}$$

From (3.37), we have

$$\begin{aligned}\ln p(\mathbf{w}_k) &= \ln\left(\frac{1}{\left(2\pi\sigma^2\right)^{(D+1)/2}} \exp\left\{-\frac{1}{2\sigma^2}\mathbf{w}_k{}^T\mathbf{w}_k\right\}\right) \\ &= \ln\left(\frac{1}{\left(2\pi\sigma^2\right)^{(D+1)/2}}\right) + \ln\left(\exp\left\{-\frac{1}{2\sigma^2}\mathbf{w}_k{}^T\mathbf{w}_k\right\}\right) \\ &= \ln\left(\left(2\pi\sigma^2\right)^{-(D+1)/2}\right) - \frac{1}{2\sigma^2}\mathbf{w}_k{}^T\mathbf{w}_k \\ &= -\frac{(D+1)}{2}\left(\ln 2\pi + \ln\sigma^2\right) - \frac{1}{2\sigma^2}\mathbf{w}_k{}^T\mathbf{w}_k \\ &= -\frac{(D+1)}{2}\ln 2\pi - \frac{(D+1)}{2}\ln\sigma^2 - \frac{1}{2\sigma^2}\sum_{d=0}^{D} w_{kd}{}^2\end{aligned} \tag{3.44}$$

Substituting (3.44) into (3.43), we obtain

$$\ln p(\mathbf{W}) = -\frac{1}{2\sigma^2}\sum_{k=1}^{K}\sum_{d=0}^{D} w_{kd}{}^2 + const \tag{3.45}$$

We maximized (3.40) with respect to $\mathbf{W}$ separately for acceleration, velocity, and displacement input and obtained the values shown below, in addition to the following confusion matrices:

$$\mathbf{W}_{MAP_{acceleration}} = \begin{bmatrix} 2.898 & -1.264 & -1.606 \\ -0.004 & 0.017 & 0.002 \\ 0.003 & 0.014 & -0.004 \\ -0.002 & -0.118 & 0.074 \\ -0.008 & -0.066 & 0.118 \end{bmatrix} \quad (3.46)$$

Table 3.7: Confusion matrix for probabilistic discriminative classification with MAP ($\sigma = 100$) using acceleration data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	95.2	2	2.8
	Over	16.8	83.2	0
	Under	21.2	0	78.8
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	88.4	1.4	10.2
	Over	12	88	0
	Under	9.6	0	90.4
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	97.2	1.8	1
	Over	37.6	62.4	0
	Under	39.2	0	60.8
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	92.8	1.6	5.6
	Over	24.8	75.2	0
	Under	24.4	0	75.6

$$\mathbf{W}_{MAP_{velocity}} = \begin{bmatrix} 2.744 & -1.150 & -1.563 \\ -0.003 & 0.026 & 0.0002 \\ 0.002 & 0.012 & -0.005 \\ 0.022 & -0.184 & 0.114 \\ -0.022 & -0.036 & 0.094 \end{bmatrix} \tag{3.47}$$

Table 3.8: Confusion matrix for probabilistic discriminative classification with MAP ($\sigma = 100$) using velocity data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	95.3	1.9	2.8
	Over	19.2	80.8	0
	Under	22.8	0	77.2
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	87.6	1.8	10.6
	Over	14.4	85.6	0
	Under	9.6	0	90.4
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	96	1.6	2.4
	Over	46.4	53.6	0
	Under	44	0	56
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	91.8	1.7	6.5
	Over	30.4	69.6	0
	Under	26.8	0	73.2

$$\mathbf{W}_{MAP_{displacement}} = \begin{bmatrix} 2.680 & -0.909 & -1.757 \\ 3.381e\text{-}05 & 0.052 & -0.015 \\ 3.961e\text{-}05 & 0.012 & -0.020 \\ 0.023 & -0.294 & 0.263 \\ -0.056 & -0.059 & 0.130 \end{bmatrix} \tag{3.48}$$

Table 3.9: Confusion matrix for probabilistic discriminative classification with MAP ($\sigma = 100$) using displacement.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	94.3	2.8	2.9
	Over	30.8	69.2	0
	Under	22.4	0	77.6
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	91.6	2.6	5.8
	Over	24.8	75.2	0
	Under	10.4	0.8	88.8
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	94.4	2.2	3.4
	Over	50.4	49.6	0
	Under	49.6	0	50.4
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	93	2.4	4.6
	Over	37.6	62.4	0
	Under	30	0.4	69.6

3.6 Discussion of Results in Table 3.7 to 3.9

Notice the similarities in results between MLE and MAP values with $\sigma = 100$ in the prior; although the parameter values are slightly different, the predictive performances as observed in the tables given above are exactly the same! This is due to the fact that the prior standard deviation $\sigma = 100$ makes our prior distribution "too broad". It is so broad that our prior acts like a flat prior and we note that the MAP result using a flat, i.e., infinitely broad prior is the same as the MLE result. In order to make use of the MAP method more efficiently, let us decrease the standard deviation σ. The following values are computed by choosing $\sigma = 10$.

$$\mathbf{W}_{MAP_{acceleration}} = \begin{bmatrix} 2.894 & -1.266 & -1.608 \\ -0.006 & 0.014 & 4.463e\text{-}05 \\ 0.001 & 0.012 & -0.005 \\ 0.001 & -0.115 & 0.077 \\ -0.012 & -0.070 & 0.113 \end{bmatrix} \quad (3.49)$$

Table 3.10: Confusion matrix for probabilistic discriminative classification with MAP ($\sigma = 10$) using acceleration data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	95.2	2	2.8
	Over	16.8	83.2	0
	Under	21.2	0	78.8
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	88.4	1.4	10.2
	Over	12	88	0
	Under	9.6	0	90.4
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	97.2	1.8	1
	Over	37.6	62.4	0
	Under	39.2	0	60.8
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	92.8	1.6	5.6
	Over	24.8	75.2	0
	Under	24.4	0	75.6

$$\mathbf{W}_{MAP_{velocity}} = \begin{bmatrix} 2.737 & -1.156 & -1.568 \\ -0.003 & 0.025 & -7.034e\text{-}05 \\ 0.0004 & 0.010 & -0.007 \\ 0.026 & -0.179 & 0.119 \\ -0.026 & -0.041 & 0.089 \end{bmatrix} \tag{3.50}$$

Table 3.11: Confusion matrix for probabilistic discriminative classification with MAP ($\sigma = 10$) using velocity data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	95.3	1.9	2.8
	Over	19.2	80.8	0
	Under	22.8	0	77.2
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	87.6	1.8	10.6
	Over	14.4	85.6	0
	Under	9.6	0	90.4
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	96	1.6	2.4
	Over	46.4	53.6	0
	Under	44	0	56
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	91.8	1.7	6.5
	Over	30.4	69.6	0
	Under	26.8	0	73.2

$$\mathbf{W}_{MAP_{displacement}} = \begin{bmatrix} 2.678 & -0.911 & -1.757 \\ -0.0001 & 0.052 & -0.015 \\ -5.469e\text{-}05 & 0.012 & -0.020 \\ 0.024 & -0.293 & 0.264 \\ -0.057 & -0.061 & 0.129 \end{bmatrix} \tag{3.51}$$

Table 3.12: Confusion matrix for probabilistic discriminative classification with MAP ($\sigma = 10$) using displacement data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	94.3	2.8	2.9
	Over	30.8	69.2	0
	Under	22.4	0	77.6
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	91.6	2.6	5.8
	Over	24.8	75.2	0
	Under	10.4	0.8	88.8
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	94.4	2.2	3.4
	Over	50.4	49.6	0
	Under	49.6	0	50.4
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	93	2.4	4.6
	Over	37.6	62.4	0
	Under	30	0.4	69.6

3.7 Discussion of Results in Table 3.10 to 3.12

Just like the case with $\sigma = 100$, the MAP results with $\sigma = 10$ are the same as the MLE results; although the parameter values are slightly different, the predictive performances as observed in the tables given above are exactly the same! This is due to the fact that the standard deviation $\sigma = 10$ (as chosen above for the prior over parameter values) makes our new prior distribution "too broad" again. It is so broad that our prior acts like a flat prior just like before. In order to make use of the MAP method more efficiently, we decreased the standard deviation more drastically; we chose $\sigma = 0.5$. Once again, the results were not too different. Then, we decided to use $\sigma = 0.05$ and the predictive performances changed notably. So, the following results are obtained using $\sigma = 0.05$.

$$\mathbf{W}_{MAP\ acceleration} = \begin{bmatrix} 0.810 & -0.410 & -0.399 \\ -0.005 & 0.003 & 0.002 \\ 0.001 & 0.003 & -0.005 \\ 0.030 & -0.048 & 0.018 \\ -0.027 & -0.026 & 0.054 \end{bmatrix} \quad (3.52)$$

Table 3.13: Confusion matrix for probabilistic discriminative classification with MAP ($\sigma = 0.05$) using acceleration data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	89	4.2	6.8
	Over	14.4	85.6	0
	Under	12.8	0	87.2
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	70	5.6	24.4
	Over	12	88	0
	Under	3.2	0	96.8
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	83.6	7.4	9
	Over	16	81.6	2.4
	Under	24	0	76
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	76.8	6.5	16.7
	Over	14	84.8	1.2
	Under	13.6	0	86.4

$$\mathbf{W}_{MAP_{velocity}} = \begin{bmatrix} 0.805 & -0.399 & -0.406 \\ -0.007 & 0.006 & 0.0003 \\ 0.002 & 0.002 & -0.004 \\ 0.037 & -0.073 & 0.036 \\ -0.027 & -0.016 & 0.043 \end{bmatrix} \tag{3.53}$$

Table 3.14: Confusion matrix for probabilistic discriminative classification with MAP ($\sigma = 0.05$) using velocity data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	89.2	3.4	7.4
	Over	11.2	88.8	0
	Under	14.4	0	85.6
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	74.4	5.4	20.2
	Over	17.6	82.4	0
	Under	4	0	96
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	79	6.2	14.8
	Over	13.6	86.4	0
	Under	28.8	0	71.2
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	76.7	5.8	17.5
	Over	15.6	84.4	0
	Under	16.4	0	83.6

$$\mathbf{W}_{\substack{MAP \\ displacement}} = \begin{bmatrix} 0.783 & -0.386 & -0.396 \\ -0.010 & 0.014 & -0.004 \\ 0.006 & 0.004 & -0.010 \\ 0.038 & -0.113 & 0.075 \\ -0.044 & -0.021 & 0.065 \end{bmatrix} \tag{3.54}$$

Table 3.15: Confusion matrix for probabilistic discriminative classification with MAP ($\sigma = 0.05$) using displacement data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	87.4	5	7.6
	Over	22.8	77.2	0
	Under	13.6	0	86.4
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	78.6	8.8	12.6
	Over	27.2	72.8	0
	Under	8	0	92
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	79.8	5.6	14.6
	Over	20.8	79.2	0
	Under	32	0	68
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	79.2	7.2	12.6
	Over	24	76	0
	Under	20	0.4	80

3.8 Discussion of Results in Table 3.13 to 3.15

After choosing $\sigma = 0.05$, we started to observe a significant increase in the prediction performance for over- and under-prediction classes. Although there is some reduction in predictive performance for okay-prediction class, the overall performance is still acceptable, i.e., at lease 70% accuracy. This showed me the effect of the prior. Choosing a very broad prior does not help us make efficient use of MAP technique. But there is still the question of how to choose the appropriate σ. One quick answer would be "experience". Fortunately, we can use a method called automatic relevance determination prior (ARD prior) where we treat σ as an unknown value and let it adapt to the data set itself! This way, we will look for "sparseness" in our model. In the upcoming section, we show the results of another powerful technique called Bayesian Model Class Selection. Then, we move to the part where we use the ARD prior.

3.9 Classification – Posterior Predictive Distribution

In the previous section of our work, we computed $\mathbf{W}_{MAP}$ by determining the value of $\mathbf{W}$ which maximizes the posterior distribution, which is proportional to the product of the likelihood and the prior distribution. In the multiclass classification case, we obtained the likelihood as defined by (3.31). We chose a prior on the parameters $\mathbf{W}$, which we defined by (3.38). Then we obtained the functional form of the posterior distribution over the parameters $\mathbf{W}$. However, the posterior distribution over the parameters is a complicated expression, as it is a product of many softmax functions and a Gaussian in the form of the prior distribution. We wanted to "approximate" this posterior by a Gaussian function with a mean centered at $\mathbf{W}_{MAP}$, which is the value that maximizes the original functional form of the posterior. In other words, we decided to use Laplace

approximation to approximate our original posterior by a Gaussian function. One might think: "we can get $\mathbf{W}_{MAP}$ by maximizing our original posterior. So, why do we want to come up with a Gaussian approximation as well?" The reason is that we aim to be able to make new predictions $p\left(C_k\middle|\mathbf{T},\mathbf{X},\mathbf{x}\right)$ for a new input feature vector $\mathbf{x}$, rather than to know the value of the parameters $\mathbf{W}$. That means we want to evaluate the posterior predictive distribution, which is computed by marginalizing with respect to $\mathbf{W}$ using its posterior distribution. Approximating our original posterior over $\mathbf{W}$ by a Gaussian makes evaluating the predictive distribution easier.

As I mentioned above, we obtain the posterior predictive distribution for class C_k, for a newly observed input vector $\mathbf{x}$, when we marginalize with respect to the posterior distribution over $\mathbf{W}$, which we decided to approximate by a Gaussian. Therefore, the posterior predictive distribution is given by

$$p\left(C_k\middle|\mathbf{T},\mathbf{X},\mathbf{x}\right)=\int p\left(C_k\middle|\mathbf{T},\mathbf{X},\mathbf{x},\mathbf{w}_k\right)q\left(\mathbf{W}\middle|\mathbf{T},\mathbf{X}\right)d\mathbf{W} \tag{3.55}$$

where $q\left(\mathbf{W}\middle|\mathbf{T},\mathbf{X}\right)$ is the approximation to the posterior distribution over $\mathbf{W}$ which is a Gaussian centered at $\mathbf{W}_{MAP}$, and the covariance matrix

$$\mathbf{S}_N^{-1}=-\nabla\nabla\ln p\left(\mathbf{W}\middle|\mathbf{T},\mathbf{X}\right)\Big|_{\mathbf{W}=\mathbf{W}_{MAP}} \tag{3.56}$$

is given in terms of the Hessian matrix of the log posterior, which is a $\left(D+1\right)K\times\left(D+1\right)K$ matrix, where $D+1$ is the number of parameters in $\mathbf{w}_k$ and K is the number of classes. That is to say, this Hessian matrix is made of $\left(D+1\right)\times\left(D+1\right)$ size blocks where the block j,k is

$$\nabla_{\mathbf{w}_k}\nabla_{\mathbf{w}_j}-\ln p\left(\mathbf{W}\middle|\mathbf{T},\mathbf{X}\right) \tag{3.57}$$

with $\nabla_{\mathbf{w}_j}$ representing the gradient with respect to parameter vector $\mathbf{w}_j$ (Bishop 2006).

In practice, evaluating the integral in (3.55), even with only a few dimensional parameter space, is not easy. Fortunately, this integral can be approximated by a discrete weighted average of predictive PDFs for each model in a given class of models (Beck and Katafygiotis 1998). Therefore, using the Laplace asymptotic approximation to evaluate the integral in (3.55), we obtain

$$\begin{aligned} p\left(C_k\middle|\mathbf{T},\mathbf{X},\mathbf{x}\right) &= \int p\left(C_k\middle|\mathbf{T},\mathbf{X},\mathbf{x},\mathbf{w}_k\right)q\left(\mathbf{W}\middle|\mathbf{T},\mathbf{X}\right)d\mathbf{W} \\ &\approx p\left(C_k\middle|\mathbf{T},\mathbf{X},\mathbf{x},\hat{\mathbf{w}}_k\right) \end{aligned} \tag{3.58}$$

where $\hat{\mathbf{w}}_k \triangleq k^{th}$ column of MAP of $\mathbf{W}$.

3.10 Classification – Bayesian Model Class Selection (Method I)

The following is a brief discussion of Bayesian model class selection, based on lecture notes from a course titled "Stochastic System Analysis and Bayesian Updating" taught by Prof. James L. Beck at Caltech. Also see (Beck and Yuen 2004) and (Beck 2010) for more details.

Until now, I did not suggest many different models as far as the input feature vector components are concerned; all types of features, i.e., derivative of kurtosis and skewness – both horizontal and vertical channels were used in our models. This may lead to over-parameterization, as I will explain shortly. In fact, I did not propose a robust methodology to overcome the over-parameterization problem. I will do that in this section. I will propose several stochastic models and compare them to choose the best performing one among them; based on the data $\left\{\mathbf{T},\mathbf{X}\right\}$, I will compare alternative

stochastic system models (SSMs) for my multiclass classification system. This is known as Bayesian model (class) selection or comparison but I will use assessment.

Posterior Probability of Alternative Stochastic System Models

Let us assume we have a set of J candidate system models $\{M_1,\cdots,M_J\}$ for a system specified by proposition $\mathbf{M}$. $\mathbf{M}$ also specifies a prior distribution over this set, $P(M_j|\mathbf{M})\forall j=1,\cdots,J$. The posterior probability of each SSM is given by Bayes' theorem:

$$P(M_j|\mathbf{T},\mathbf{X})=\frac{p(\mathbf{T}|\mathbf{X},M_j)P(M_j|\mathbf{M})}{p(\mathbf{T},\mathbf{X}|\mathbf{M})} \tag{3.59}$$

where the evidence (or marginal likelihood) for M_j given by $\{\mathbf{T},\mathbf{X}\}$ is $p(\mathbf{T}|\mathbf{X},M_j)$:

$$p(\mathbf{T}|\mathbf{X},M_j)=\int p(\mathbf{T}|\mathbf{X},\mathbf{W}_j,M_j)p(\mathbf{W}_j|M_j)d\mathbf{W}_j \tag{3.60}$$

To indicate different M_j might have different numbers of parameters, we use $\mathbf{W}_j$ there. Looking at (3.59), we notice that the only data-dependent term is the evidence. The posterior probability of each model class $p(M_j|\mathbf{T},\mathbf{X})$ is controlled by it.

Evaluation of evidence for M

We can use Laplace's method to approximate the integral in (3.60) if M is globally identifiable and the number of data points is sufficiently large. If $\mathbf{W}_{MAP}$ is the MAP value for $\mathbf{W}$ under $\{\mathbf{T},\mathbf{X}\}$, then

$$p(\mathbf{T}|\mathbf{X},M)\approx\frac{p(\mathbf{T}|\mathbf{X},\mathbf{W}_{MAP},M)p(\mathbf{W}_{MAP}|M)(2\pi)^{D/2}}{|H_N(\mathbf{W}_{MAP})|^{1/2}} \tag{3.61}$$

where

$$H_N\left(\mathbf{W}\right) \triangleq -\nabla\nabla \ln p\left(\mathbf{T}\middle|\mathbf{X},\mathbf{W},M\right) - \nabla\nabla \ln p\left(\mathbf{W}\middle|M\right) \tag{3.62}$$

Parsimonious Models and Ockham's Razor

Using a quantitative form of William of Ockham's (Occam's) razor will help us avoid "over-parameterization" or "over-fitting" of the data when identifying model from data. It quantifies Ockham's philosophy that can be paraphrased as "Don't multiply entities unnecessarily." It is obvious that we cannot solely use the fit of the model to the data to implement Ockham's razor because a more complex model (with addition of more parameters) will always have an improved fit.

Principle of Model Parsimony

We can rank the plausibility of a set of J candidate models $\bigcup_{j=1}^{J}\left\{M_j\right\}$ for a system based on data by their posterior probability using Bayes' Theorem and this will automatically penalize the fit to the data of each M_j by a measure of the "complexity" of the model.

Proof

Let us first introduce the following notation:

$$\begin{aligned}
&\text{Evidence,} && EV\left(M_j\middle|\mathbf{T},\mathbf{X}\right) \triangleq p\left(\mathbf{T}\middle|\mathbf{X},M_j\right) \\
&\text{Likelihood,} && L\left(\mathbf{W}_j\middle|\mathbf{T},\mathbf{X},M_j\right) \triangleq p\left(\mathbf{T}\middle|\mathbf{X},\mathbf{W}_j,M_j\right) \\
&\text{Ockham factor,} && OF\left(\mathbf{W}_j\middle|\mathbf{T},\mathbf{X},M_j\right) \triangleq \left|\hat{H}_N\left(\mathbf{W}_j\right)\right|^{-\frac{1}{2}}\left(2\pi\right)^{D_j/2}\, p\left(\mathbf{W}_j\middle|M_j\right)
\end{aligned} \tag{3.63}$$

where

$$\hat{H}_N\left(\mathbf{W}_j\right) \triangleq H_N\left(\mathbf{W}_j\right) \triangleq -\nabla_{\mathbf{w}_j}\nabla_{\mathbf{w}_j} \ln L\left(\mathbf{W}_j\middle|\mathbf{T},\mathbf{X},M_j\right) - \nabla_{\mathbf{w}_j}\nabla_{\mathbf{w}_j} \ln p\left(\mathbf{W}_j\middle|M\right) \tag{3.64}$$

From Bayes' Theorem:

$$\ln P\left(M_j\middle|\mathbf{T},\mathbf{X},\mathbf{M}\right)=\underbrace{\ln EV\left(M_j\middle|\mathbf{T},\mathbf{X}\right)}_{O(N)}+\underbrace{\ln P\left(M_j\middle|\mathbf{M}\right)}_{O(1)}+\text{normalizing constant} \qquad (3.65)$$

Explanation using Laplace's asymptotic approximation

Let us assume each M_j is globally identifiable under $\{\mathbf{T},\mathbf{X}\}$ and N is sufficiently large. If we use $\hat{\mathbf{W}}$ as an optimal parameter value for M_j, e.g., $\mathbf{W}_{MAP}{}^{(j)}$:

$$\begin{aligned}\ln EV\left(M_j\middle|\mathbf{T},\mathbf{X}\right)&\approx \ln L\left(\hat{\mathbf{W}}_j\middle|\mathbf{T},\mathbf{X}\right)+\ln OF\left(\hat{\mathbf{W}}_j\middle|\mathbf{T},\mathbf{X}\right)\\&=\ln L\left(\hat{\mathbf{W}}_j\middle|\mathbf{T},\mathbf{X}\right)-\frac{1}{2}D_j\ln N+\left[\frac{1}{2}D_j\ln(2\pi)-\frac{1}{2}\ln\left|\frac{1}{N}\hat{H}_N\left(\hat{\mathbf{W}}_j\right)\right|+\ln p\left(\hat{\mathbf{W}}_j\middle|M_j\right)\right]\end{aligned} \qquad (3.66)$$

The first term on the right hand side of (3.66) is $O(N)$, the second term is $O(\ln N)$ and the last three terms are $O(1)$. Also note that $\hat{H}_N\left(\hat{\mathbf{W}}_j\right)=O(N)$ and:

$$\left|\hat{H}_N\left(\hat{\mathbf{W}}_j\right)\right|=\left|\left(N\mathbf{I}_{D_j}\right)\left(\frac{1}{N}\hat{H}_N\left(\hat{\mathbf{W}}_j\right)\right)\right|=N^{D_j}\left|\frac{1}{N}\hat{H}_N\left(\hat{\mathbf{W}}_j\right)\right| \qquad (3.67)$$

The first term on the right hand side of (3.66), the log likelihood term of $O(N)$ ($\ln L\left(\hat{\mathbf{W}}_j\middle|\mathbf{T},\mathbf{X}\right)$), gives a measure of the data fit for the model M_j, which is specified by $\hat{\mathbf{W}}_j$. The second term $-\frac{1}{2}D_j\ln N$, which is $O(\ln N)$, gives a bias against over-parameterization; the number of parameters D_j can be considered a simple way of measuring the complexity of M_j.

Recall that in Chapter 2 we indicated that we need at least one skewness measure to be able to distinguish between over- and under-prediction cases. Keeping in mind that

we should include at least one measure of skewness in all of the proposed models, we obtained Table 3.16. Note that we assume each alternative model is equally plausible a priori, so they have the same prior probability. Therefore, we can choose the optimum model among the list of proposed ones by maximizing only the value of their evidence.

Bayesian model class selection results with acceleration input are given in Table 3.17. Table 3.17 shows that the model 10 has the highest evidence value. The following parameter values are computed for model 10 with acceleration input:

$$\mathbf{W}_{MAP_{acceleration}} = \begin{bmatrix} 0.814 & -0.405 & -0.409 \\ 0.023 & -0.055 & 0.031 \\ -0.016 & -0.016 & 0.032 \\ -0.004 & 0.005 & -0.001 \end{bmatrix} \tag{3.68}$$

Note that the structure of the matrix above is slightly different from the one used so far: The columns still represent coefficients for different classes, i.e., first column is for okay-prediction class, second column and third column are for over- and under-prediction classes respectively and the first row is for the dummy input, i.e., $x_0 = 1$. However, the second row is the coefficient for derivative of horizontal skewness, the third row is for derivative of vertical skewness, and the fourth row is for derivative of horizontal kurtosis.

Bayesian model class selection results with velocity input are given in Table 3.18. Table 3.18 shows that the model 10 has the highest evidence value. The following parameter values are computed for model 10 with velocity input:

$$\mathbf{W}_{MAP_{velocity}} = \begin{bmatrix} 0.809 & -0.398 & -0.411 \\ 0.031 & -0.077 & 0.046 \\ -0.017 & -0.008 & 0.025 \\ -0.005 & 0.008 & -0.002 \end{bmatrix} \tag{3.69}$$

The structure of the matrix above is the same as for (3.68).

Bayesian model class selection results with displacement input are given in Table 3.19. Table 3.19 shows that the model 12 has the highest evidence value (with model 10 giving a fairly close second highest value). This model happens to be the one we have been using for the displacement input since the beginning of our classification. One might expect to see the same parameter values as (3.54) because these models are the same. However, the optimization algorithm I used is a stochastic one and we may get different parameter values even though the difference is not significant. The following parameter values are computed for model 12 with displacement input:

$$\mathbf{W}_{MAP_{displacement}} = \begin{bmatrix} 0.783 & -0.386 & -0.396 \\ 0.038 & -0.113 & 0.075 \\ -0.044 & -0.021 & 0.065 \\ -0.010 & 0.014 & -0.004 \\ 0.006 & 0.004 & -0.010 \end{bmatrix} \tag{3.70}$$

Table 3.16: List of features included in alternative models.

Model	$\frac{d}{dt}\left(\text{Horizontal Skewness}\right)$	$\frac{d}{dt}\left(\text{Vertical Skewness}\right)$	$\frac{d}{dt}\left(\text{Horizontal Kurtosis}\right)$	$\frac{d}{dt}\left(\text{Vertical Kurtosis}\right)$
1	YES	NO	NO	NO
2	NO	YES	NO	NO
3	YES	NO	YES	NO
4	YES	NO	NO	YES
5	NO	YES	YES	NO
6	NO	YES	NO	YES
7	YES	YES	NO	NO
8	YES	NO	YES	YES
9	NO	YES	YES	YES
10	YES	YES	YES	NO
11	YES	YES	NO	YES
12	YES	YES	YES	YES

Table 3.17: Bayesian model class selection results for acceleration input.

Model	$\ln L$	$\ln OF$	$\ln EV$
1	-796.6905878	-192.7533557	-989.4439435
2	-823.9070973	-188.880831	-1012.787928
3	-753.3949416	-218.6867016	-972.0816432
4	-750.6115197	-221.6867784	-972.2982981
5	-768.6552803	-222.1422717	-990.797552
6	-779.6578831	-213.1267318	-992.7846149
7	-775.8544848	-198.9565556	-974.8110404
8	-742.3141105	-232.81717	-975.1312805
9	-758.881568	-230.6736099	-989.5551779
10	-732.2044958	-224.7531183	-956.9576141
11	-739.7176866	-220.9872754	-960.704962
12	-728.4732518	-231.9182509	-960.3915028

Table 3.18: Bayesian model class selection results for velocity input.

Model	$\ln L$	$\ln OF$	$\ln EV$
1	-804.1471411	-191.4777145	-995.6248557
2	-891.3299278	-174.8778011	-1066.207729
3	-756.8400798	-216.7472545	-973.5873343
4	-770.7451819	-214.740959	-985.4861409
5	-819.9112367	-213.4734752	-1033.384712
6	-856.6070691	-196.2207625	-1052.827832
7	-790.3019239	-197.4815243	-987.7834483
8	-751.2525198	-228.3901151	-979.6426349
9	-815.0599131	-220.5028865	-1035.5628
10	-743.1684811	-222.7195605	-965.8880416
11	-762.8911059	-216.1464477	-979.0375535
12	-741.937857	-229.4082319	-971.3460889

Table 3.19: Bayesian model class selection results for displacement input.

Model	$\ln L$	$\ln OF$	$\ln EV$
1	-869.3012837	-180.4653091	-1049.766593
2	-989.0176498	-156.287171	-1145.304821
3	-803.7357577	-209.6282134	-1013.363971
4	-837.2586776	-202.5986302	-1039.857308
5	-905.2722922	-198.4699835	-1103.742276
6	-949.0037145	-175.5844723	-1124.588187
7	-855.8888103	-187.0384564	-1042.927267
8	-801.9021195	-219.0314594	-1020.933579
9	-887.0434732	-203.7117187	-1090.755192
10	-792.8375594	-215.1139263	-1007.951486
11	-820.2266283	-204.006558	-1024.233186
12	-788.3007944	-219.1738527	-1007.474647

3.11 Classification – Sparse Bayesian Learning (Method II)

In this section, we describe Sparse Bayesian Learning (SBL) (Tipping 2001), as used in the relevance vector machine (RVM) for sparse kernel regression and classification (Bishop 2006) and which we use in the reality check algorithm (RCA) for the actual earthquake early warning system, i.e., CISN ShakeAlert. We again use the linear probabilistic model where we computed maximum posterior (MAP) values for the parameters. However, there is an important distinction: we use a prior where the precision (inverse variance) for each parameter is also a variable, i.e., it is not fixed and it will be estimated as a function of the training data. This separate variable is called a *hyperparameter*.

Let us start with introducing the ARD (Automatic Relevance Determination) prior:

$$p\left(\mathbf{W}\middle|\mathbf{A}\right)=\prod_{k=1}^{K}p\left(\mathbf{w}_{k}\middle|\mathbf{a}_{k}\right) \tag{3.71}$$

where

$$p\left(\mathbf{w}_{k}\middle|\mathbf{a}_{k}\right)=\prod_{d=0}^{D}p\left(w_{kd}\middle|\alpha_{kd}\right) \tag{3.72}$$

where

$$p\left(w_{kd}\middle|\alpha_{kd}\right)=\mathrm{N}\left(w_{kd}\middle|0,\alpha_{kd}^{-1}\right) \tag{3.73}$$

Then, we obtain

$$\begin{aligned} p\left(\mathbf{W}\middle|\mathbf{A}\right)&=\prod_{k=1}^{K}\prod_{d=0}^{D}p\left(w_{kd}\middle|\alpha_{kd}\right)\\ &=\prod_{k=1}^{K}\prod_{d=0}^{D}\frac{\sqrt{\alpha_{kd}}}{\sqrt{2\pi}}\exp\left\{-\frac{\alpha_{kd}}{2}w_{kd}^{\ 2}\right\} \end{aligned} \tag{3.74}$$

where

$$\mathbf{A}=\begin{bmatrix} \alpha_{10} & \alpha_{20} & \cdots & \alpha_{K0} \\ \alpha_{11} & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{1D} & \alpha_{2D} & \cdots & \alpha_{KD} \end{bmatrix} \tag{3.75}$$

Taking the logarithm of (3.74), we obtain

$$\begin{aligned} \ln p\left(\mathbf{W}\middle|\mathbf{A}\right) &= \sum_{k=1}^{K}\sum_{d=0}^{D}\ln\left(\frac{\sqrt{\alpha_{kd}}}{\sqrt{2\pi}}\right)-\frac{\alpha_{kd}}{2}w_{kd}^{\ 2} \\ &= \sum_{k=1}^{K}\sum_{d=0}^{D}\ln\left(\sqrt{\alpha_{kd}}\right)-\ln\left(\sqrt{2\pi}\right)-\frac{\alpha_{kd}}{2}w_{kd}^{\ 2} \\ &= \sum_{k=1}^{K}\sum_{d=0}^{D}\frac{1}{2}\ln\left(\alpha_{kd}\right)-\frac{1}{2}\ln\left(2\pi\right)-\frac{\alpha_{kd}}{2}w_{kd}^{\ 2} \end{aligned} \tag{3.76}$$

We shall see that when we maximize the evidence with respect to these hyperparameters, a significant portion of them go to infinity, and the corresponding weight parameters then have posterior distributions with zero mean and variance so they are concentrated at zero, causing their corresponding basis functions to have coefficients equal to zero. Thus, the basis functions associated with these parameters play no role in the predictions made by the model, that is, they are effectively pruned out, resulting in a sparse model (Bishop 2006).

We want to maximize the posterior over $\mathbf{A}$:

$$p\left(\mathbf{A}\middle|\mathbf{T},\mathbf{X}\right)\propto p\left(\mathbf{T}\middle|\mathbf{A},\mathbf{X}\right)\times p\left(\mathbf{A}\right) \tag{3.77}$$

Let us define the hyperprior over $\mathbf{A}$. We choose Gamma distributions as suitable priors (Bishop and Tipping 2003):

$$p\left(\mathbf{A}\right)=\prod_{k=1}^{K}p\left(\mathbf{a}_{k}\right) \tag{3.78}$$

where

$$p\left(\mathbf{a}_{k}\right)=\prod_{d=0}^{D}Gamma\left(\alpha_{kd}\middle|a,b\right) \tag{3.79}$$

where

$$Gamma\left(\alpha\middle|a,b\right)=\Gamma\left(a\right)^{-1}b^{a}\alpha^{a-1}e^{-b\alpha} \tag{3.80}$$

then, we obtain

$$p\left(\mathbf{A}\right)=\prod_{k=1}^{K}\prod_{d=0}^{D}Gamma\left(\alpha_{kd}\middle|a,b\right) \tag{3.81}$$

At the end of this process, we will obtain equations to re-estimate the hyperparameters.

Next, we obtain $p\left(\mathbf{T}\middle|\mathbf{X},\mathbf{A}\right)$ by first deriving the posterior over all unknowns, given the data (Bishop and Tipping 2003):

$$p\left(\mathbf{W},\mathbf{A}\middle|\mathbf{T},\mathbf{X}\right)=p\left(\mathbf{W}\middle|\mathbf{A},\mathbf{T},\mathbf{X}\right)p\left(\mathbf{A}\middle|\mathbf{T},\mathbf{X}\right) \tag{3.82}$$

$$p\left(\mathbf{W}\middle|\mathbf{A},\mathbf{T},\mathbf{X}\right)=\frac{p\left(\mathbf{T}\middle|\mathbf{W},\mathbf{A},\mathbf{X}\right)p\left(\mathbf{W}\middle|\mathbf{A}\right)}{p\left(\mathbf{T}\middle|\mathbf{A},\mathbf{X}\right)} \tag{3.83}$$

$$p\left(\mathbf{T}\middle|\mathbf{A},\mathbf{X}\right)=\frac{p\left(\mathbf{T}\middle|\mathbf{W},\mathbf{A},\mathbf{X}\right)p\left(\mathbf{W}\middle|\mathbf{A}\right)}{p\left(\mathbf{W}\middle|\mathbf{A},\mathbf{T},\mathbf{X}\right)} \tag{3.84}$$

Maximizing (3.77) is equivalent to minimizing its negative logarithm:

$$\begin{aligned}-\ln p\left(\mathbf{A}\middle|\mathbf{T},\mathbf{X}\right)&=-\ln p\left(\mathbf{T}\middle|\mathbf{A},\mathbf{X}\right)-\ln p\left(\mathbf{A}\right)+const\\&=-\ln p\left(\mathbf{T}\middle|\mathbf{W}^{*},\mathbf{A},\mathbf{X}\right)-\ln p\left(\mathbf{W}^{*}\middle|\mathbf{A}\right)+\ln p\left(\mathbf{W}^{*}\middle|\mathbf{A},\mathbf{T},\mathbf{X}\right)-\ln p\left(\mathbf{A}\right)+const\end{aligned} \tag{3.85}$$

Define $\mathbf{W}^{*}$ as the MAP value for the posterior over $\mathbf{W}$ for a fixed value of $\mathbf{A}$.

Therefore, $\mathbf{W}^{*}$ is calculated by minimizing

$$E\left(\mathbf{W}\right)=-\ln p\left(\mathbf{W}\middle|\mathbf{T},\mathbf{X},\mathbf{A}\right)=-\ln p\left(\mathbf{T}\middle|\mathbf{X},\mathbf{W},\mathbf{A}\right)-\ln p\left(\mathbf{W}\middle|\mathbf{A}\right)+const \tag{3.86}$$

where $p\left(\mathbf{W}\middle|\mathbf{A}\right)$ is given by (3.71).

The third term on the right hand side of (3.85) can be approximated by Laplace's asymptotic approximation as

$$\ln p\left(\mathbf{W}\middle|\mathbf{A},\mathbf{T},\mathbf{X}\right)\approx\ln\left[\mathrm{N}\left(\mathbf{W}\middle|\mathbf{W}^{*},H\left(\mathbf{W}^{*}\right)^{-1}\right)\right] \tag{3.87}$$

Substituting for $\mathbf{W}$ by its MAP value $\mathbf{W}^*$:

$$\ln p\left(\mathbf{W}^*\middle|\mathbf{A},\mathbf{T},\mathbf{X}\right)=\frac{1}{2}\ln\left|H\left(\mathbf{W}^*\right)\right|+const \tag{3.88}$$

where $H\left(\mathbf{W}^*\right)$ is the Hessian matrix evaluated at $\mathbf{W}^*$ and given by:

$$H\left(\mathbf{W}\right)=\nabla\nabla E\left(\mathbf{W}\right) \tag{3.89}$$

Therefore, we can write (3.85) using $E\left(\mathbf{W}\right)$ and (3.88) as

$$-\ln p\left(\mathbf{A}\middle|\mathbf{T},\mathbf{X}\right)=E\left(\mathbf{W}^*\right)+\frac{1}{2}\ln\left|H\left(\mathbf{W}^*\right)\right|-\ln p\left(\mathbf{A}\right)+const \tag{3.90}$$

In order to find the equation that will update the hyperparameters $\mathbf{A}$, we take the derivative of (3.90) with respect to individual hyperparameters α_{kd}. We see that $E\left(\mathbf{W}^*\right)$ is a function of α_{kd} in two ways: a direct way through the terms involving the prior on $\mathbf{W}$ and indirectly by the MAP value $\mathbf{W}^*$, which depends on the value of α_{kd}. However, taking advantage of the fact that $\mathbf{W}^*$ is the MAP value so that the second part has derivative zero (Zhang and Malik 2005):

$$\begin{aligned}\frac{\partial E\left(\mathbf{W}^*\right)}{\partial\alpha_{kd}}&=\left.\frac{\partial E\left(\mathbf{W}^*\right)}{\partial\alpha_{kd}}\right|_{\text{fixed }\mathbf{W}^*}+\underbrace{\left.\frac{\partial E\left(\mathbf{W}^*\right)}{\partial\mathbf{W}^*}\right|_{\text{fixed }\alpha}}_{0}\frac{\partial\mathbf{W}^*}{\partial\alpha_{kd}}\\&=\left.\frac{\partial E\left(\mathbf{W}^*\right)}{\partial\alpha_{kd}}\right|_{\text{fixed }\mathbf{W}^*}+0\frac{\partial\mathbf{W}^*}{\partial\alpha_{kd}}\\&=\frac{1}{2}\left(\mathbf{W}^*_{kd}-\frac{1}{\alpha_{kd}}\right)\end{aligned} \tag{3.91}$$

We next make use of the expression provided in Bishop, 2006, C.22:

$$\frac{\partial\ln\left|\mathbf{A}\right|}{\partial x}=Tr\left(\mathbf{A}^{-1}\frac{\partial\mathbf{A}}{\partial x}\right) \tag{3.92}$$

to get the derivative of the second term in (3.90):

$$\frac{1}{2}\frac{\partial}{\partial\alpha_{kd}}\ln\left|H\left(\mathbf{W}^{*}\right)\right| = \frac{1}{2}Tr\left(H\left(\mathbf{W}^{*}\right)^{-1}\frac{\partial H\left(\mathbf{W}^{*}\right)}{\partial\alpha_{kd}}\right) = \frac{1}{2}\left[H\left(\mathbf{W}^{*}\right)^{-1}\right]_{kd} = \frac{1}{2}\Sigma_{kd} \tag{3.93}$$

where Σ_{kd} is the $\left(k\times d\right)^{th}$ diagonal element of $H\left(\mathbf{W}^{*}\right)^{-1}$.

Finally, let us take a look at derivative of the third term in (3.90) with respect to α_{kd} (Zhang and Malik 2005). From (3.81), we have

$$\frac{\partial\left(-\ln p\left(\mathbf{A}\right)\right)}{\partial\alpha_{kd}} = b - \frac{a}{\alpha_{kd}} \tag{3.94}$$

We set the final form of the derivatives to zero, and then we obtain:

$$\alpha_{kd}^{\ NEW} = \frac{1-\alpha_{kd}\Sigma_{kd}+2a}{\mathbf{W}^{*}_{\ kd}+2b} \tag{3.95}$$

We use Jeffrey hyperpriors:

$$a = b = 0 \Rightarrow p\left(\alpha_{kd}\right) \propto \alpha_{kd}^{\ -1} \tag{3.96}$$

Therefore,

$$\alpha_{kd}^{\ NEW} = \frac{1-\alpha_{kd}\Sigma_{kd}}{\mathbf{W}^{*}_{\ kd}} \tag{3.97}$$

An iterative procedure is used to find the MAP values. We first assign initial values to $\mathbf{A}$, and then evaluate $\mathbf{W}^{*}$ for those values by minimizing $E\left(\mathbf{W}^{*}\right)$. Then, we re-estimate $\mathbf{A}$ using (3.97). After that, we re-estimate $\mathbf{W}^{*}$, and so on until a convergence criterion is satisfied. We can stop the iterations when the change in the norm of $\mathbf{W}^{*}$ (converted into a column vector form) is less than 5% compared to the previous iteration. Each column of $\mathbf{W}$ represents the parameter values for different classes: the first column is for the okay-prediction, the second column is for the over-prediction, and the third column is for the under-prediction class. The input feature vector is chosen as:

$$\mathbf{x}=\begin{bmatrix} 1 \\ \frac{d}{dt}\left(\text{Horizontal Skewness}\right) \\ \frac{d}{dt}\left(\text{Vertical Skewness}\right) \\ \frac{d}{dt}\left(\text{Horizontal Kurtosis}\right) \\ \frac{d}{dt}\left(\text{Vertical Kurtosis}\right) \end{bmatrix} \tag{3.98}$$

Therefore, when a parameter is pruned out from the model, the corresponding input vector element (of the corresponding) class will not be used in the predictions.

Using the above convergence criterion, we obtain the following results for acceleration input:

$$\mathbf{W}_{MAP_{acceleration}}=\begin{bmatrix} 4.176 & 0.005 & -0.159 \\ -1.182\text{e-}08 & -0.117 & 0.082 \\ 3.291\text{e-}06 & -0.057 & 0.096 \\ -0.004 & 0.016 & 3.459\text{e-}09 \\ 3.413\text{e-}07 & 0.011 & -0.002 \end{bmatrix} \tag{3.99}$$

From (3.99) we can prune out $\mathbf{W}_{MAP_{acceleration}}\left(2,1\right)$, $\mathbf{W}_{MAP_{acceleration}}\left(3,1\right)$, $\mathbf{W}_{MAP_{acceleration}}\left(5,1\right)$, and $\mathbf{W}_{MAP_{acceleration}}\left(4,3\right)$ from the model by setting them to zero.

Using the convergence criterion, we obtain the following results for velocity input:

$$\mathbf{W}_{MAP_{velocity}}=\begin{bmatrix} 3.916 & 0.009 & -0.248 \\ -3.212\text{e-}05 & -0.212 & 0.102 \\ 0.0002 & -0.006 & 0.091 \\ -0.001 & 0.028 & 8.984\text{e-}08 \\ 3.673\text{e-}09 & 0.009 & -0.003 \end{bmatrix} \tag{3.100}$$

Similar to the acceleration input case, we can prune out the following parameters: $\mathbf{W}_{MAP_{velocity}}(2,1)$, $\mathbf{W}_{MAP_{velocity}}(3,1)$, $\mathbf{W}_{MAP_{velocity}}(5,1)$, and $\mathbf{W}_{MAP_{velocity}}(4,3)$. Note that $\mathbf{W}_{MAP_{velocity}}(3,1)$ is not as small as the corresponding value in the acceleration case given in (3.99), but it is relatively small compared to the rest of the parameter values in the first column in (3.100).

Using the convergence criterion, we obtain the following results for displacement input:

$$\mathbf{W}_{MAP_{displacement}} = \begin{bmatrix} 3.597 & 0.026 & -0.701 \\ -1.623e\text{-}06 & -0.316 & 0.227 \\ -0.001 & -0.003 & 0.168 \\ 8.383e\text{-}05 & 0.0530 & -0.013 \\ -2.521e\text{-}09 & 0.010 & -0.017 \end{bmatrix} \tag{3.101}$$

We can therefore prune out the following parameters: $\mathbf{W}_{MAP_{displacement}}(2,1)$, $\mathbf{W}_{MAP_{displacement}}(4,1)$ and $\mathbf{W}_{MAP_{displacement}}(5,1)$.

In the next few chapters, we analyze the performances produced by the models given above.

Chapter 4

Case Studies – Okay Predictions

In this chapter, we demonstrate the performance of the Reality Check Algorithm (RCA) via several example case studies. We start with two trivial examples of okay-predictions, and then we explain the performance on one of the examples with small perturbations in arrival times of the predicted envelopes. We compute the class probabilities by using the Sparse Bayesian Learning (SBL) technique (Method II). In order to show the superiority of Sparse Bayesian Learning (the models with the ARD prior) over the classification model chosen by applying Ockham's razor on a given set of models (Method I), comparison of their performances are displayed in several figures below.

The following is a general figure description, which is used in the following figures where necessary.

General figure description:

Left panel: first row shows the seismogram recorded by the seismic station whose identification information is given above (seismogram in blue and its envelope in red); note that although only the vertical channel is shown in the plot for demonstration purposes, both vertical and horizontal channels were used in RCA computations; the second row shows both the predicted and observed envelopes; the third row shows the test function after high-pass filtering by the values given in Chapter 2. Right panel shows the probability values for each class with a different color and marker.

4.1 Okay-Prediction Examples

Before we start demonstrating RCA's performance, we clarify the definition of okay-prediction used in this thesis. According to the Decision Module Review Tool adopted by the scientists working on the California earthquake early warning project, a prediction made by the Decision Module (DM) is considered accurate if the origin time error is less than or equal to 30 seconds, and the location error is less than or equal to 100 kilometers, relative to the Advanced National Seismic System (ANSS) composite catalogue (this is the default match criteria in the DM Review Tool Web Page Description for the CISN ShakeAlert project). Because our algorithm is sensitive enough to classify predictions with these amounts of error as inaccurate and RCA has the potential to be a stand-alone algorithm, we use the ANSS catalogue values as the location and magnitude when creating the predicted envelopes. We also match the P-wave arrival times of the predicted envelopes with that of observed ones for seismic stations of interest. By doing this, we aim to decrease the margin of error associated with RCA's performance. We indicate when we do not follow this pattern. For quantitative definition of the okay-predictions, please see Chapter 2.

4.1.1 Okay-prediction example 1

Let us start with an event for which the ANSS catalogue indicates the following information gathered from the Southern California Earthquake Data Center:

Event ID: 37314320

Magnitude: 4.89

Latitude: 31.5237

Longitude: -115.6743

We show the performance of RCA for the seismic station with the following information:

Network: CI

Station: JEM

Type: Strong Motion Seismometer

4.1.2 Discussion of okay-prediction example 1

Figures 4.1 to 4.6 show results of Method I and Method II on the okay-prediction example 1. Observe the noisy nature of the results for the models chosen using Method I. These models have significant fluctuations within each class compared to the results we computed using the models from Method II. This observation can be made regardless of the frequency content of the ground motion; that is, acceleration, velocity, and displacement all show significant fluctuations within each class probability values that are computed using Method I, while Method II probability values are close to each other for a given class in comparison. In addition to the noisy looking values, this particular technique resulted in a smaller gap between class probabilities compared with the ARD prior models, which show greater separation between the accurate class and the other two.

Furthermore, it is observed that the displacement based results are the least accurate among all three, i.e., acceleration, velocity, and displacement. The decrease in performance is more significant in Figure 4.3 where the computations are done using Method I. In fact, as far as Method II is concerned, the RCA performance with displacement input is relatively acceptable, as clearly seen in Figure 4.6 where the computations are performed using a model with the ARD prior. However, the discrepancy associated with displacement indicates that sampling the envelopes of

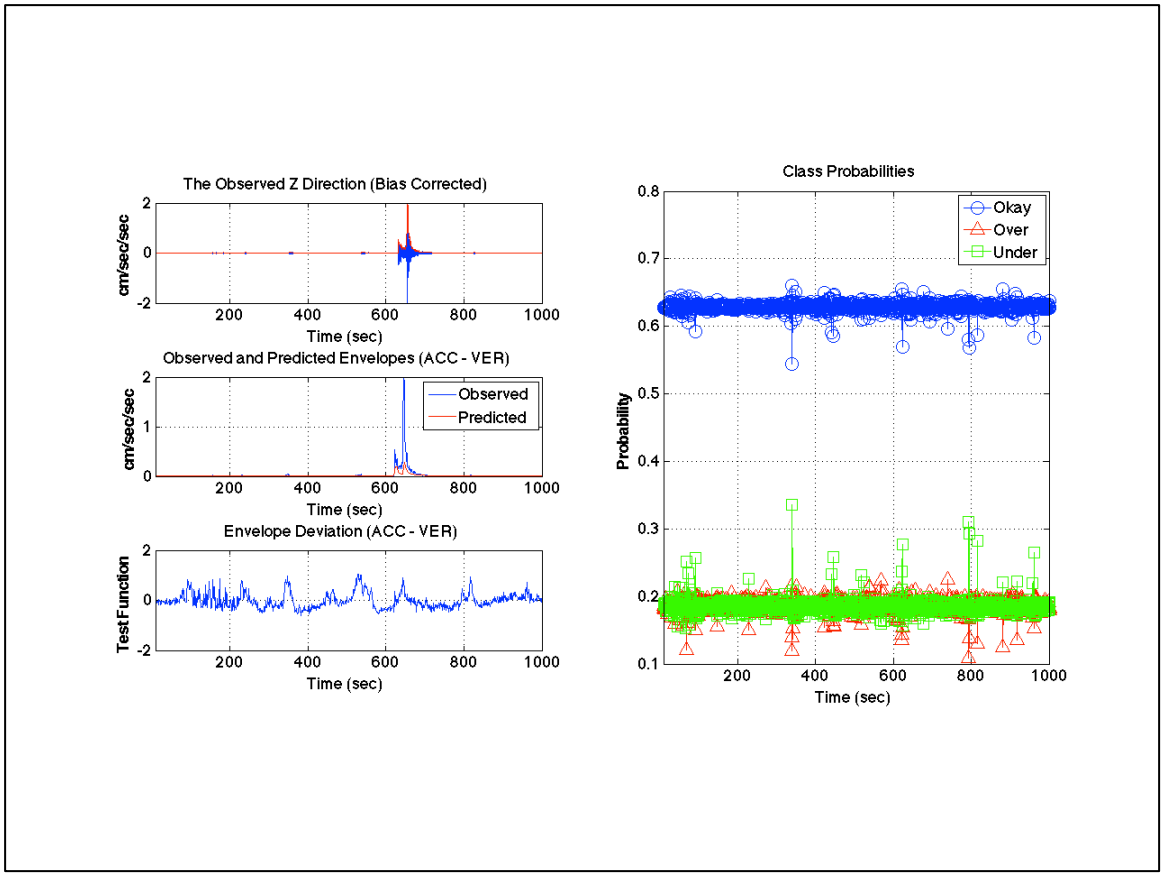

Figure 4.1: Reality Check Algorithm's performance plot for the okay-prediction example 1 computed using (3.68) from acceleration input using Method I. For description of the figure, see *"General figure description"* given above.

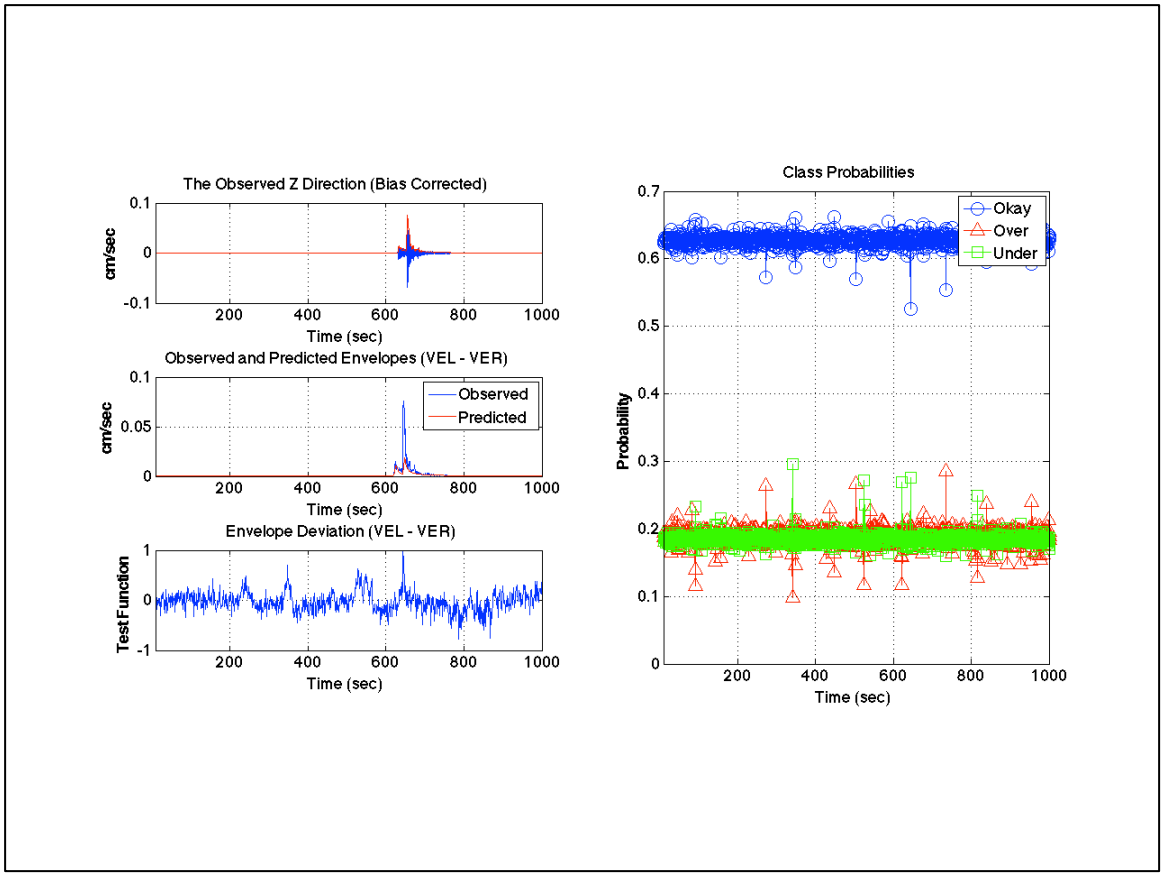

Figure 4.2: Reality Check Algorithm's performance plot for the okay-prediction example 1 computed using (3.69) from velocity input using Method I. For description of the figure, see *"General figure description"* given above.

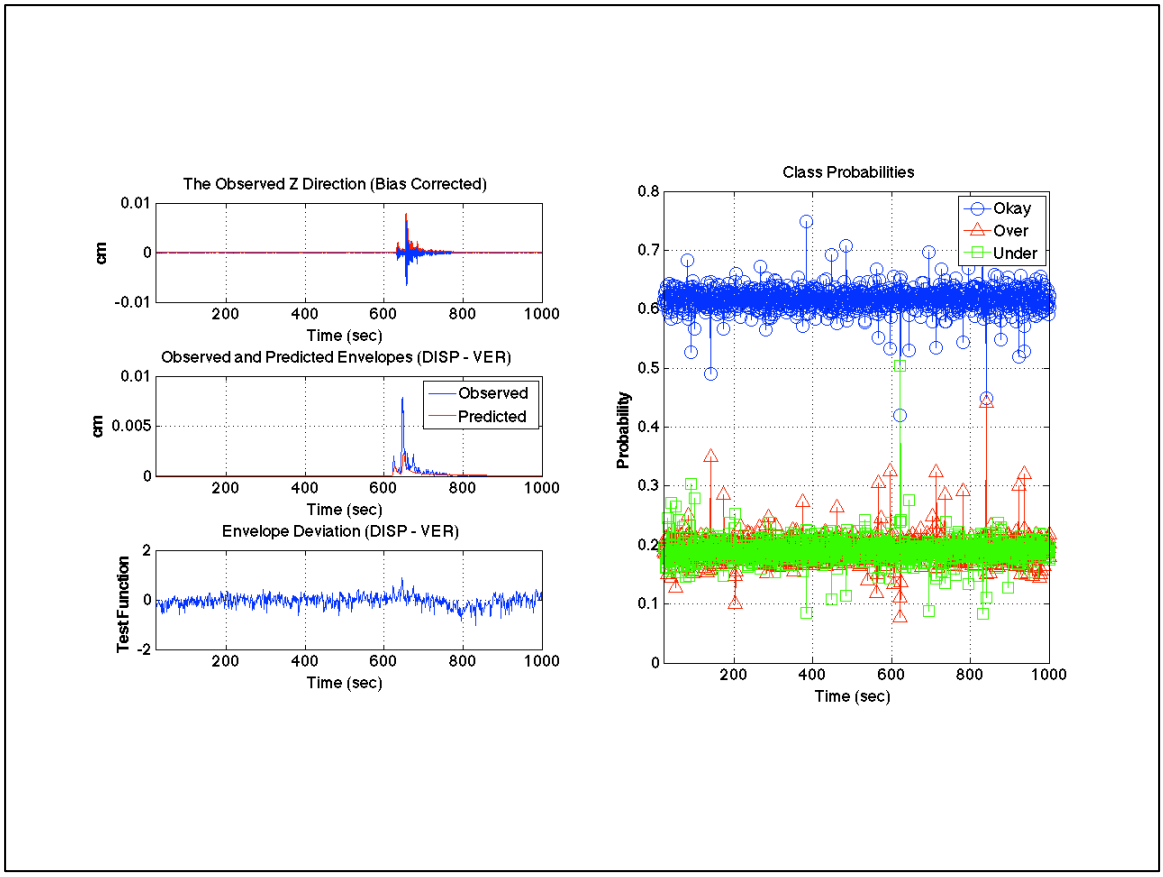

Figure 4.3: Reality Check Algorithm's performance plot for the okay-prediction example 1 computed using (3.70) from displacement input using Method I. For description of the figure, see *"General figure description"* given above.

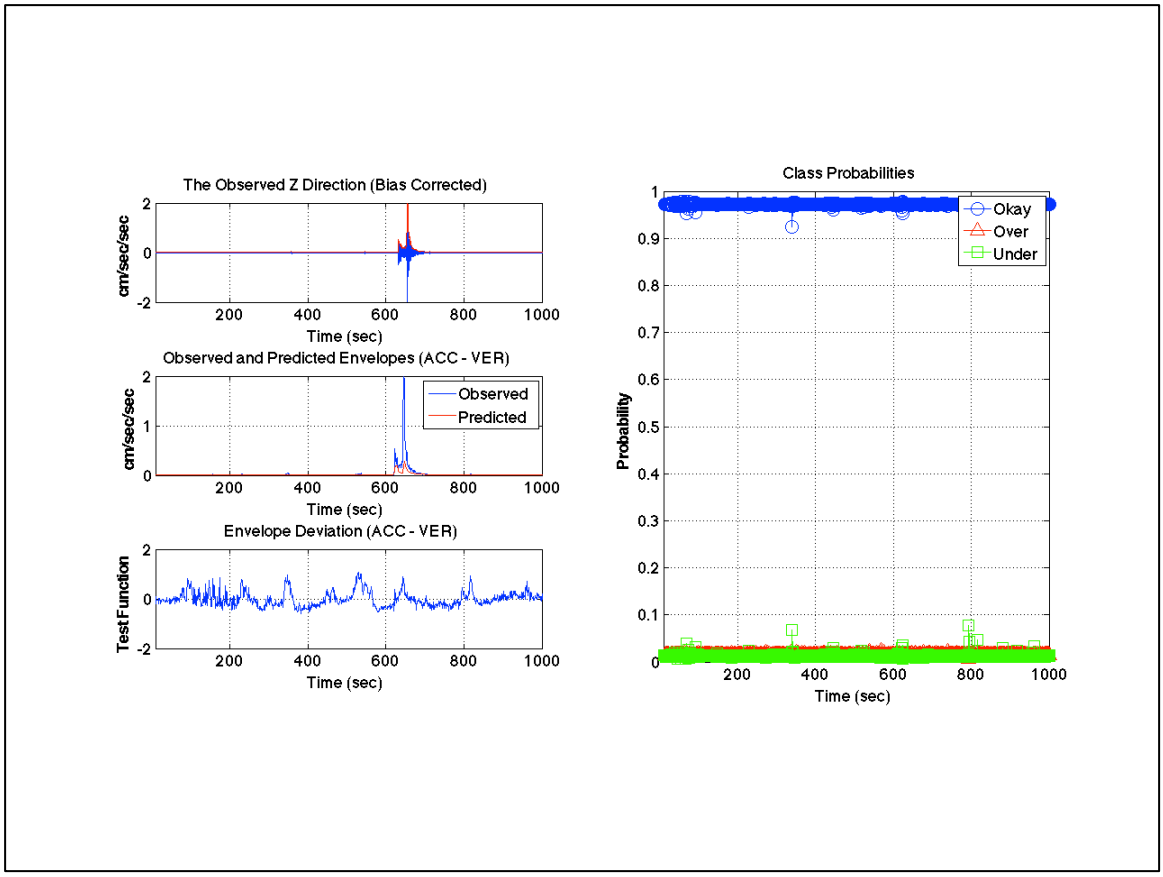

Figure 4.4: Reality Check Algorithm's performance plot for the okay-prediction example 1 computed using (3.99) from acceleration input using Method II. For description of the figure, see *"General figure description"* given above.

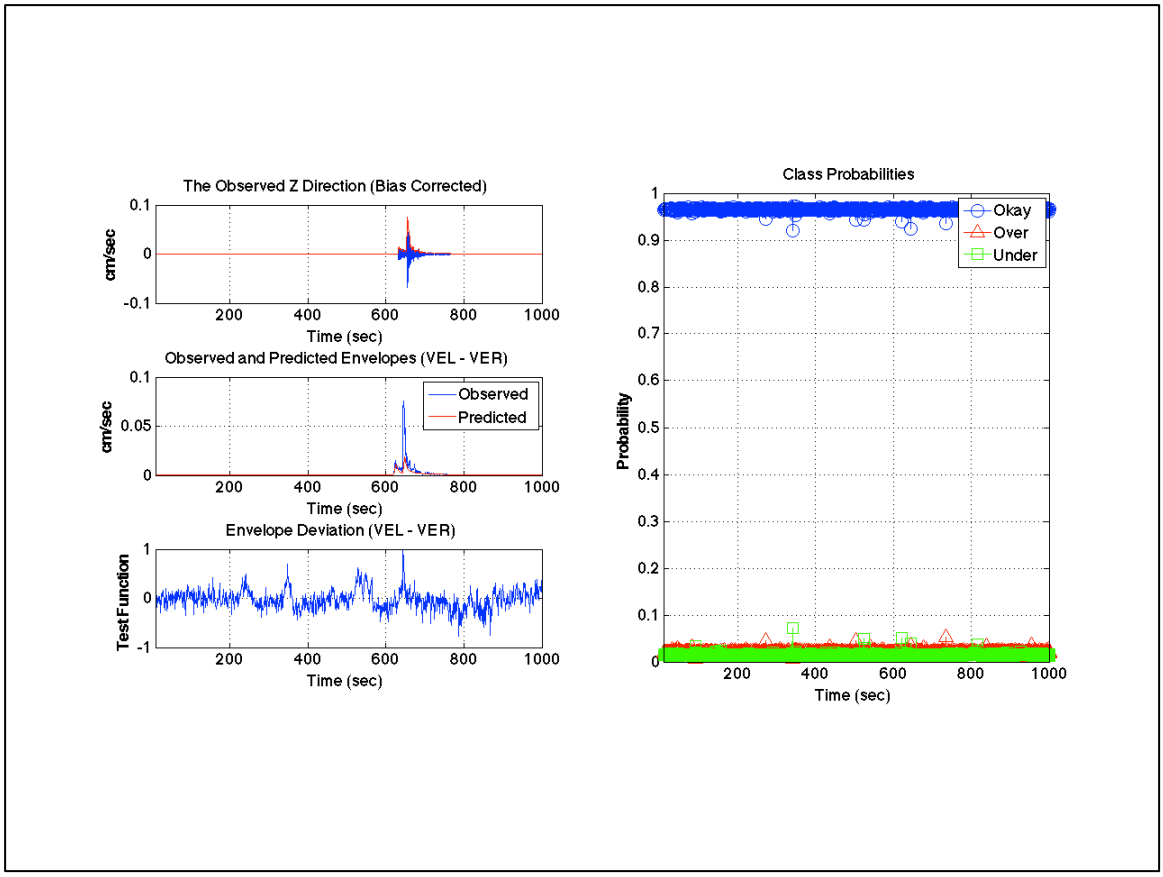

Figure 4.5: Reality Check Algorithm's performance plot for the okay-prediction example 1 computed using (3.100) from velocity input using Method II. For description of the figure, see *"General figure description"* given above.

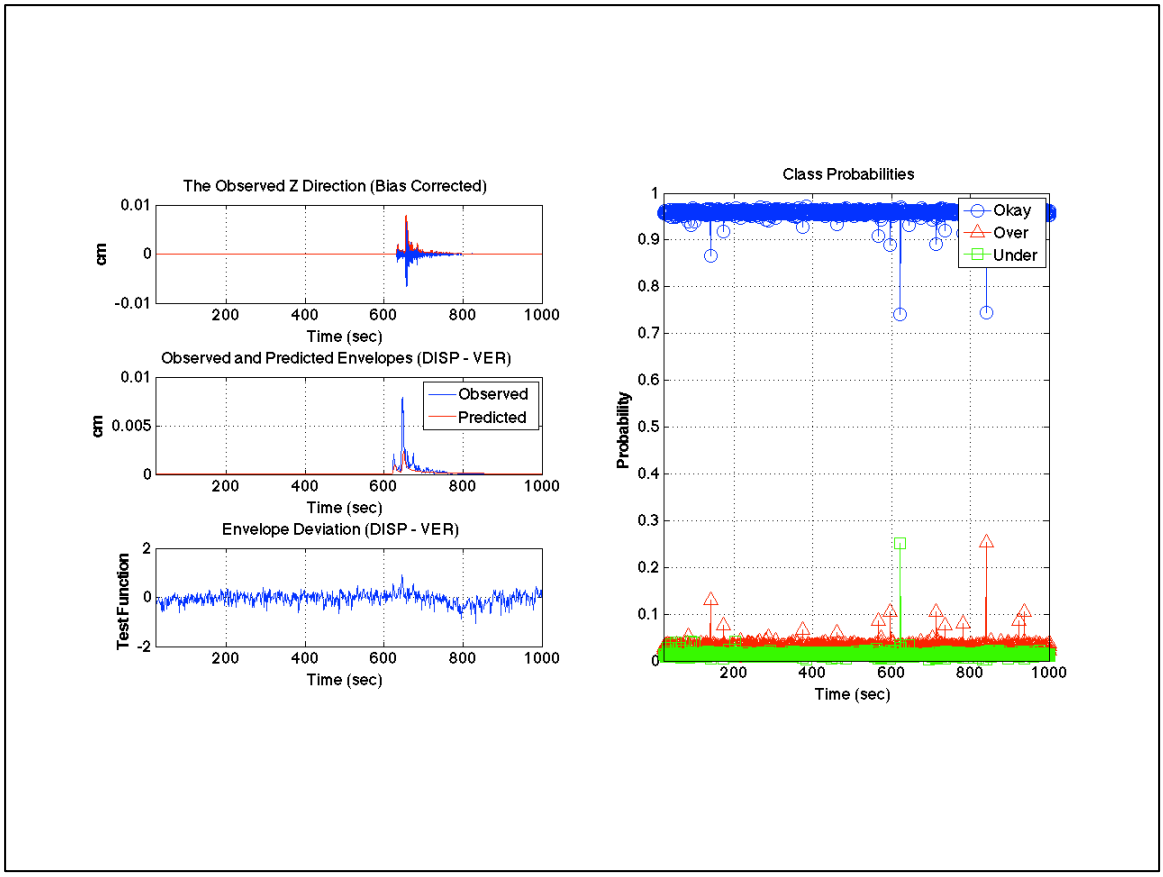

Figure 4.6: Reality Check Algorithm's performance plot for the okay-prediction example 1 computed using (3.101) from displacement input using Method II. For description of the figure, see *"General figure description"* given above.

ground motions at one-second intervals (see Chapter 2) may not be suitable for low-frequency content, i.e., displacement ground motions. One may want to increase the number of seconds at which the displacement envelopes are created. Moreover, we may need another measure of misfit, which would somehow guarantee an acceptable prediction class when combined with RCA results. We introduce such a new method below.

4.1.3 Okay-prediction example 2

We continue the RCA demonstration with another event.

Recall that the predicted P-wave arrival time is matched with that of the observed one.

Event ID: 37301704

Magnitude: 4.25

Latitude: 34.6173

Longitude: -118.6302

I will show the performance of RCA for the seismic station with the following information:

Network: CI

Station: SLM

Type: Strong Motion Seismometer

4.1.4 Discussion of okay-prediction example 2

The same pattern of ‘noisy results’ versus ‘less noisy results’ can be observed in Figures 4.7 to 4.12 as in Figures 4.1 to 4.6. This observation shows us that the Method II is superior to Method I.

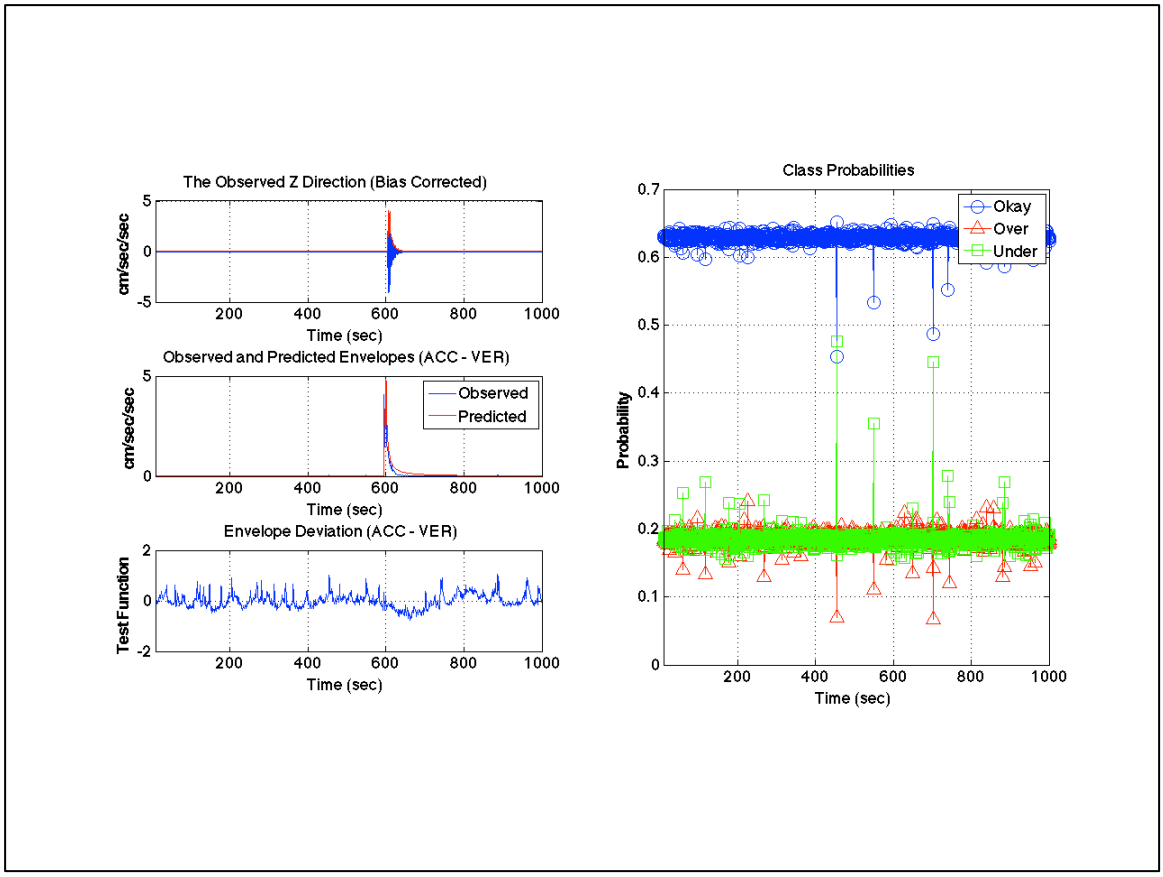

Figure 4.7: Reality Check Algorithm's performance plot for the okay-prediction example 2 computed using (3.68) from acceleration input using Method I. For description of the figure, see *"General figure description"* given above.

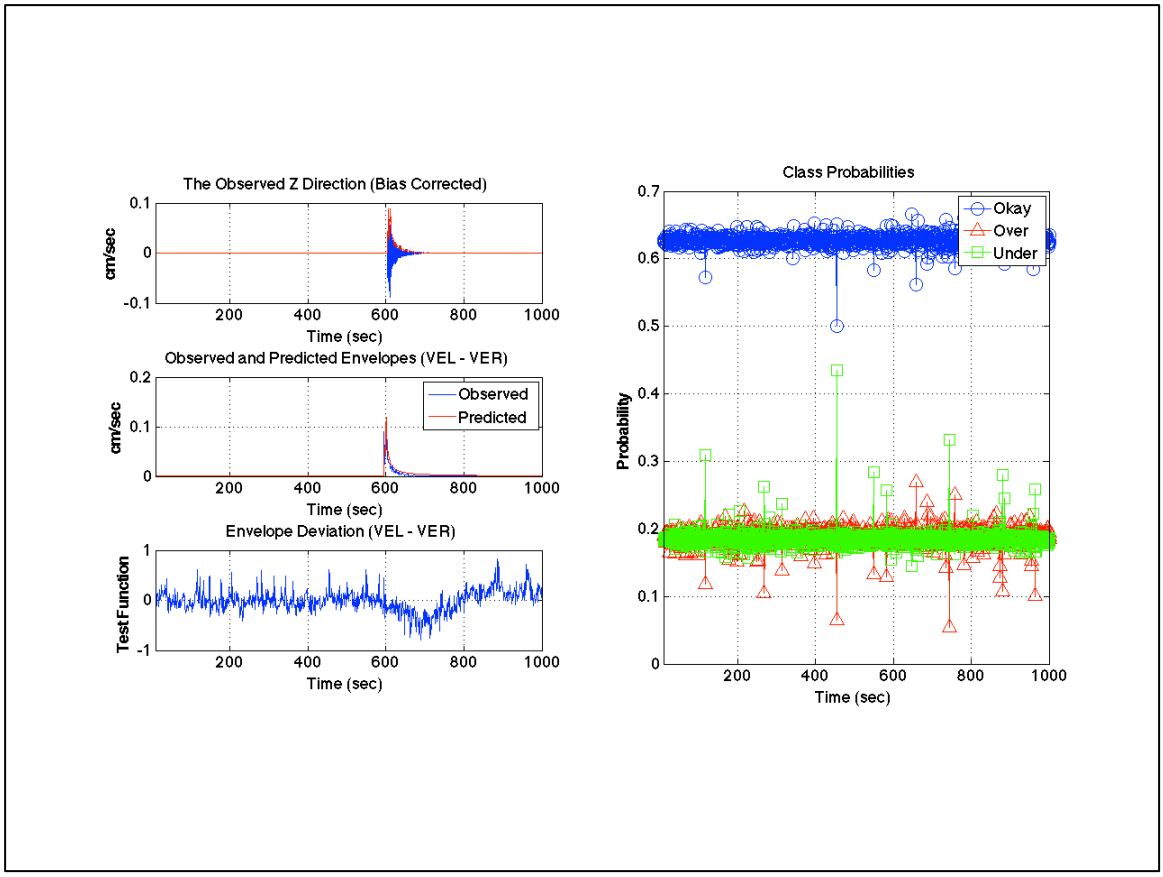

Figure 4.8: Reality Check Algorithm's performance plot for the okay-prediction example 2 computed using (3.69) from velocity input using Method I. For description of the figure, see *"General figure description"* given above.

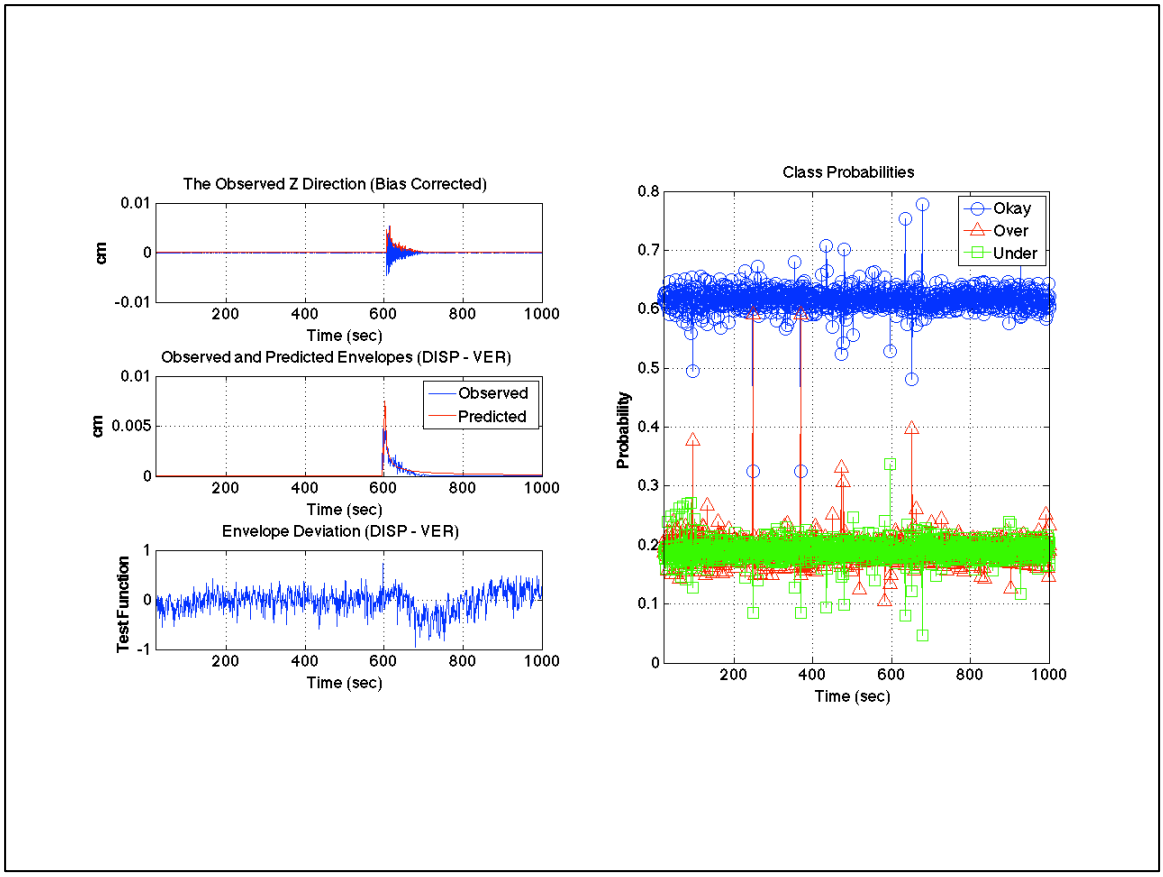

Figure 4.9: Reality Check Algorithm's performance plot for the okay-prediction example 2 computed using (3.70) from displacement input using Method I. For description of the figure, see *"General figure description"* given above.

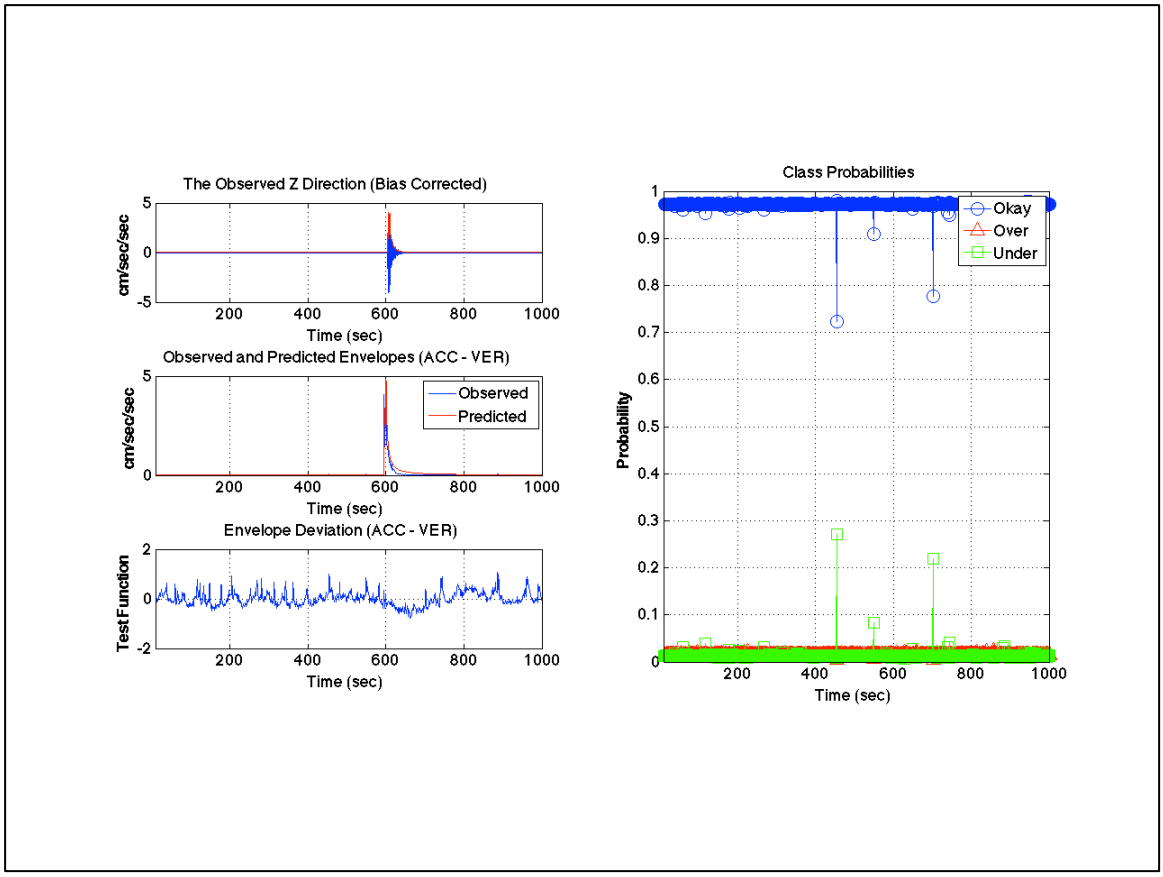

Figure 4.10: Reality Check Algorithm's performance plot for the okay-prediction example 2 computed using (3.99) from acceleration input using Method II. For description of the figure, see *"General figure description"* given above.

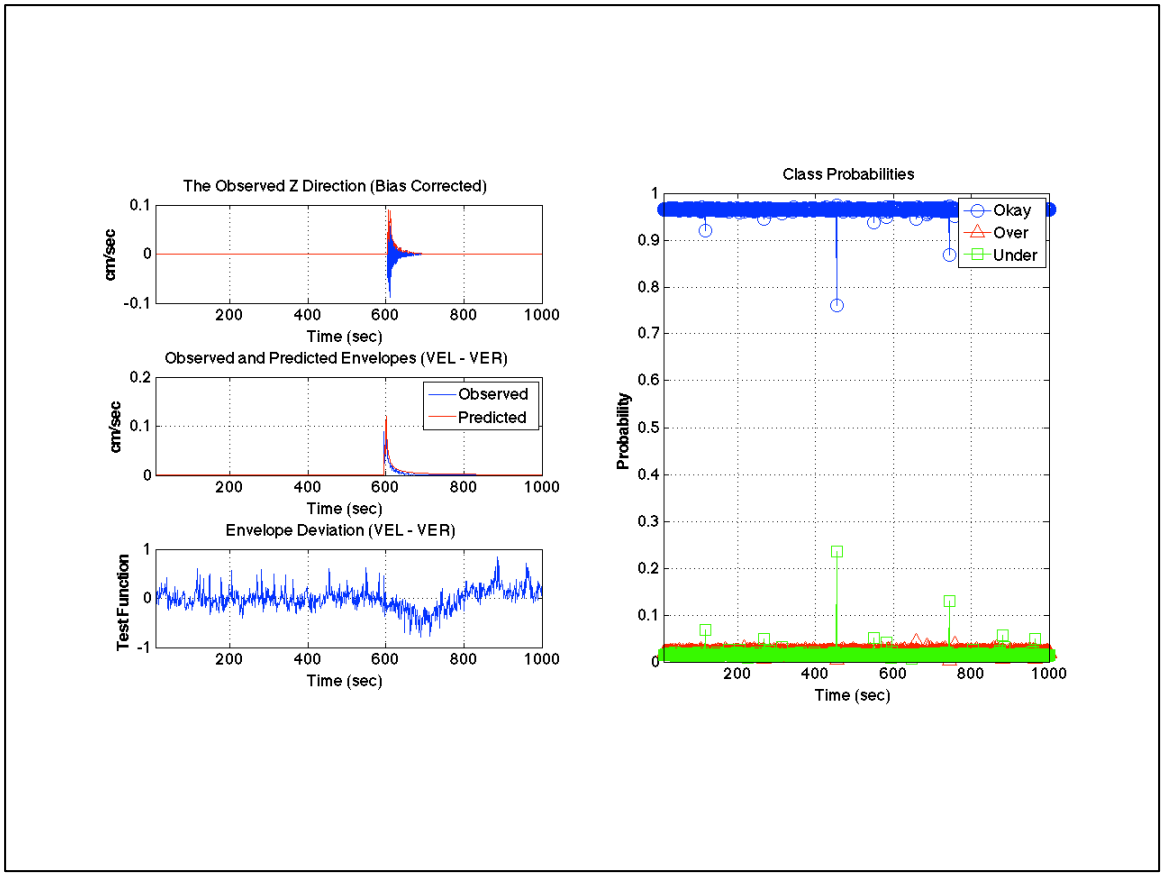

Figure 4.11: Reality Check Algorithm's performance plot for the okay-prediction example 2 computed using (3.100) from velocity input using Method II. For description of the figure, see *"General figure description"* given above.

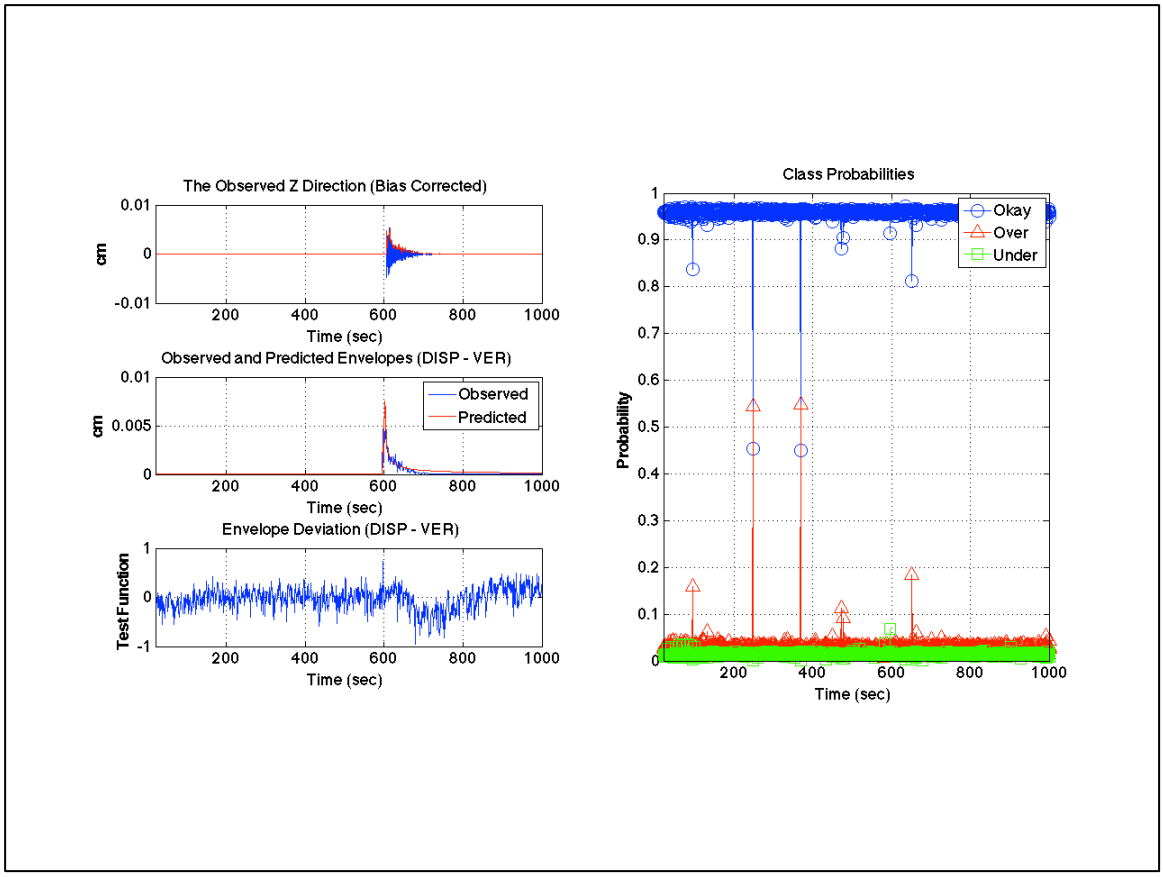

Figure 4.12: Reality Check Algorithm's performance plot for the okay-prediction example 2 computed using (3.101) from displacement input using Method II. For description of the figure, see *"General figure description"* given above.

Similar to the first okay-prediction example, displacement results are the most problematic ones, regardless of the method we used. Unlike the first okay-prediction example, however, unacceptable performance of displacement is observed before the earthquake waves reach the seismic station; in other words, the errors for the displacement results are in the part of the record where there is only supposed to be noise. Although the over-prediction probability values observed in the displacement results of the second method (see Figure 4.12) are comparable to the corresponding okay-prediction values, i.e., over-prediction probability at seconds 247 and 369 are approximately 0.545 and 0.548 respectively, while the corresponding okay-prediction values are approximately 0.454 and 0.451, these results alone might be misleading if acceleration and velocity results are not taken into consideration. On the positive side, this situation proves that the algorithm is very sensitive to the changes between observed and predicted values, as shown in the zoomed-in versions of Figure 4.12 (see Figures 4.13 and 4.14), that is to say, even when the noise levels at a station change more than enough to classify departure from normality, RCA is able to detect that!

The discrepancies observed in the displacement results are not the only ones as far as the second okay-prediction example is concerned. Figure 4.7, which is based on acceleration input, shows an under-prediction at 455th second of the record. The under-prediction probability does not exceed that of the okay- and over- predictions in either velocity or displacement results for both Method I and Method II. In fact, the acceleration results for Method II (see Figure 4.10) indicate accurate prediction. This under-prediction signal is due to a slight increase in the observed noise level as can be seen in the zoomed-in version of Figure 4.7 (see Figure 4.15). If the acceleration result of the first method

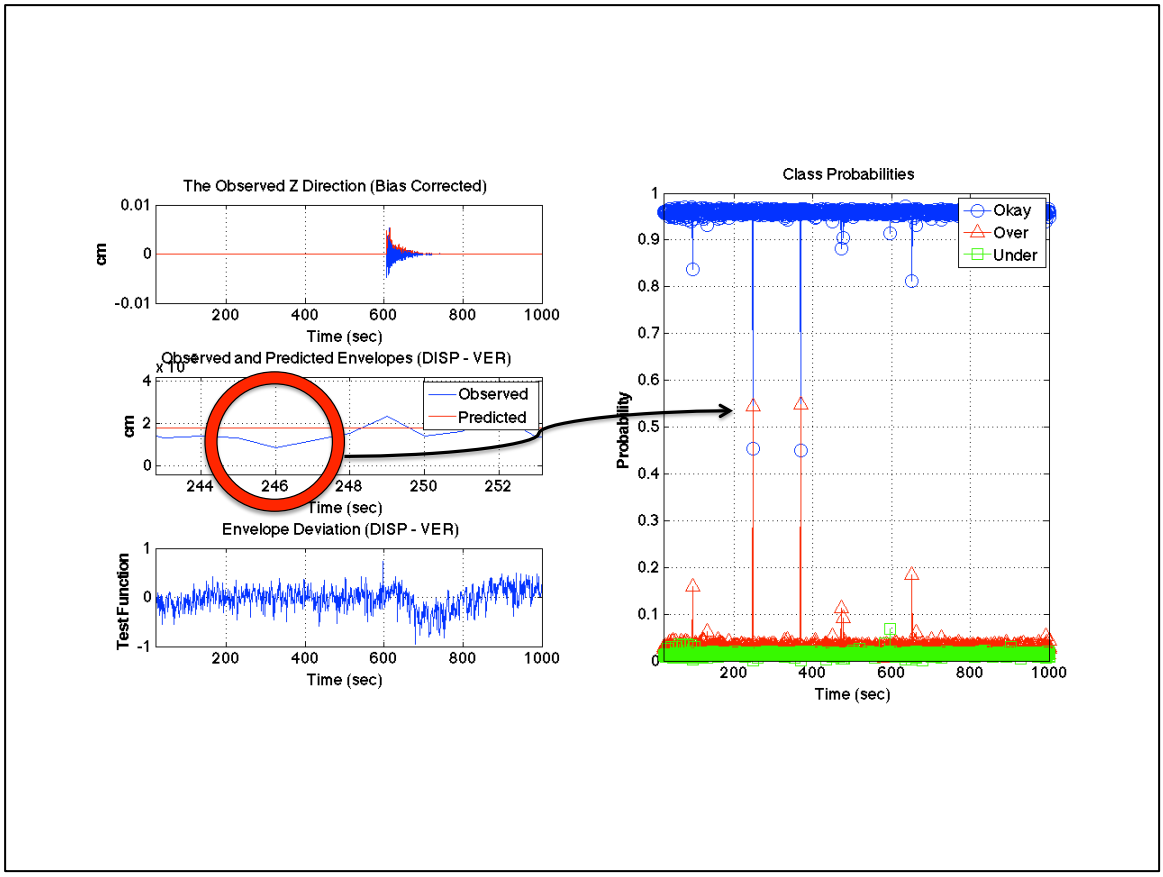

Figure 4.13: Reality Check Algorithm's performance plot for the okay-prediction example 2 computed using (3.101) from displacement input using Method II (zoomed-in on the first over-prediction indication). For description of the figure, see *"General figure description"* given above.

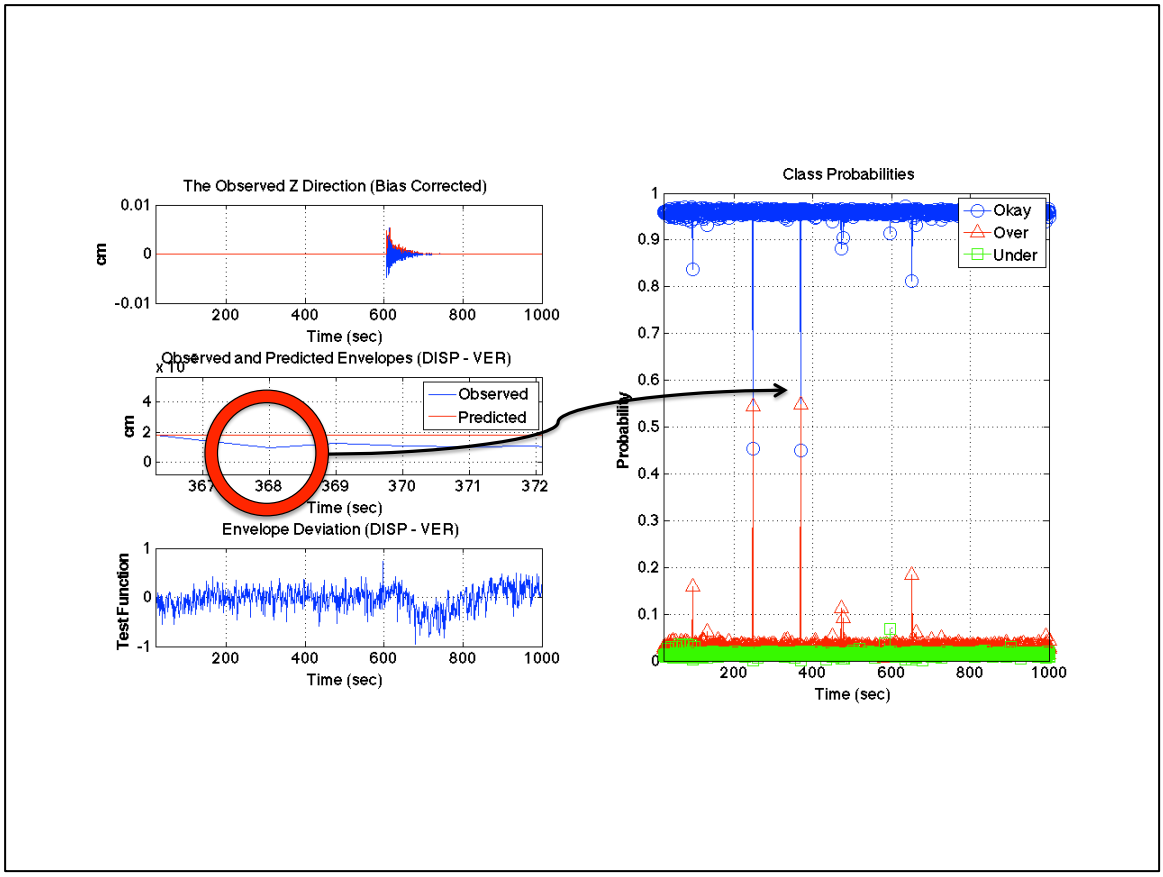

Figure 4.14: Reality Check Algorithm's performance plot for the okay-prediction example 2 computed using (3.101) from displacement input using Method II (zoomed-in on the second over-prediction indication). For description of the figure, see *"General figure description"* given above.

were the only measure we had, we would make a misclassification. Fortunately, we have velocity and displacement results of the first method contradicting that of acceleration. In addition to that, even the acceleration results of the second method (Figure 4.10) provide accurate classification as we mentioned above. The change in the noise levels could be so high that all measures may indicate an under-prediction (or over-prediction). For that reason, we suggest another measure of misfit, which will guarantee better performance for the predictions of RCA, in the upcoming sections of this chapter.

4.1.5 Okay-prediction example 1 with arrival time perturbations

Next, we investigate what happens to the RCA performance if the arrival times of P-waves are slightly miscalculated. This form of miscalculation could be due to location and/or origin time errors. First, we consider the case where the predicted P-wave arrives one second earlier than the observed one. Then, we look at the case where the predicted P-wave is one second late. Besides the perturbations mentioned above, the examples of this section are identical to the ones presented for okay-prediction example 1.

The purpose of these arrival time perturbations is to point out the need for another measure of misfit. This measure, however, will not be enough to make accurate classifications by itself; but the new and the already presented RCA will ‘complement’ each other’s weaknesses.

From this point on, we only show the figures for Method II, i.e., the results we computed using the models with the SBL technique.

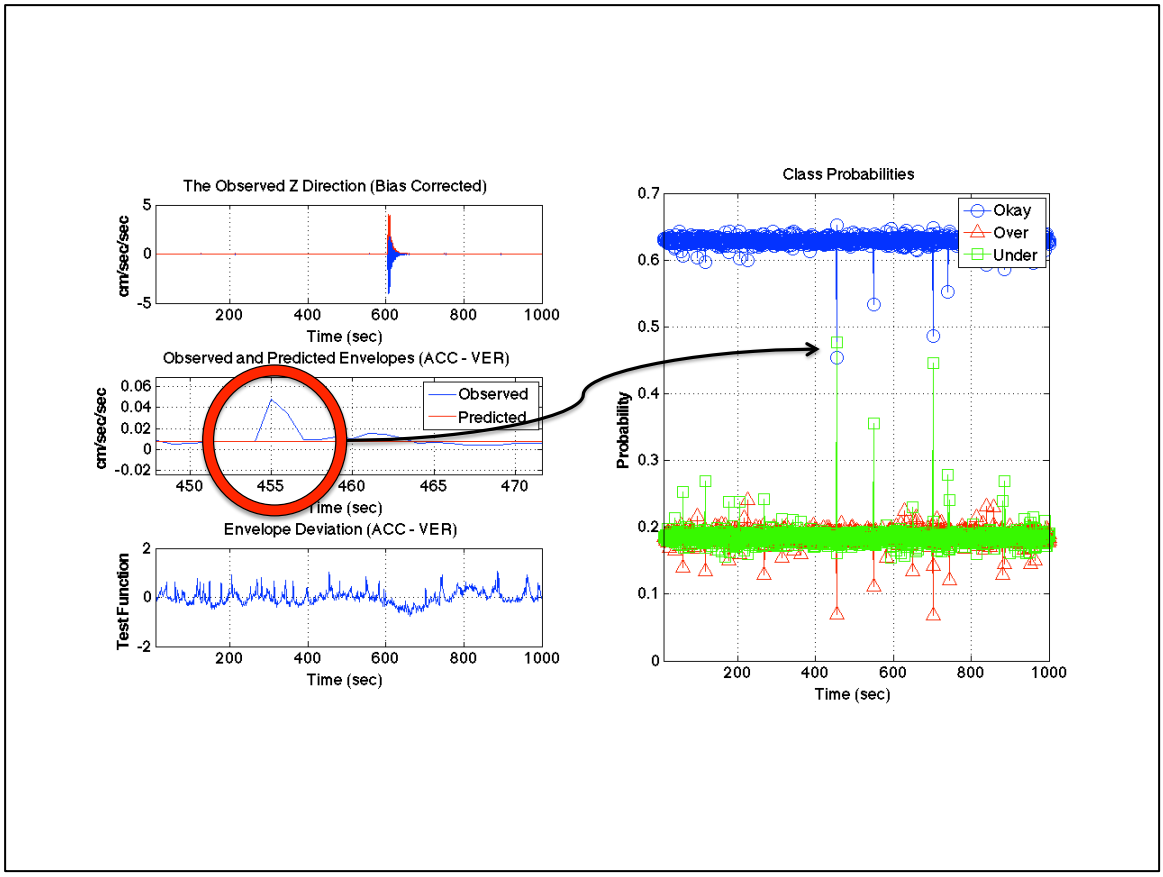

Figure 4.15: Reality Check Algorithm's performance plot for the okay-prediction example 2 computed using (3.68) from acceleration input using Method I. For description of the figure, see *"General figure description"* given above.

Let us start with the case where the predicted P-wave arrives one second earlier. As it can be seen in Figures 4.16 to 4.18 (in addition to Figure 4.19 which is the zoomed-in and highlighted version of Figure 4.17), RCA indicates an over-prediction. This is due to the fact that at the time of over-prediction indication, the observed ground motion envelope value is just the noise level at the station, whereas the predicted envelope counterpart is the P-wave value, which is significantly larger than the observed value. This perturbation alone can make RCA indicate an over-prediction as if an entire earthquake is assumed to exist but the assumption was wrong, that is, RCA, in this scenario, indicates an over-prediction as it would in the case of a false alarm. This is not acceptable because looking at the 'overall' fit tells us that the event is quite accurately predicted except for the arrival time. Therefore, we need a measure of the 'overall' fit in addition to RCA. The keyword is 'overall'!

Note that these figures also show that displacement alone does not indicate an over-prediction even though the probability of over-prediction increases significantly.

Next, we make the predicted P-wave arrive one second later than the observed one. In this scenario, while the observed envelope value at the time of P-wave arrival is significantly larger than the noise level, the corresponding predicted envelope value is still the noise! It is an example of under-prediction where the observed value is much higher than the predicted value. Figures 4.20 to 4.23 show that displacement is the only ground motion that indicates under-prediction for Method II. In fact, even though the corresponding acceleration and velocity under-prediction probabilities increase noticeably, the governing class of okay-, over-, and under-prediction is still the okay-prediction one.

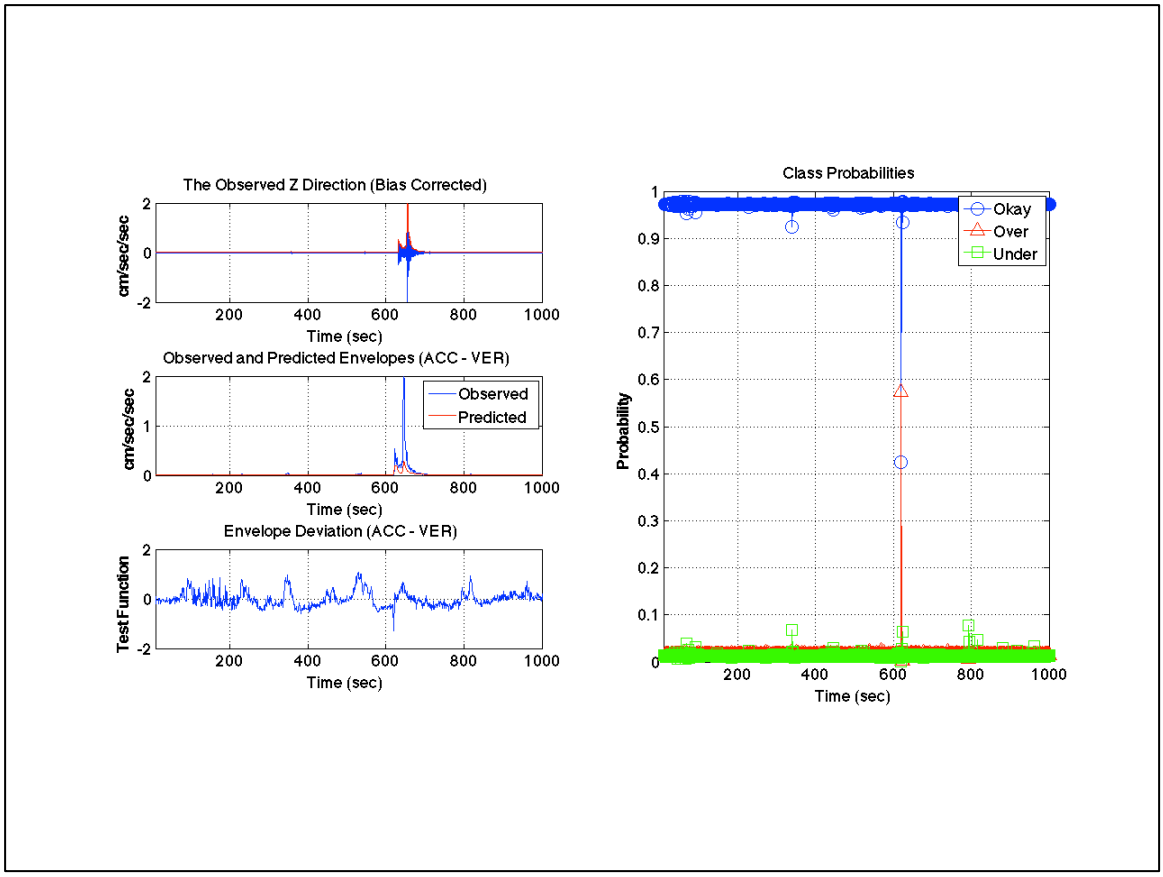

Figure 4.16: Reality Check Algorithm's performance plot for the okay-prediction example 1 (with early predicted P-wave arrival) computed using (3.99) from acceleration input using Method II. For description of the figure, see *"General figure description"* given above.

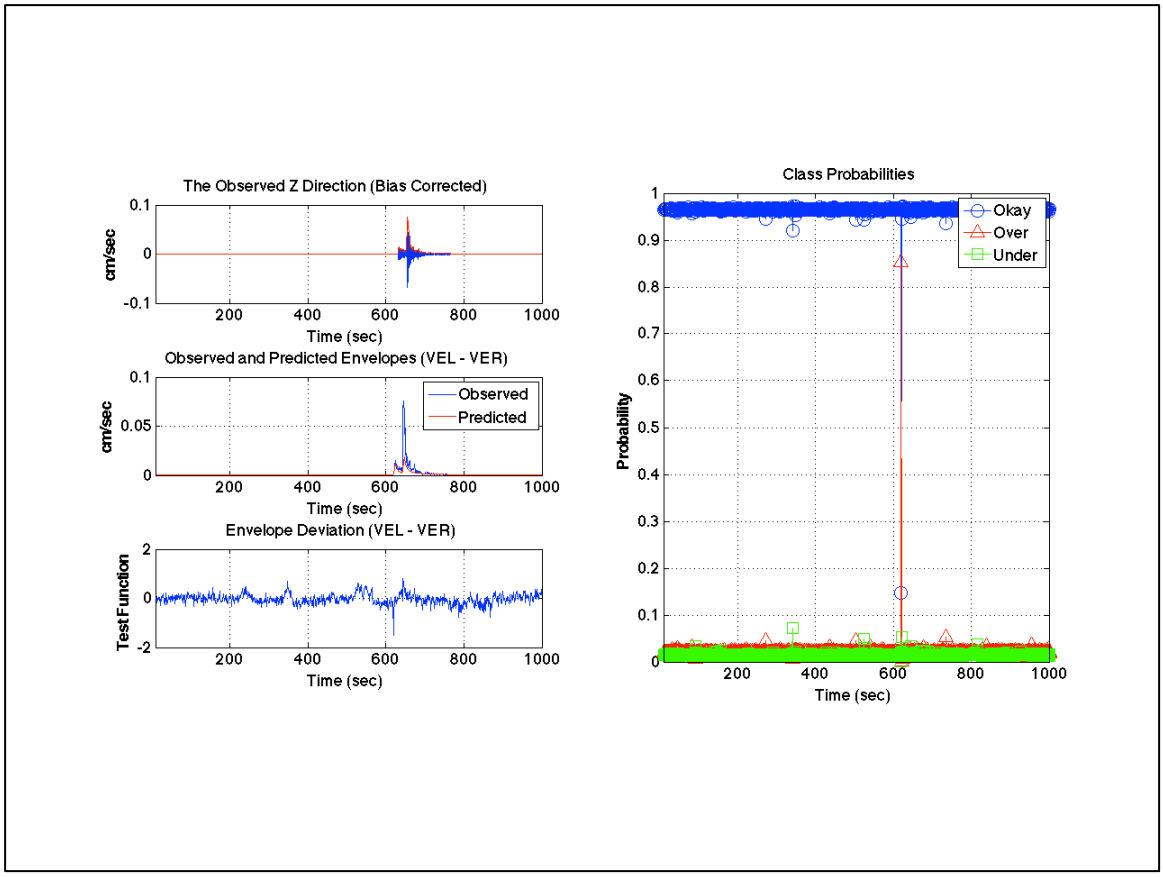

Figure 4.17: Reality Check Algorithm's performance plot for the okay-prediction example 1 (with early predicted P-wave arrival) computed using (3.100) from velocity input using Method II. For description of the figure, see *"General figure description"* given above.

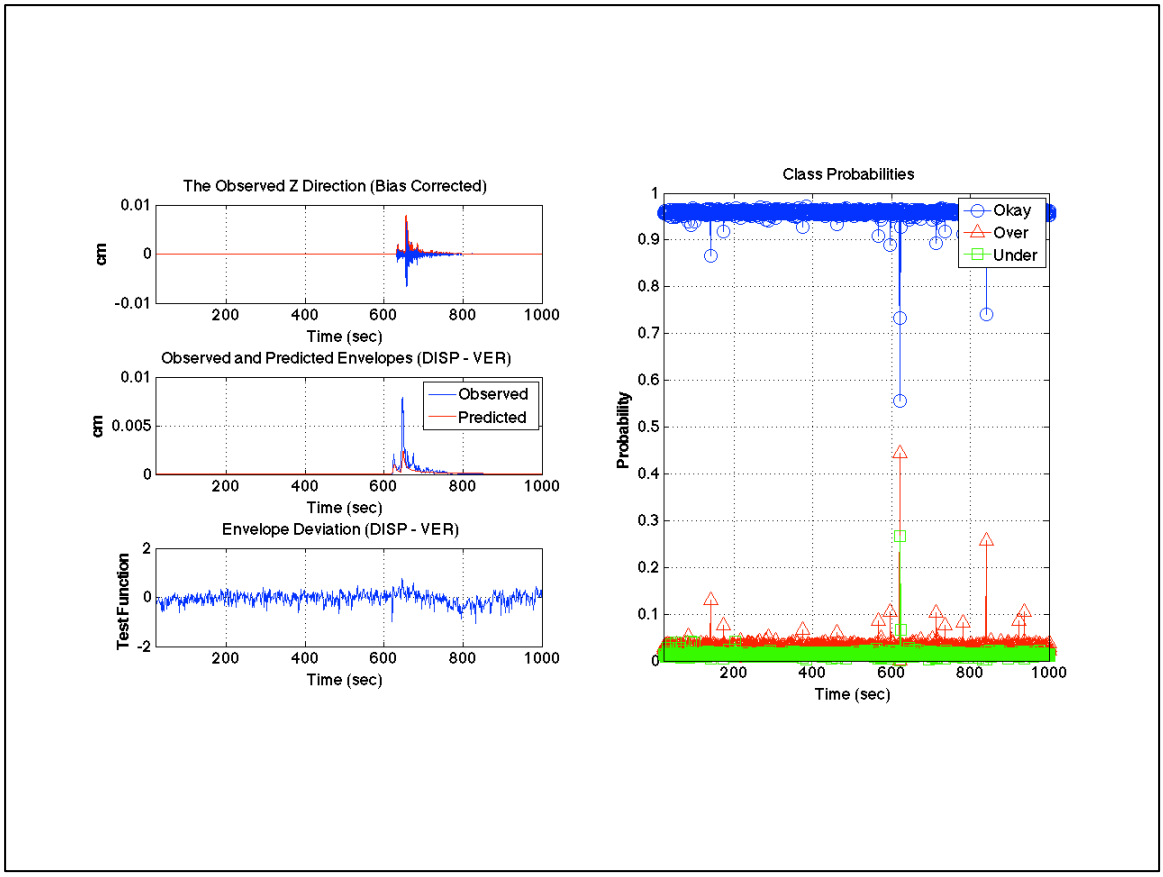

Figure 4.18: Reality Check Algorithm's performance plot for the okay-prediction example 1 (with early predicted P-wave arrival) computed using (3.101) from displacement input using Method II. For description of the figure, see *"General figure description"* given above.

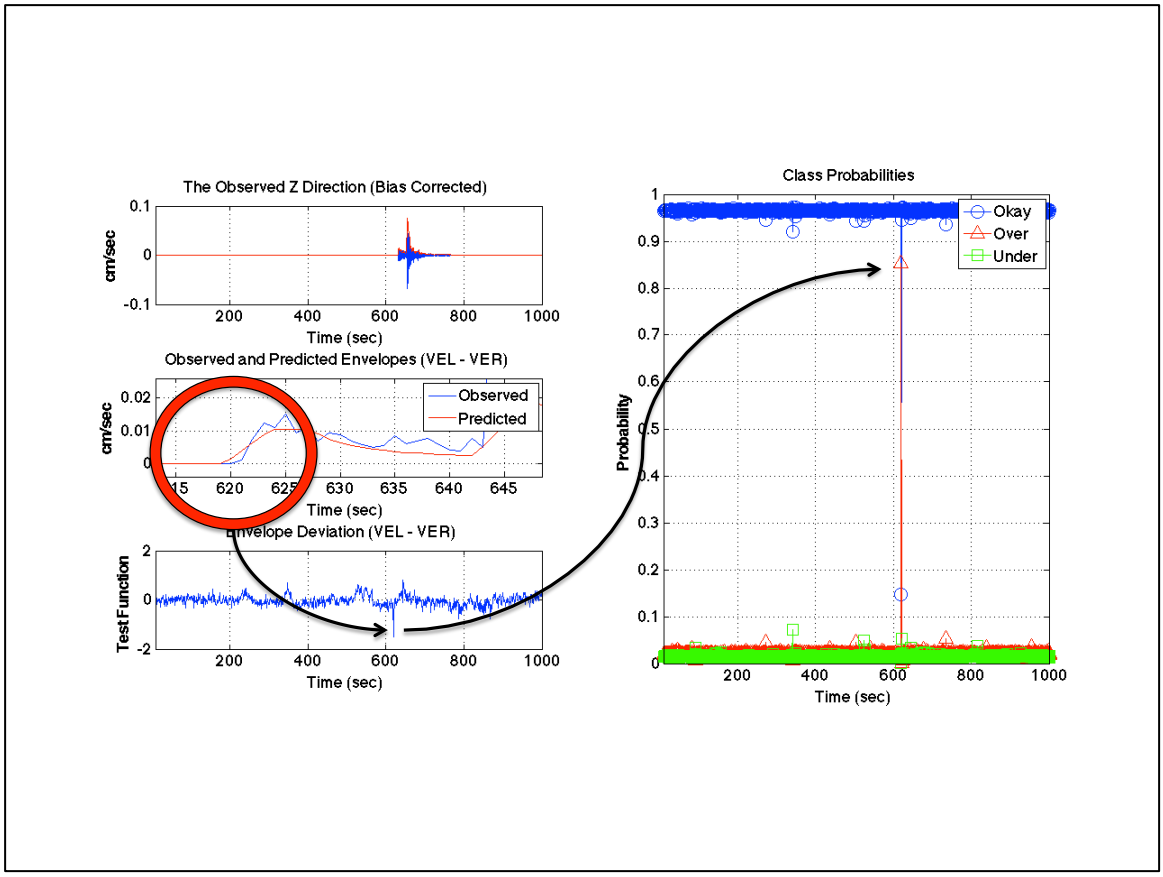

Figure 4.19: Reality Check Algorithm's performance plot for the okay-prediction example 1 (with early predicted P-wave arrival) computed using (3.100) from velocity input using Method II (zoomed-in on the early predicted P-wave arrival). For description of the figure, see *"General figure description"* given above.

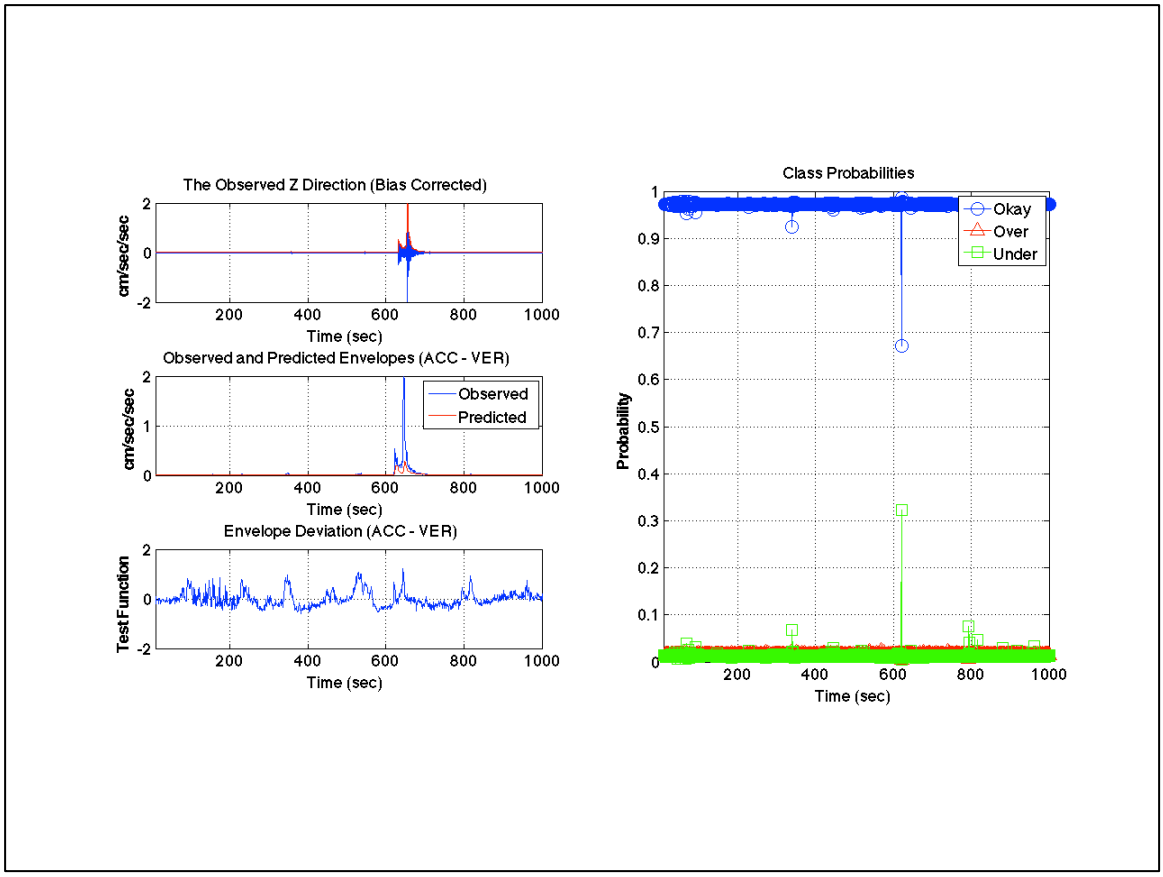

Figure 4.20: Reality Check Algorithm's performance plot for the okay-prediction example 1 (with late predicted P-wave arrival) computed using (3.99) from acceleration input using Method II. For description of the figure, see *"General figure description"* given above.

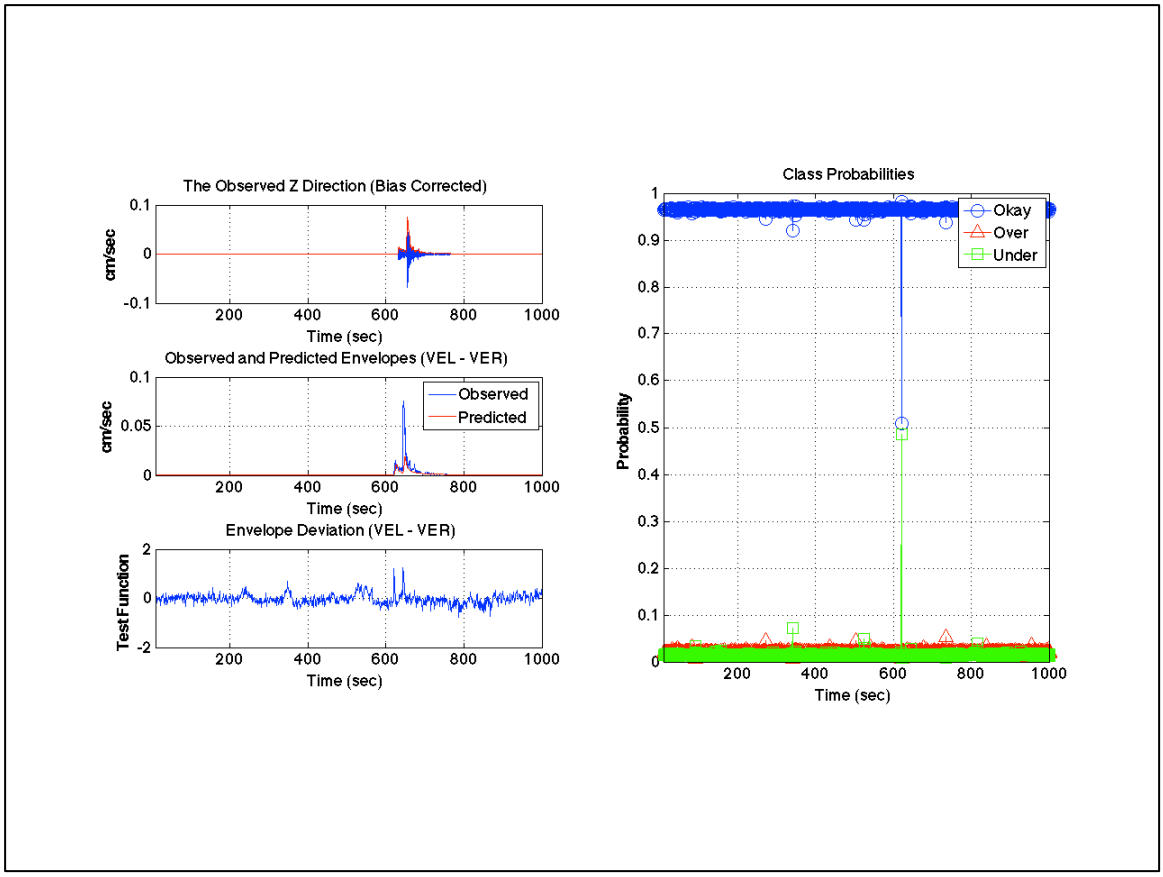

Figure 4.21: Reality Check Algorithm's performance plot for the okay-prediction example 1 (with late predicted P-wave arrival) computed using (3.100) from velocity input using Method II. For description of the figure, see *"General figure description"* given above.

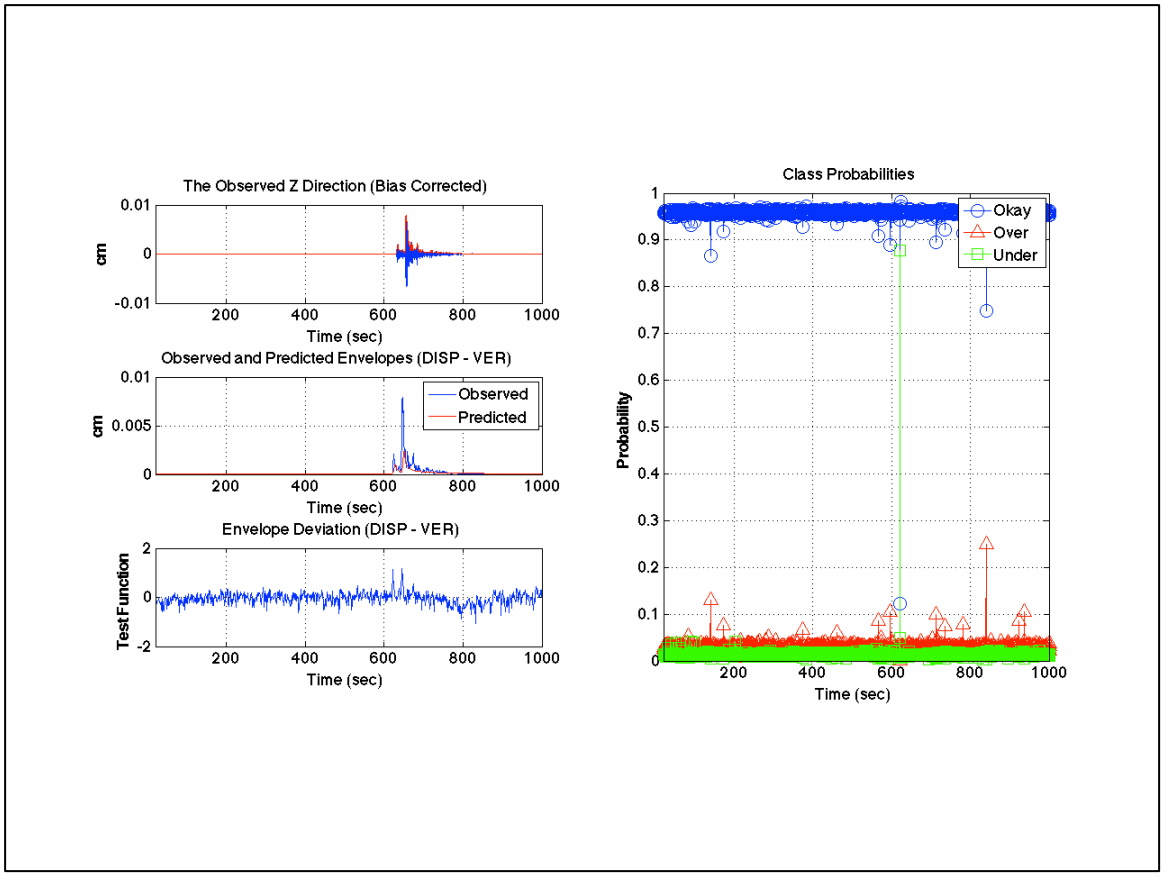

Figure 4.22: Reality Check Algorithm's performance plot for the okay-prediction example 1 (with late predicted P-wave arrival) computed using (3.101) from displacement input using Method II. For description of the figure, see *"General figure description"* given above.

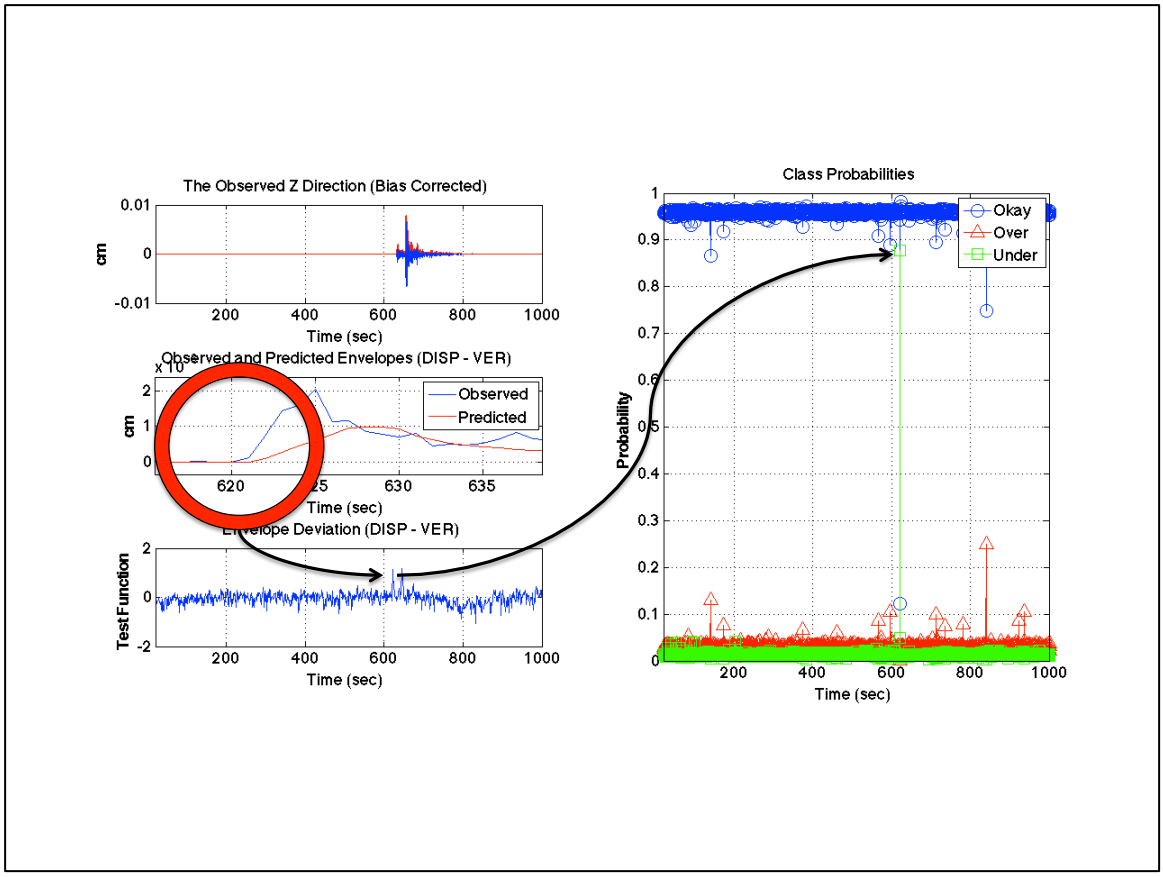

Figure 4.23: Reality Check Algorithm's performance plot for the okay-prediction example 1 (with late predicted P-wave arrival) computed using (3.101) from displacement input using Method II (zoomed-in on the late predicted P-wave arrival). For description of the figure, see *"General figure description"* given above.

There is an interesting lesson we can draw from these perturbation examples: while displacement is very sensitive to under-prediction cases and not so sensitive to over-prediction cases, the situation is reversed for acceleration and velocity, that is, acceleration and velocity are more sensitive to over-prediction cases than they are to under-prediction cases! Nevertheless, it is observed that a slight perturbation in the predicted P-wave arrival may cause an under-prediction classification by even Method II (in the case of displacement), as if the system completely missed an earthquake. Notice that, similar to the scenario in which the predicted P-wave arrives earlier, the 'overall' fit between the observed and predicted envelopes is somewhat acceptable. In light of these results for perturbations, it is even more clear that we should use a measure of 'overall' misfit, in addition to the current RCA computations.

Before we explain the new method, we would like to draw attention to an interesting difference between over- and under-prediction cases: when we made the predicted envelope arrive earlier (later), we did not just make the P-wave arrive earlier (later), we actually made the entire predicted envelopes, i.e., P- and S-wave envelopes combined with the noise envelope arrive earlier (later). Therefore, not only the P-wave arrival times between observed and predicted envelopes are different, but also the S-wave arrival times are different from each other. S-wave arrival time mismatch in the late arrival case can be observed in Figures 4.20 to 4.23 as the impulse-like pattern seen at the expected arrival time of the S-waves. However, the mismatch between the S-waves of observed and predicted envelopes in the case of early arriving predicted envelopes is not observed in Figures 4.16 to 4.19! The explanation is simple: the amplitude of the observed S-wave is larger than that of the predicted one. In the case of the early arrival,

the two values that are used to compute the test function, i.e., observed P-wave coda amplitude and predicted S-wave amplitude are not too different from each other. But in the case of late arrival, the two values that are used to compute the test function, i.e., observed S-wave amplitude and predicted P-wave coda amplitude are significantly different. This is the reason that we can observe the second impulse-like signal in the late arrival case.

4.2 A Supplementary Method for the Reality Check Algorithm

In the previous section, we showed the need for a measure of misfit that would supplement the RCA results. The new method will be used as a guaranteeing agent for the RCA messages generated for earthquake early warning systems.

The first thing that comes to mind is to use the RCA results on the test function computations after a prediction is made! Because a test function after a prediction is made can be considered as a measure of 'whatever the earth gives us minus whatever the system predicts' which is supposed to be zero in the ideal case, any value that is too nonzero should be an indication of something wrong. Also, because we have a classification scheme for such a measure, i.e., test function, we could try using the same classification on it. However, RCA did not perform well on the test functions after a prediction is made. This is due to the fact that oscillations among test function values after a prediction is made are so large that RCA is not able to detect outliers. For that reason, we need a different measure of misfit.

We have showed above that small perturbations in wave arrival times may cause RCA to indicate wrong predictions even though the 'overall' form of the event is accurately predicted. Therefore, we need a measure that checks the 'overall' fit between

observed and predicted envelopes. Thus, if there is any isolated impulse-like test function values, as there would be in case of small arrival time perturbations, the new measure should ignore them. This can be achieved by considering the current time test function value in relation to the previous test function values. Although we somehow already do that with the help of kurtosis and skewness, we also need make sure that an impulse-like test function value does not have too much impact on the results. As a result of the previous considerations for the new measure, we propose to use a running mean of the test function instead of the test function values directly. Because a running mean can be considered as a low-pass filter, by doing this, we make sure that the effects of an isolated impulse-like test function value are diminished. Of course, like with any running mean, we need to decide the length of the window that will slide across the continuous computations. For the time being, let us call this window length "running-mean-window-length". A specific value will be given later.

$$\text{Test Function With Running Mean}\left(N\right) = \frac{\sum_{n=N-M+1}^{N} \text{Test Function}\left(n\right)}{M} \tag{4.1}$$

where M is the running-mean-window-length. Remembering that ϕ denotes the test function defined in Chapter 2, let ϕ_r denote the test function with the running mean. Then, (4.1) can be written as

$$\phi_r\left(N\Delta t\right) = \frac{\sum_{n=N-M+1}^{N} \phi\left(n\Delta t\right)}{M} \tag{4.2}$$

So far, we have a way to make the current test function value be related to the previous test function values without using kurtosis and/or skewness. The next step is to

make the new measure detect irregularities, and more importantly, ignore the irregularity if it is due to a small wave-arrival-time perturbation. Since having a running mean through the test function after making a prediction smoothens the test function, traditional kurtosis and/or skewness computations did not perform well. As a result, a new method of computing moments of signal data is proposed based on classical mechanics, or to be more accurate, statics that are used in Civil Engineering. This new method of computing moments of time series data is the backbone of our supplementary technique to the RCA, and it is explained below. We call our method "Karakus-Heaton Moment of Signal Data".

4.2.1 Karakus-Heaton Moment of Signal Data

Let us start with defining our concept of moment. In statics, moment is defined as a quantity with a rotational direction, and a magnitude at a location. The direction of the moment depends on the convention one chooses; we choose the clockwise direction as positive and counter-clockwise direction as negative. The magnitude of a moment (in 2-D) at a location is the load amplitude times the perpendicular distance, i.e., moment arm between the application point of the load and the location at which the moment is computed (see Figure 4.24).

$$\text{Moment at the dashed line} = \frac{P_1 L_1 + P_2 L_2 - P_3 L_3 - P_4 L_4}{4} \tag{4.3}$$

Notice in (4.3) how we use the convention of clockwise and counterclockwise rotation: loads to the right of the dashed line in the figure tend to rotate the beam in the clockwise direction, so they have the positive sign, whereas the loads on the left side of the dashed line tend to rotate the beam in the counterclockwise direction, and therefore have the negative sign. We can think of a set of test function values for a given length of a window

as point loads on a beam that is supported right at the middle. In that case, the middle of the beam would correspond to the mid-location of the window. Amplitudes of the test function values would be the point load amplitudes, and the difference between the mid-location of the window, where the support is assumed to be, and the test function value location in time would be the moment arm. So, computing the moment at the mid-location would be nothing but multiplying the test function values by their moment arms, and then doing a summation using the convention given above.

Our original idea of computing the moments of signals was to find an alternative to kurtosis computations. For this reason, we make the Karakus-Heaton Moment of Signal Data computations resemble that of higher order statistics; in the following section we introduce the concepts of 'order of the moment' and 'normalizing factor'.

We define our 'order of the moment' as the power to which we raise the individual moments due to point loads (test function values), that is, while equation (4.3) is considered a first order moment calculation, an s^{th} order moment computation would be computed by the following:

$$s^{th} \text{ moment at the dashed line} = \frac{\left(P_1 L_1\right)^s + \left(P_2 L_2\right)^s + \left(-P_3 L_3\right)^s + \left(-P_4 L_4\right)^s}{4} \qquad (4.4)$$

In order to generalize the result of (4.4), let us think of computing the s^{th} order Karakus-Heaton Moment of Signal Data for a set of N test function values with running mean. Let L_n denote the location of test function value ϕ_{r_n} within N values such that $L_1 = 1, L_2 = 2, \cdots, L_N = N$. The s^{th} order Karakus-Heaton Moment of Signal Data is denoted KHM_s, and is defined by:

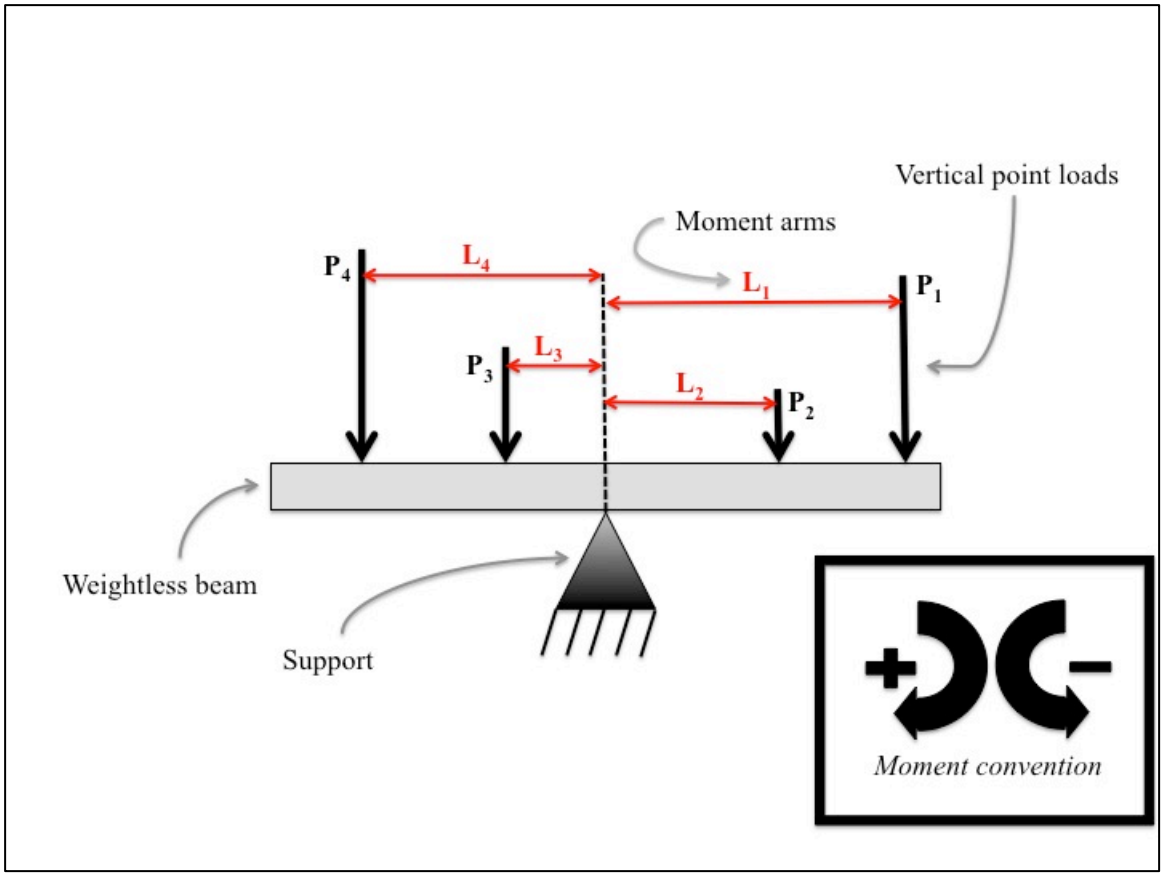

Figure 4.24: Static moment computation illustration in 2D. The purpose of this figure is only to clarify how the Karakus-Heaton Moment of Signal Data is computed and to present a physical representation of our concept of moment. Karakus-Heaton Moment of Signal Data is computed at the vertical section indicated by the dashed line using the formula in (4.3).

$$KHM_s = \frac{\left[\frac{\sum_{n=1}^{N}\left(\phi_{r_n}\left(L_n - \frac{N}{2}\right)\right)^s}{N}\right]}{\left[\left(KHM_2\right)^{\frac{s}{2}}\right]} \tag{4.5}$$

Notice that KHM_2 is used in the denominator raised to the appropriate power so that both the numerator and the denominator have the same units. This process is done to simulate the higher order statistic computations where the fourth and third moments of data about the mean value are divided by the variance of the data raised to such a power that resulting value is dimensionless (DeCarlo 1997). Therefore, we do not directly compute the s^{th} moment of the data, but we also normalize the computations by using an appropriate power of the 2^{nd} moment of the same data. The expression in the denominator of (4.5) is called the 'normalizing factor'.

The higher order Karakus-Heaton Moment of Signal Data, KHM_s, is devised to make exceptional (out of the ordinary) values in test function computations stand out. This idea can be best explained by looking at Figure 4.25. By using the logarithm, differences among such values are visually suppressed. In contrast, we aim to visually amplify the differences among test function values, when there are exceptional values after taking the mean. In this case, they will stand out in the KHM_s computations, provided they are not isolated. By isolated, we mean an impulse-like test function value due to small perturbations in arrival time. If we miss an entire event, there will be several

exceptional values, which will not go away after taking the mean. Then, the KHM_s computations will make these values easily detectable.

The following steps describe our new method:

1. First, we get rid of an isolated impulse-like value in the test function results with the help of a running mean.
2. Then, we apply a high-pass filter with the same parameters as used in Chapter 2. (Remember that we used that filter to get rid of long period trends because of coda mismatch between predicted and observed envelopes).
3. Then, we apply the KHM_s computation on the results of step 2.
4. Finally, we take the time derivative of the results of step 3 (similar to taking the time derivative of kurtosis and skewness results in Chapter 2).

Although one can argue that the optimum values of window lengths for the running mean, order, and window length of the KHM_s computation can benefit from Bayesian regression analysis, experience with several examples, including those presented in this chapter, suggests the following specific values for relatively good results:

$$\begin{aligned} \text{Window length of running mean} &= 40\text{sec} \\ \text{Order of KHM} &= 10 \\ \text{Window length of KHM} &= 80\text{sec} \end{aligned} \tag{4.6}$$

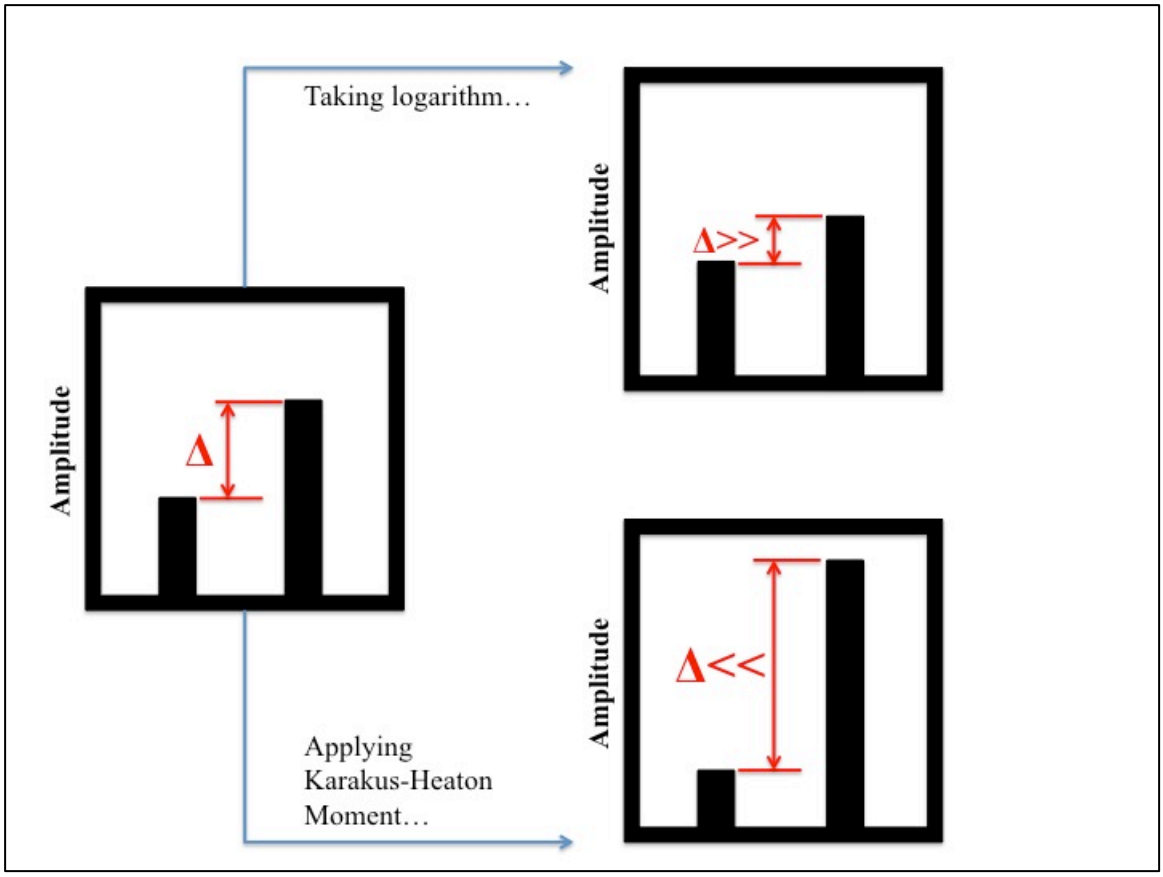

Figure 4.25: Comparison of taking logarithm with applying Karakus-Heaton Moment of Signal Data. Horizontal axis in the black squares shown above can be considered as time; then, the importance of using Karakus-Heaton Moment of Signal Data for real-time applications becomes more pronounced because an out-of-place value will stand out as soon as it is detected.

The KHM_s results, which can be seen in the second row on the right column of Figures 4.26 to 4.28, are virtually the same for all cases, i.e., okay-prediction and okay-prediction with perturbations. Because a running mean can also be considered as a low-pass filter, the test function looks free of isolated impulses (Figures 4.27 and 4.28) and much smoother than before, as it can be seen in the figures in the second row on the left. Because there are no further significant mismatches between observed and predicted values, KHM_s does not produce values that are significantly different from each other, that is, the 'overall' fit between observed and predicted envelopes is confirmed. Therefore, the new method we propose gives the ability to distinguish a small perturbation from a total missed event and a total false alarm.

When we visit the examples of over- and under-prediction in the next chapters, the differences among the KHM_s values will be much larger than those seen in Figures 4.26 – 4.28.

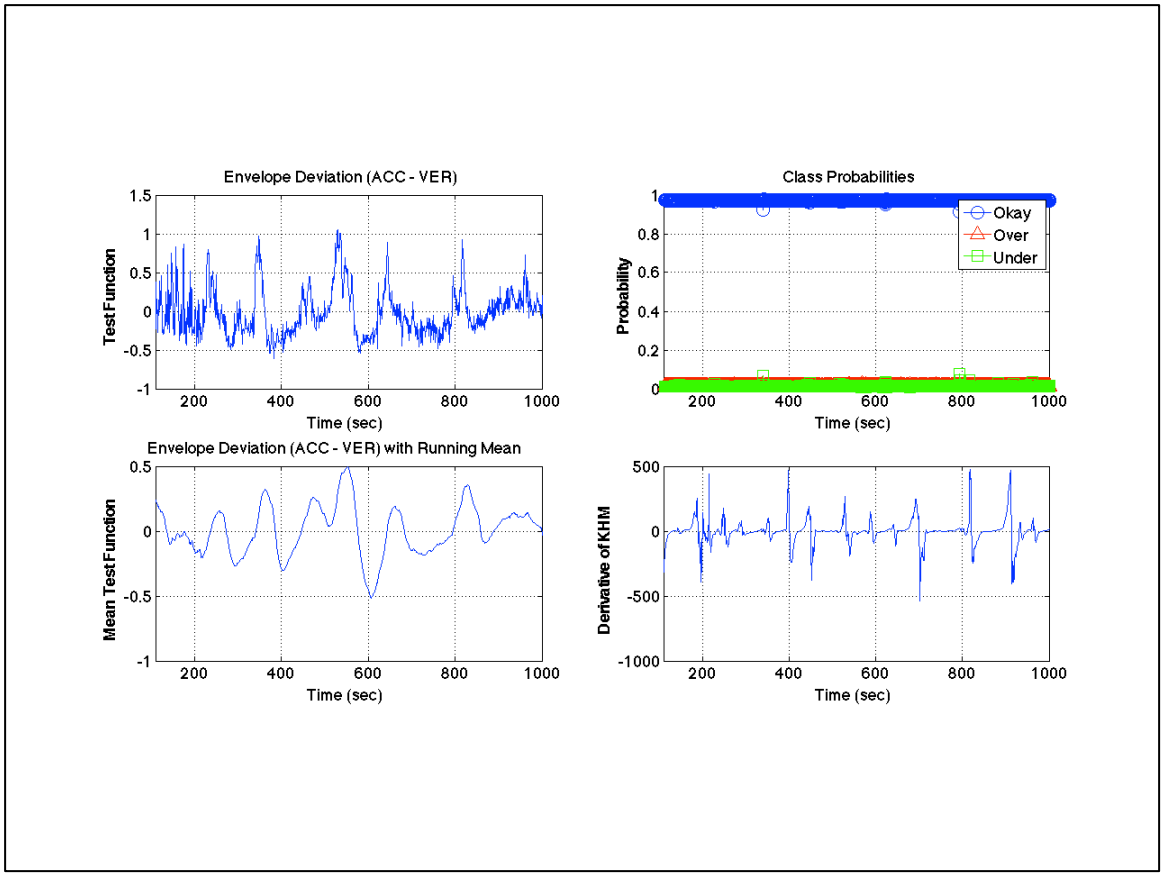

Figure 4.26: Performance of the supplementary method to RCA for the okay-prediction example 1. Left panel: first row shows the test function after high-pass filtering by the values given in Chapter 2; second row shows the test function values after applying a running mean and a high-pass filter by the values given in Chapter 2. Right panel: first row shows the probability values computed using (3.99) from acceleration input using Method II for each class with different color and marker; second row shows the results of the supplementary technique.

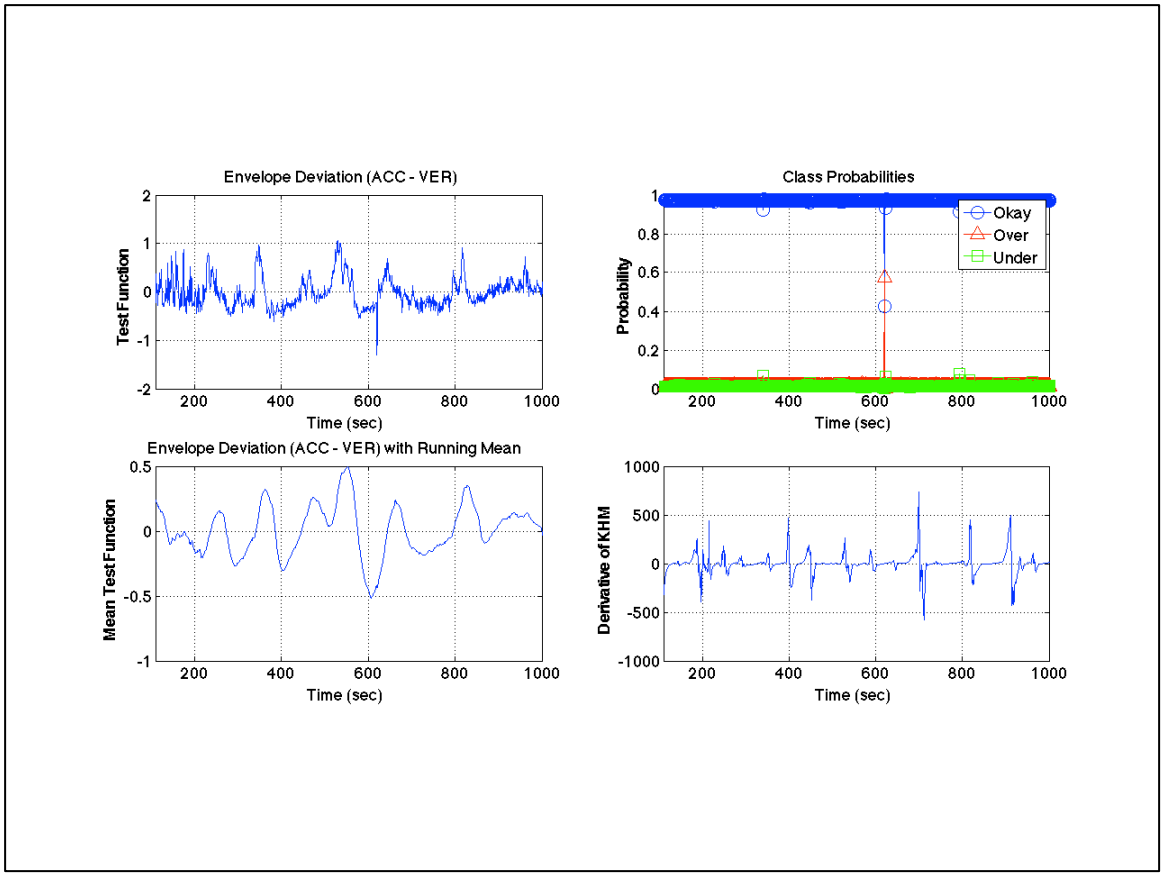

Figure 4.27: Performance of the supplementary method to RCA for the okay-prediction example 1 (with early predicted P-wave arrival). Left panel: first row shows the test function after high-pass filtering by the values given in Chapter 2; second row shows the test function values after applying a running mean and a high-pass filter by the values given in Chapter 2. Right panel: first row shows the probability values computed using (3.99) from acceleration input using Method II for each class with different color and marker; second row shows the results of the supplementary technique.

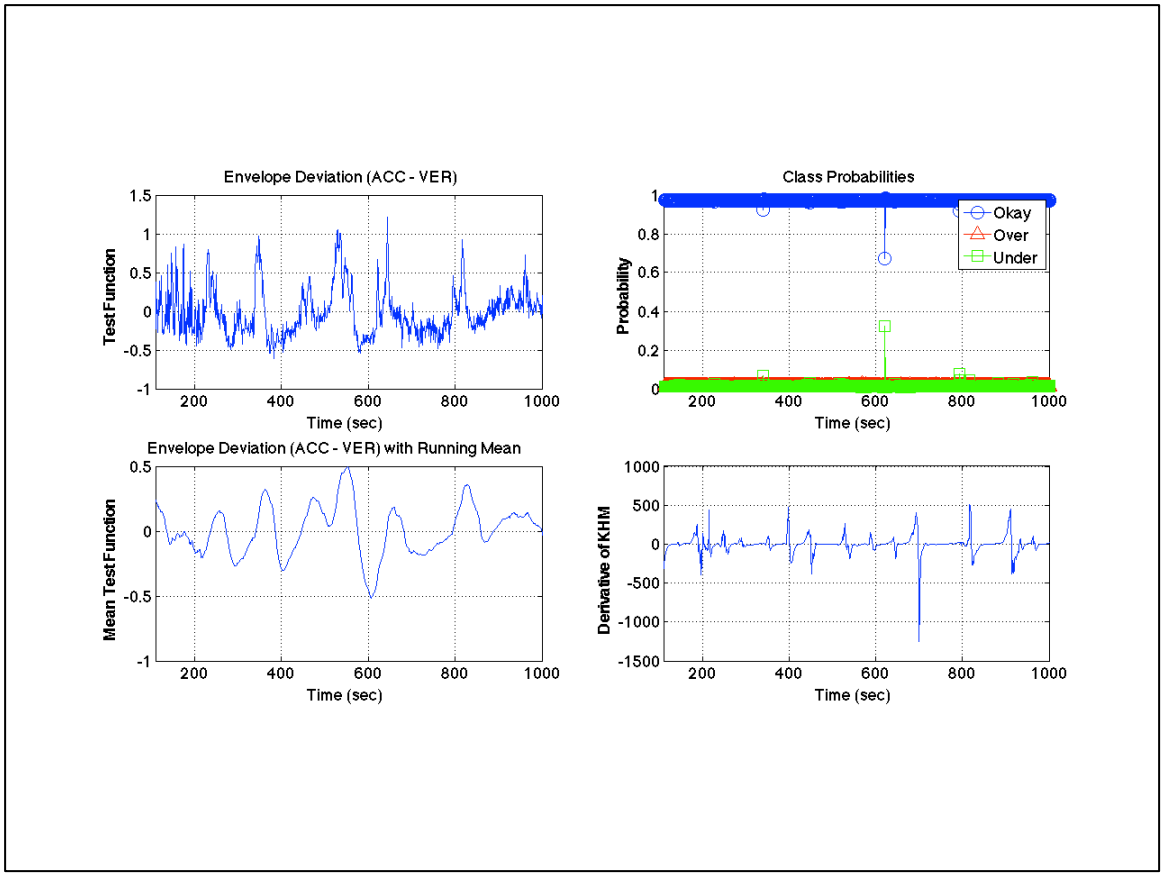

Figure 4.28: Performance of the supplementary method to RCA for the okay-prediction example 1 (with late predicted P-wave arrival). Left panel: first row shows the test function after high-pass filtering by the values given in Chapter 2; second row shows the test function values after applying a running mean and a high-pass filter by the values given in Chapter 2. Right panel: first row shows the probability values computed using (3.99) from acceleration input using Method II for each class with different color and marker; second row shows the results of the supplementary technique.

Chapter 5

Case Studies – Over Predictions

In this chapter, we will demonstrate the performances of both the Reality Check Algorithm (RCA) and the Karakus-Heaton Moment of Signal Data (KHM) using two over-prediction cases the Decision Module (DM) experienced in the past. Unlike the previous chapter, we only compute the class probabilities by using Method II (the Automatic Relevance Determination (ARD) Prior technique in Chapter 3) to demonstrate the RCA algorithm. Then, we use the technique introduced in Chapter 4, i.e., KHM (Karakus-Heaton Moment of Signal Data) to mitigate problems associated with the examples.

5.1 Over-prediction example 1

Our first example is an event that was experienced on July 1st, 2015 due to a calibration pulse. DM sent out an alert indicating there was a magnitude 8.2 event at latitude 34.753 and longitude -122.402, with an origin time of 2015/07/01, 00:55:06 UTC (see Figure 5.1). The particular stations that caused DM to send such an alert were experiencing calibration pulses at the same time. We demonstrate our system's performance for one of those stations:

Network: PG

Station: ARD

Type: Strong Motion Seismometer

False events:

ID	Alert UTC Time	Origin UTC Time	Mag	TpMag	PdMag	Lat	Lon
447393	2015/07/01 00:55:39	2015/07/01 00:55:06	8.2	7.8	8.2	34.753	-122.402
447394	2015/07/01 00:55:43	2015/07/01 00:55:30	6.3	7.7	6.3	35.517	-121.023

Channel triggers for the M8.2 event:

	sta	net	chn	time	tpmag	pdmag
1	ARD	PG	HNZ	00:55:31.005	7.844	8.271
2	CSD	PG	HNZ	00:55:31.010	7.842	8.282
3	LSD	PG	HNZ	00:55:32.025	7.840	8.158
4	WRD	PG	HNZ	00:55:33.025	7.549	7.928

Channel triggers for the M6.3 event:

	sta	net	chn	time	tpmag	pdmag
1	ARD	PG	HNZ	00:55:32.005	7.844	6.445
2	CSD	PG	HNZ	00:55:32.010	7.842	6.725
3	WRD	PG	HNZ	00:55:34.025	7.549	6.485
4	PBD	PG	HNZ	00:55:37.020	10.476	5.611

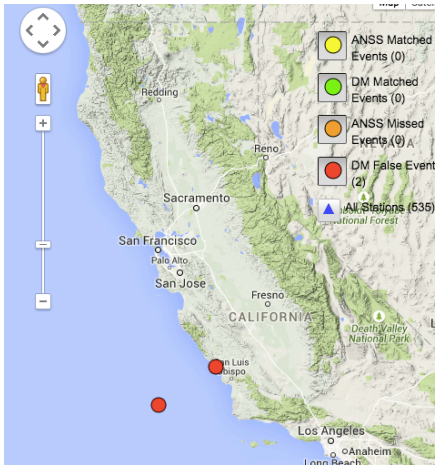

Figure 5.1: Summary of July 1st 2015 false alarms (Source: Prof. Richard Allen)

First, we clarify a point about the Virtual Seismologist (VS); it was designed to work for a maximum event magnitude of 6.5, and a maximum distance of 200 km (Cua 2005). We should start considering the finiteness of the fault if the event has a magnitude larger than 6.5. However, for the purpose of demonstrating performance of our methods, we use VS envelopes created using M6.5 (instead of M8.2, which is what DM predicted), and distance of 151.8315 km, which is the distance between the assumed epicenter and the location of the seismic station given above. Because the duration of the calibration pulse is so short, the choice of the site condition, which determines whether soil or rock coefficients are used to create the VS envelopes, does not make much difference; therefore, we choose 'soil' for simplicity. We show that even M6.5 prediction will result in an over-prediction classification, ultimately resulting in an indication of a false alarm; that is, even if DM predicted M6.5, it would have been classified as an over-prediction by our methods.

The trigger time of the calibration pulse at the station ARD in network PG is given as 00:55:31.005 UTC. We start by computing the arrival time of predicted waves at that particular seismic station by assuming a mean P-wave velocity of 6.5 km/sec and taking a distance between the DM epicenter and the stations of 151.8315 km; the result is approximately 24 seconds. Adding this result to the UTC origin time published by DM, which is 00:55:06, we get approximately 00:55:30. That means, according to DM, our predicted P-wave should have arrived at the particular station at 00:55:30 UTC, which is one second before the reported time of the trigger due to the calibration pulse. However, the choice of mean P-wave velocity of 6.5 km is somehow arbitrary, and in the next sections we make arrival times of observed and predicted envelopes match. Figures 5.2 to

5.4 show the performance of RCA, and Figure 5.5 shows the result of the supplementary technique introduced in Chapter 4. Recall that the supplementary technique aims to 'support' the indication by RCA.

Similar to Chapter 4, we provide two general descriptions for figures of this chapter.

General figure description - I:

Left panel: first row shows the seismogram recorded by the seismic station whose identification information is given above (seismogram in blue and its envelope in red); note that although only the vertical channel is shown in the plot for demonstration purposes, both vertical and horizontal channels were used in RCA computations; the second row shows both the predicted and observed envelopes; the third row shows the test function after high-pass filtering by the values given in Chapter 2. Right panel shows the probability values for each class with a different color and marker.

General figure description - II:

Left panel: first row shows the test function after high-pass filtering by the values given in Chapter 2; second row shows the test function values after applying a running mean and a high-pass filter by the values given in Chapter 2. Right panel: first row shows the probability values computed using (3.99) from acceleration input using Method II for each class with different color and marker; second row shows the results of the supplementary technique.

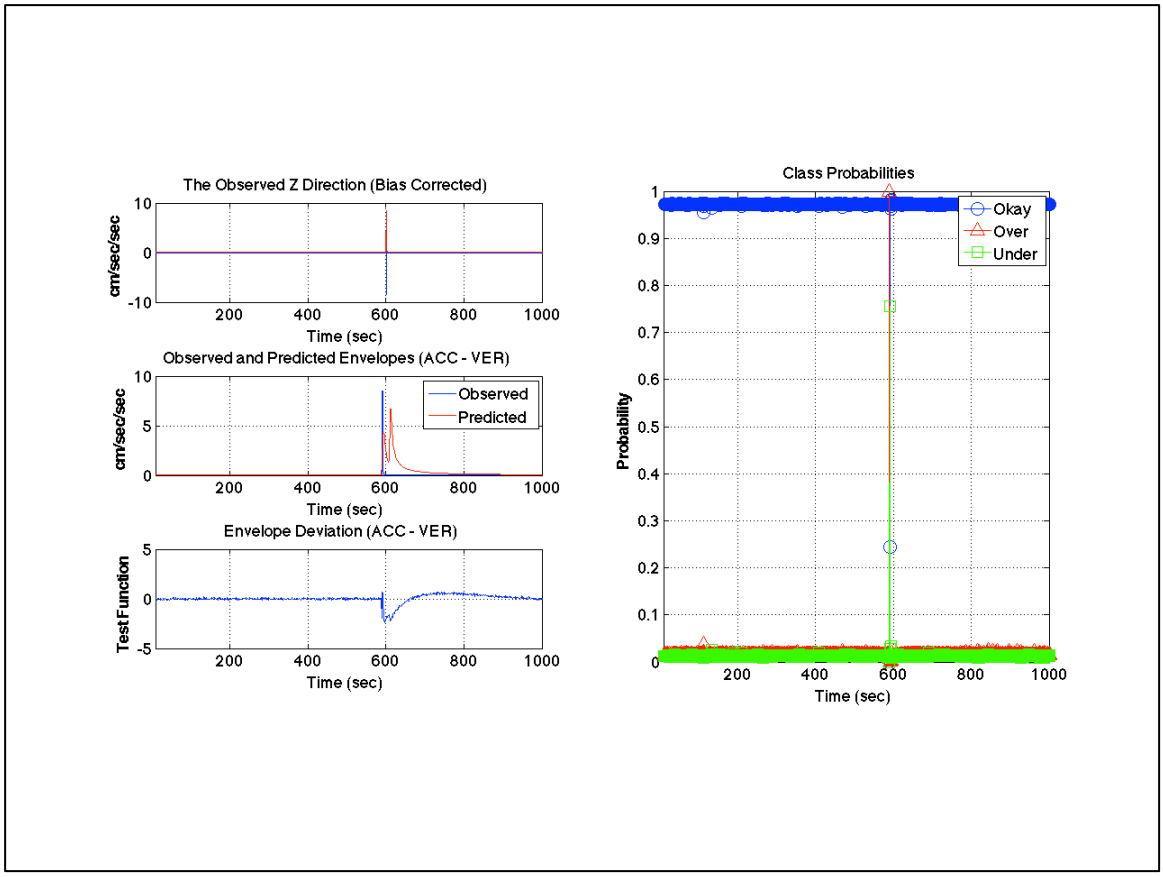

Figure 5.2: Reality Check Algorithm's performance plot for the over-prediction example 1 (with predicted wave arriving earlier than observed calibration pulse) computed using (3.99) from acceleration input using Method II. For description of the figure, see *"General figure description - I"* given above.

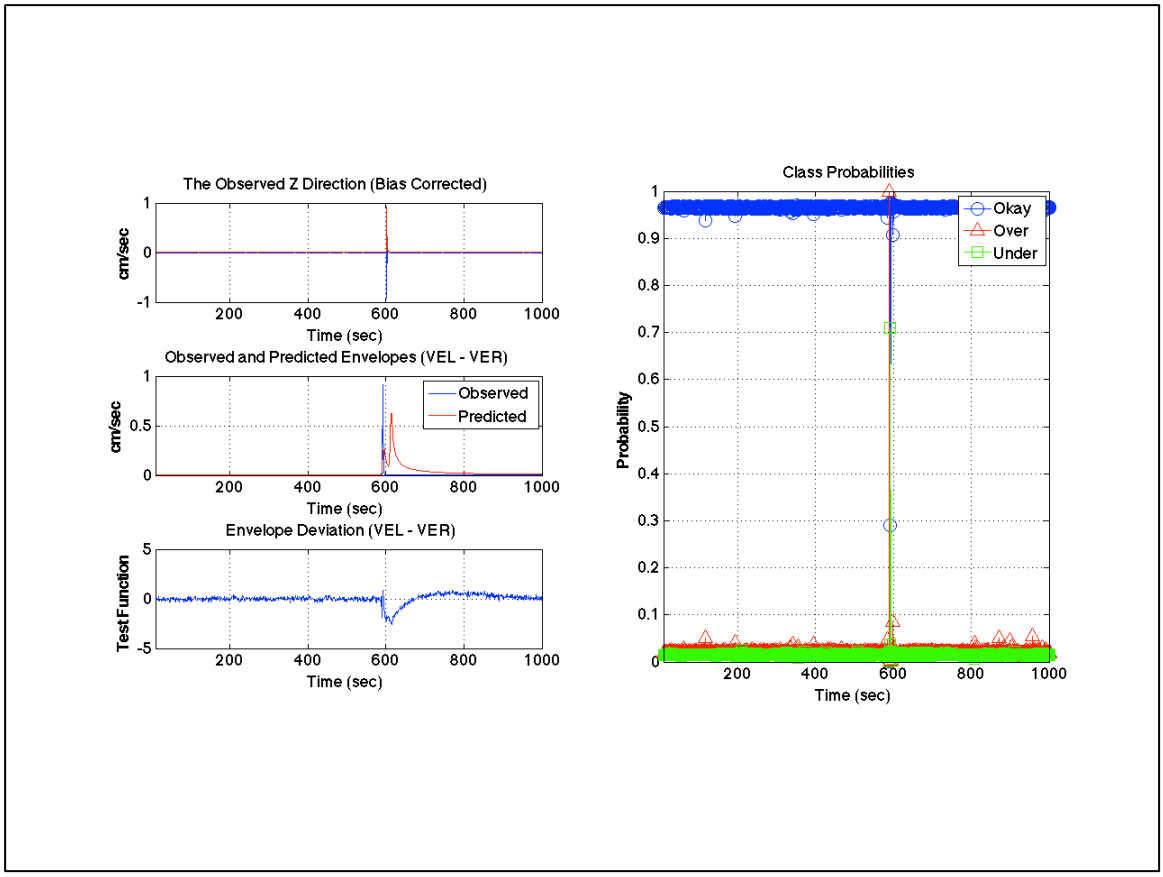

Figure 5.3: Reality Check Algorithm's performance plot for the over-prediction example 1 (with predicted wave arriving earlier than observed calibration pulse) computed using (3.100) from velocity input using Method II. For description of the figure, see *"General figure description - I"* given above.

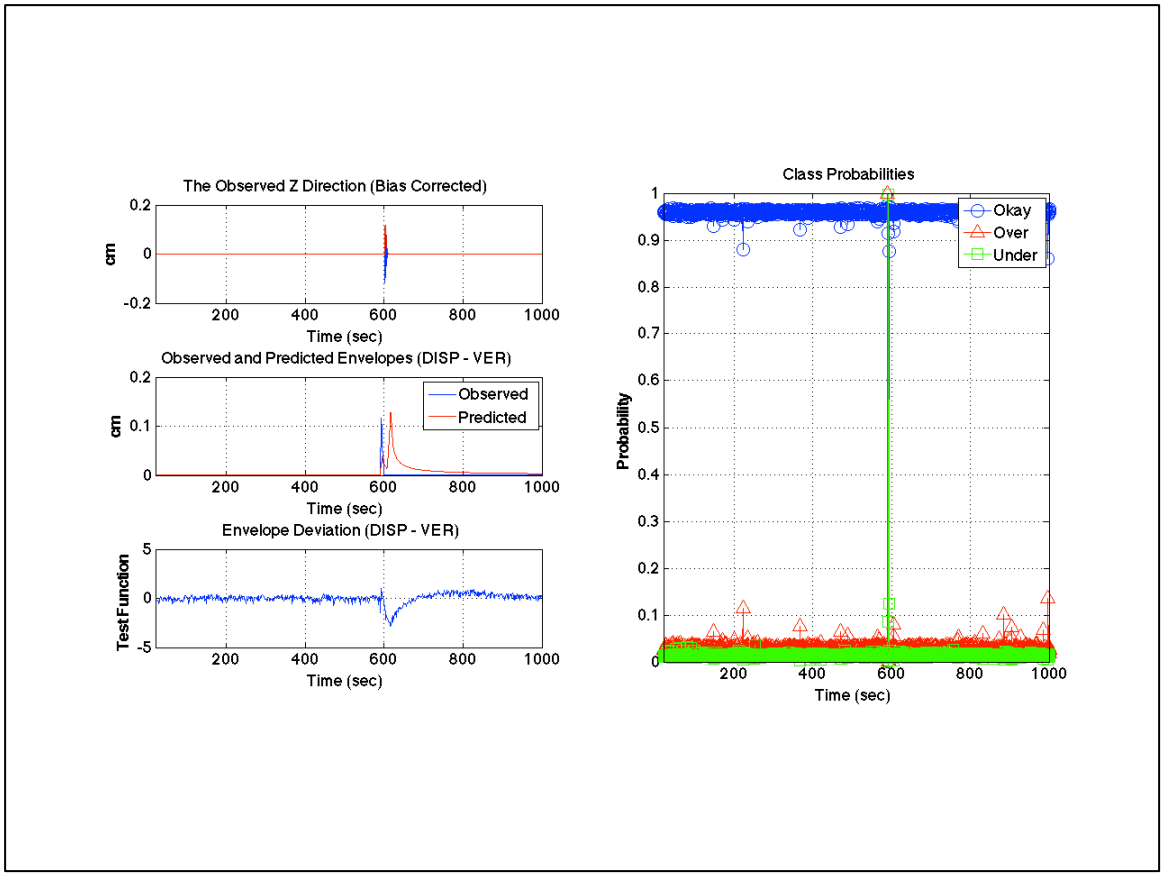

Figure 5.4: Reality Check Algorithm's performance plot for the over-prediction example 1 (with predicted wave arriving earlier than observed calibration pulse) computed using (3.101) from displacement input using Method II. For description of the figure, see *"General figure description - I"* given above.

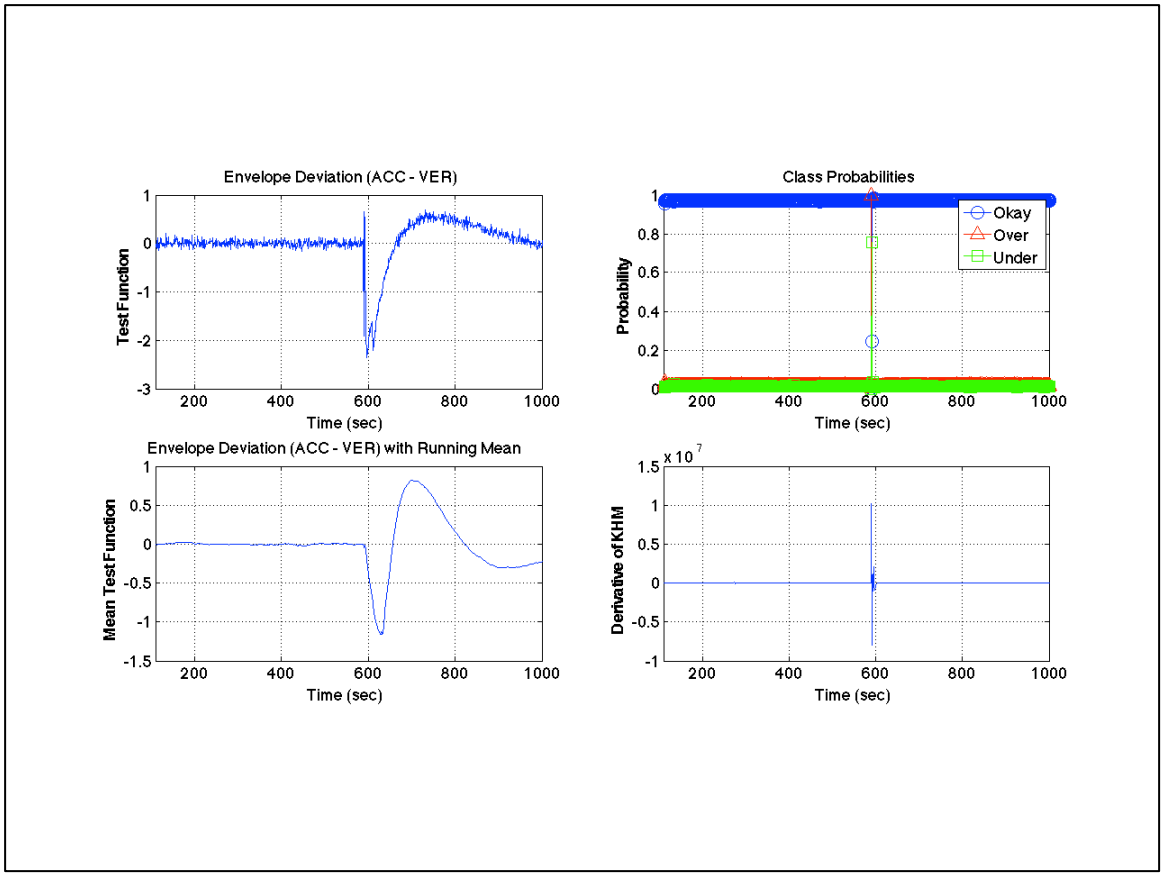

Figure 5.5: Performance of the supplementary method to RCA for the over-prediction example 1 (with predicted wave arriving earlier than observed calibration pulse). For description of the figure, see *"General figure description - II"* given above.

5.1.1 Discussion of over-prediction example 1

All of the Figures 5.2 to 5.4 indicate an over-prediction classification without exception. Over-prediction class indication is due to the fact that the observed envelope value, by the time the predicted envelope arrives, consists of only noise (see Figure 5.6). Recalling that same type of indication could be due to small arrival time perturbations, it is natural to ask if this is in fact an over-prediction case. The answer to that question is provided by the supplementary technique KHM, which is introduced in the previous chapter. Figure 5.5 shows a giant spike at the time of over-prediction, and this is a support that RCA over-prediction is accurate. Although quantification of the supplementary technique is not done yet, comparing the values of the new technique from the previous chapter (Figures 4.26, 4.27, and 4.28) with the values seen in Figure 5.5 indicates that when RCA not-okay-prediction flag is up, KHM values will be significantly different from those in case of RCA okay-prediction cases.

However, considering that envelope verification methods could evolve into a stand-alone earthquake early warning algorithm in which one probably would match the arrival times of observed and predicted envelopes, we should look at the performance of RCA and the KHM technique in the case of the arrival time of the calibration pulse of this example matching that of the predicted P-wave. The following section investigates that case.

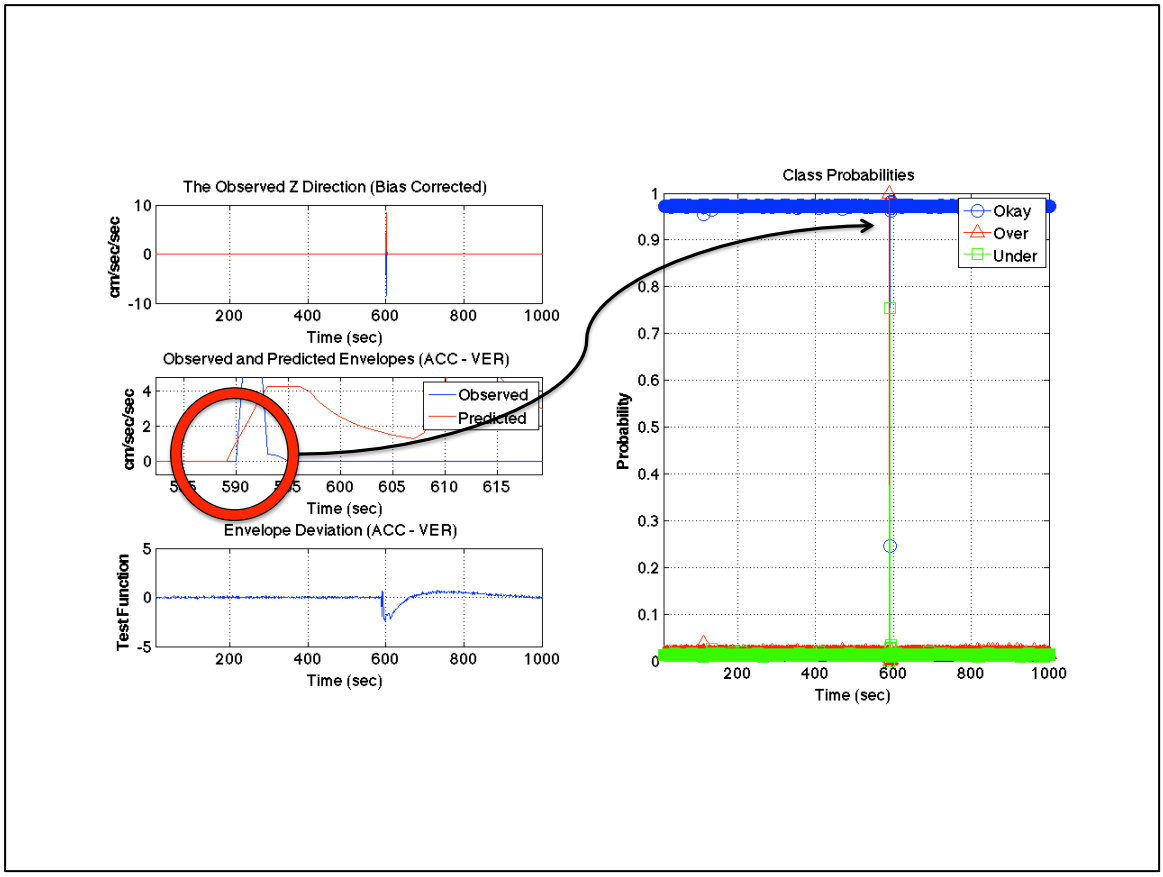

Figure 5.6: Reality Check Algorithm's performance plot for the over-prediction example 1 (with predicted wave arriving earlier than observed calibration pulse) computed using (3.99) from acceleration input using Method II (zoomed-in at arrival times of observed and predicted envelopes). For description of the figure, see *"General figure description - I"* given above.

5.2 Over-prediction example 1 with matching envelope arrival times

It is only natural for the RCA to assume the arrival times of predicted envelopes to be the same as the arrival times of observed ones. In fact, if our method is installed, it will look for solutions of origin time and epicenter location that result in predicted wave arrival times that match those of observed ones. So, in order to see what would happen if the arrival times of predicted envelopes match those of the observed calibration pulse, Figures 5.7 to 5.10 are presented and discussed next.

5.2.1 Discussion of over-prediction example 1 with matching envelope arrival times

The example at hand is clearly an over-prediction case. However, Figures 5.7 to 5.10, with the exception of velocity results, i.e., Figure 5.8, indicate under-prediction! The calibration pulse confused DM to send a M8.2 alarm, so an under-prediction is not acceptable. We can pinpoint the problem, however, by using the KHM.

The cause for the under-prediction classification by most of the inputs is the abnormal rise time of the calibration pulse, which had an amplitude that is equivalent to that of a significantly large earthquake, and the time it took the station to experience that amplitude right after an ambient noise level is too short compared to a real event. That means, by the time RCA compares the calibration pulse amplitude to the corresponding amplitude of the predicted waves, there is a significant difference with the predicted values being much smaller (Figure 5.11). There is an exception though: Figure 5.12 shows that computing displacement from acceleration by integrating twice causes the calibration pulse to appear 1 second later than acceleration in real-time; that is, displacement results still have the predicted waves arrive earlier than the observed

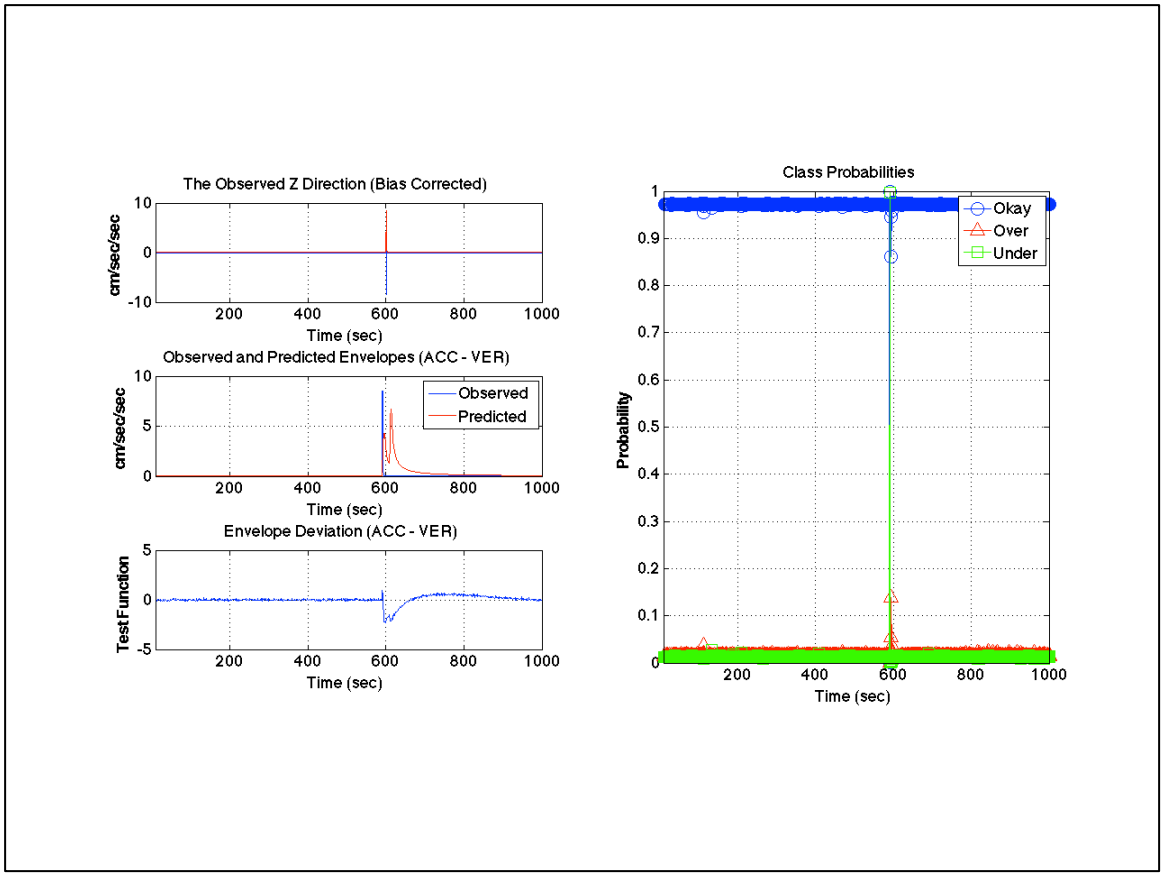

Figure 5.7: Reality Check Algorithm's performance plot for the over-prediction example 1 (with predicted wave arriving at the same time as observed calibration pulse) computed using (3.99) from acceleration input using Method II. For description of the figure, see *"General figure description - I"* given above.

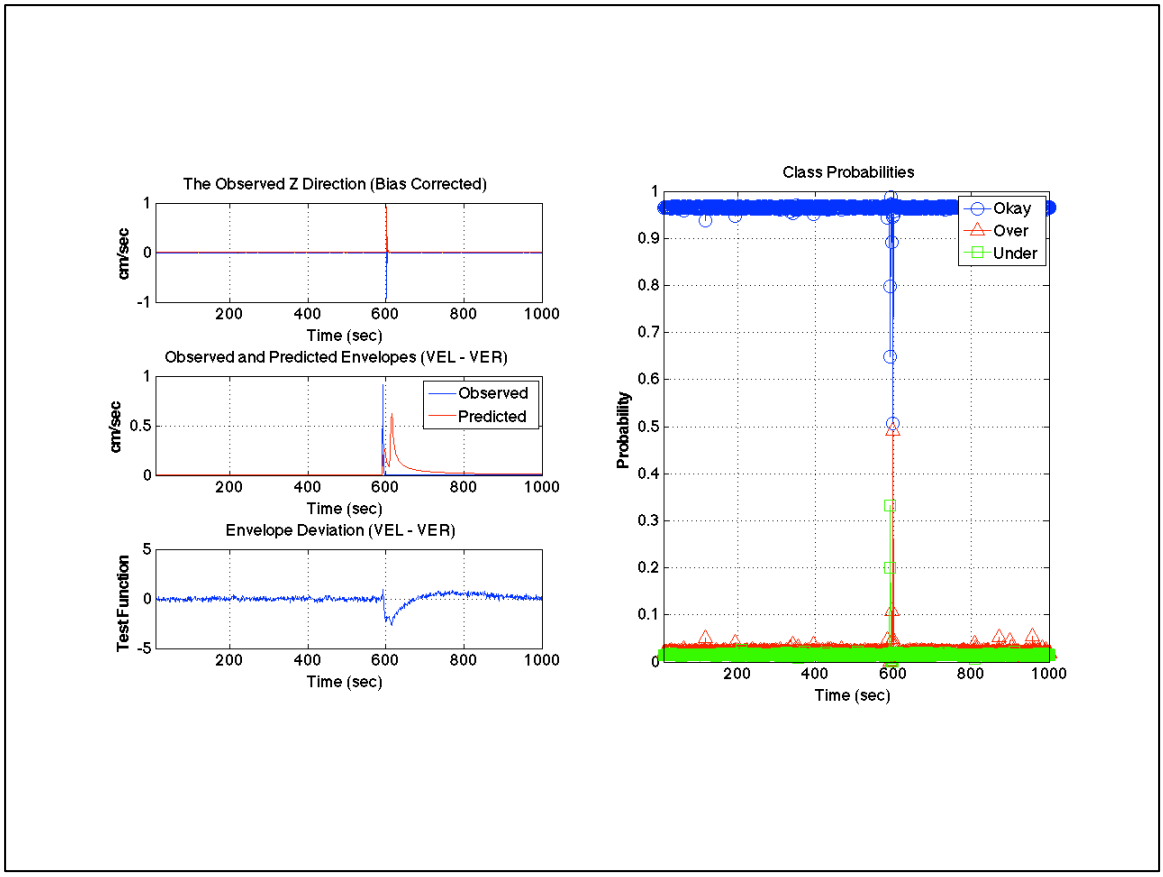

Figure 5.8: Reality Check Algorithm's performance plot for the over-prediction example 1 (with predicted wave arriving at the same time as observed calibration pulse) computed using (3.100) from velocity input using Method II. For description of the figure, see *"General figure description - I"* given above.

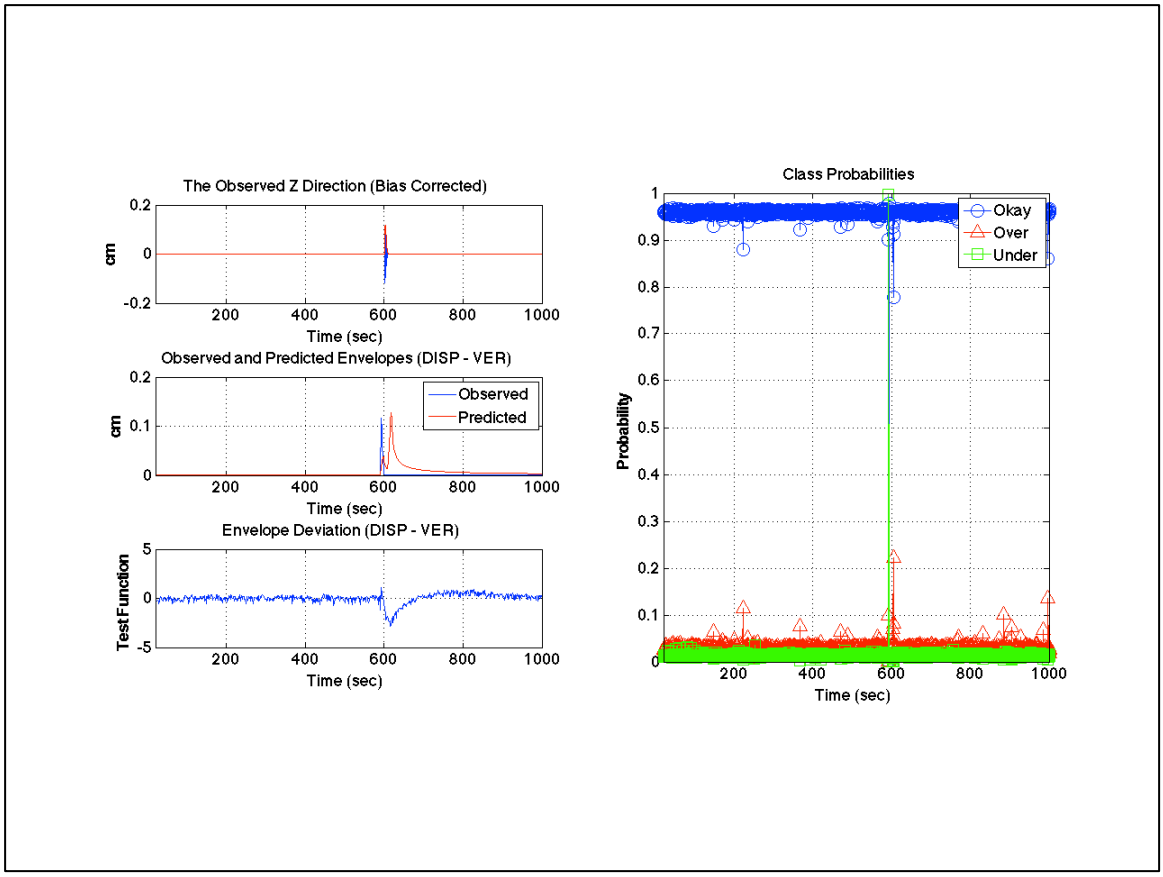

Figure 5.9: Reality Check Algorithm's performance plot for the over-prediction example 1 (with predicted wave arriving at the same time as observed calibration pulse) computed using (3.101) from displacement input using Method II. For description of the figure, see *"General figure description - I"* given above.

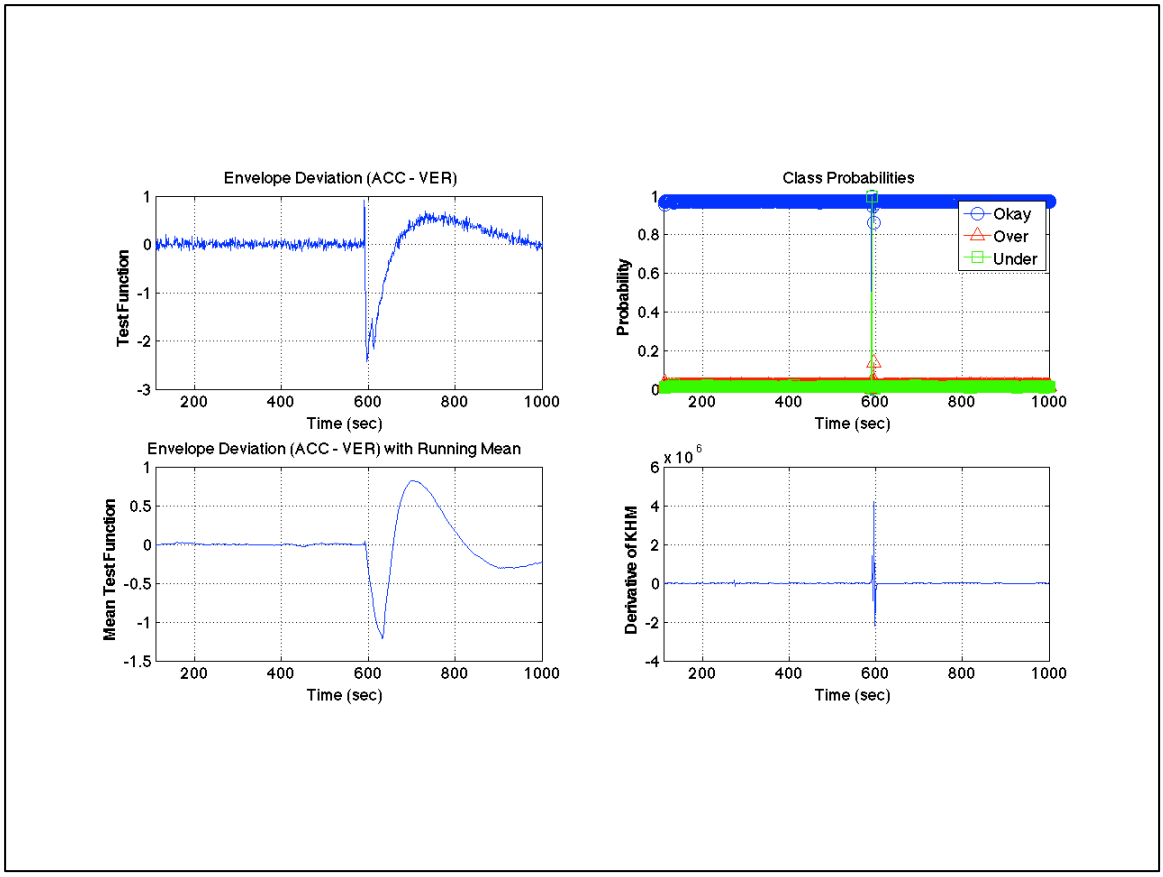

Figure 5.10: Performance of the supplementary method to RCA for the over-prediction example 1 (with predicted wave arriving at the same time as observed calibration pulse). For description of the figure, see *"General figure description - II"* given above.

calibration pulse, just like the previous case. However, even in a situation like that, displacement results still indicate an under-prediction. This is because, by the time predicted P-wave arrives at the station, the corresponding observed envelopes' value is comparable in amplitude to that of the predicted envelope.

Of course, a case where the observed values are significantly larger than predicted values might be considered as under-predictions, but not in this particular context. What do we do? The answer is that we look at the KHM results in Figure 5.10 to make sure we actually have an under-classification as RCA indicates. The abnormally large impulse-like trend in the bottom right plot of Figure 5.10 tells us that there is something wrong. But we still do not know if it is an under-prediction or over-prediction. Because KHM with order 10 would make all components add up to each other since it is an even numbered order, we experience the same discrepancy we experienced with kurtosis in Chapter 2: as far as the even ordered KHM results are concerned, there is not a 'sign' difference between under- and over-predictions. However, an odd numbered moment would preserve the sign of the moment we aim to compute, and therefore could be used to distinguish over-predictions from under-predictions. This fact will become much clearer in the next chapter when we investigate the performance of KHM on an under-prediction case, and compare over-prediction KHM results to that of under-prediction cases.

A point to note is that seismic data acquired from Northern California Earthquake Data Center for the calibration pulse example included a missing data portion indicated by red asterisk in Figures 5.13 and 5.14. Although our methods assume continuous data flow, having some missing data some time later than the cause of the messages for this

particular example, which is the calibration pulse, does not affect the results. However, missing data in the continuous data stream in real-time is a serious issue, and so far we have no solutions to this problem.

5.3 Over-prediction example 2

On May 4th, 2011, Decision Module (DM) experienced a false alarm of M8.0 (an email about this false alarm was sent to several scientists working on the project). The alert was due to a signal from seismic station PLM in network CI. By using information from the emails exchanged about this particular false alarm among scientists, we demonstrate performances of both the RCA and the KHM below. We did not use the log files of DM that included this false alarm, but we used pick time information from the emails as the assumed arrival time of the predicted P-wave. Upon further investigating the continuous strong motion records from seismic station PLM in network CI, we noticed that the indicated pick time did not have a visible abnormality. However, that station experienced an abnormal increase in the noise amplitude several seconds prior to the pick time indicated in the emails. Just to be sure, we provide the performance of RCA and KHM for two cases: case one is where we take the arrival time of the predicted P-waves as it is indicated in the emails, and case two is where we take the arrival time of the predicted P-waves as the arrival time of abnormally large non-earthquake noise.

As mentioned before, we did not use a documented origin time and location for this alarm. Considering we have a station, a pick time, and a magnitude, we decided to use predicted envelopes created using a magnitude of 6.5 and a distance of 200 km. Figures 5.15 to 5.22 show that even if the RCA and KHM use predicted waves created using the information above, the case is still classified as over-prediction.

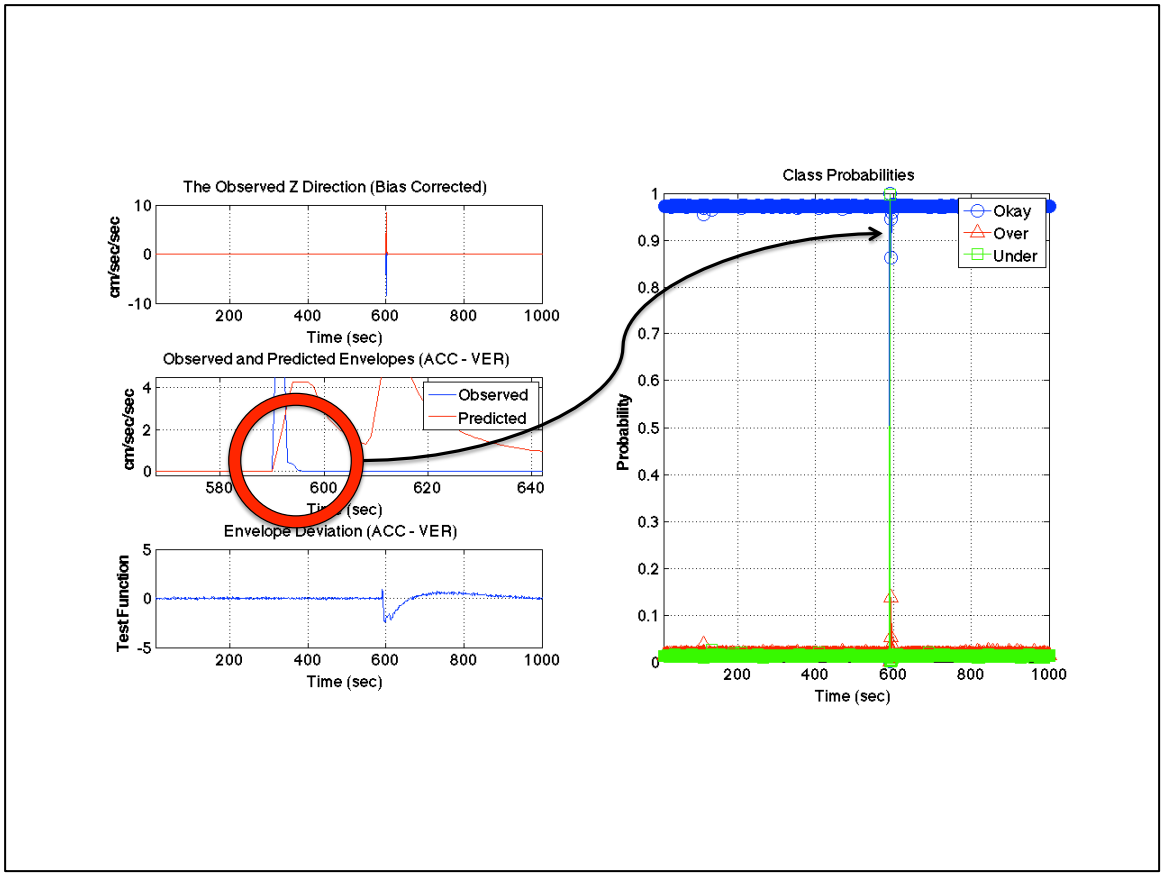

Figure 5.11: Reality Check Algorithm's performance plot for the over-prediction example 1 (with predicted wave arriving at the same time as observed calibration pulse) computed using (3.99) from acceleration input using Method II (zoomed-in at arrival times of observed and predicted envelopes). For description of the figure, see *"General figure description - I"* given above.

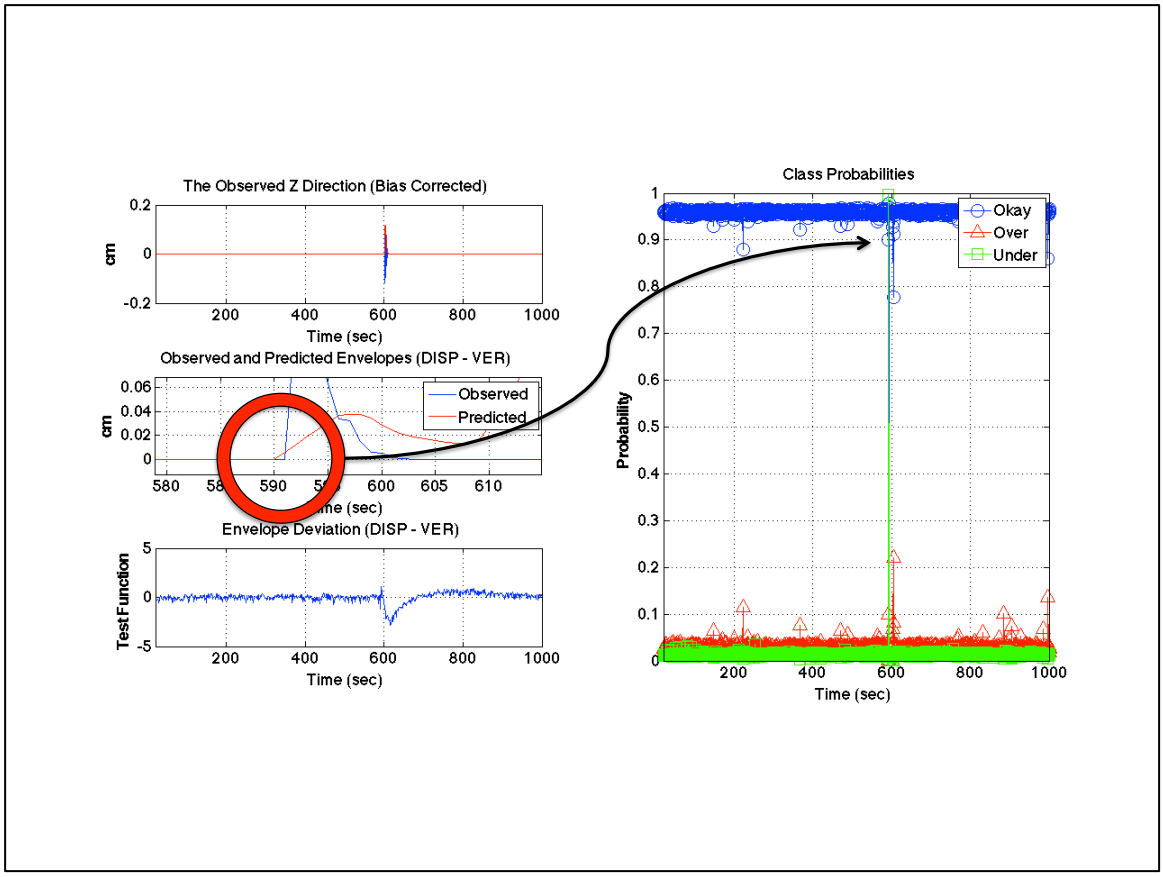

Figure 5.12: Reality Check Algorithm's performance plot for the over-prediction example 1 (with predicted wave arriving at the same time as observed calibration pulse) computed using (3.101) from displacement input using Method II (zoomed-in at arrival times of observed and predicted envelopes). For description of the figure, see *"General figure description - I"* given above.

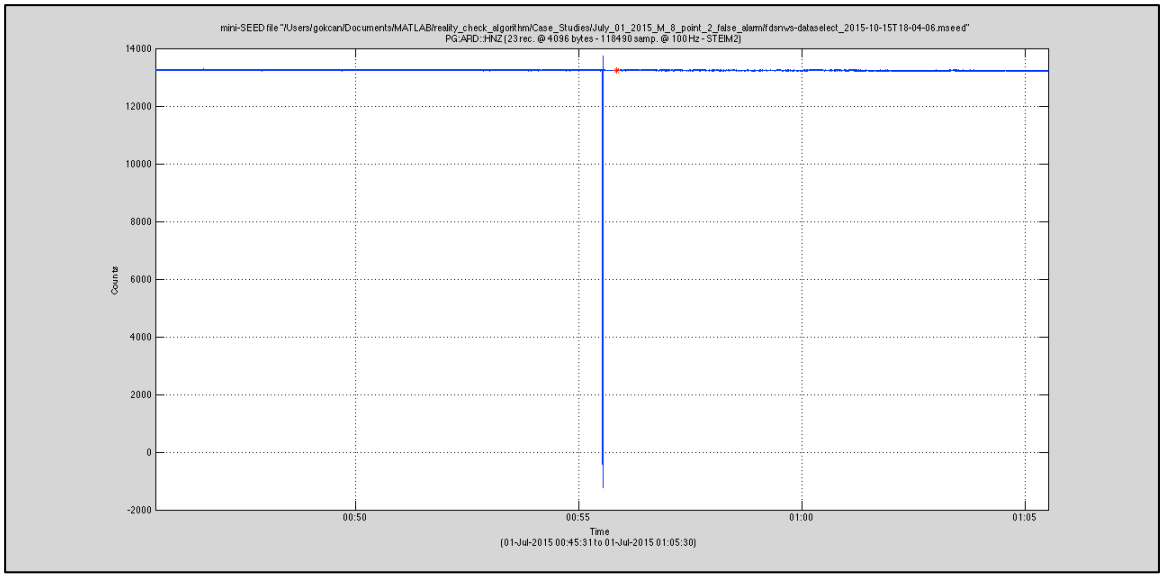

Figure 5.13: Seismic data for the calibration pulse that is used as an over-prediction example 1 from the Northern California Earthquake Data Center. A couple of seconds after the arrival of the calibration pulse, the continuous data stream has some missing data, which are indicated by the red asterisk.

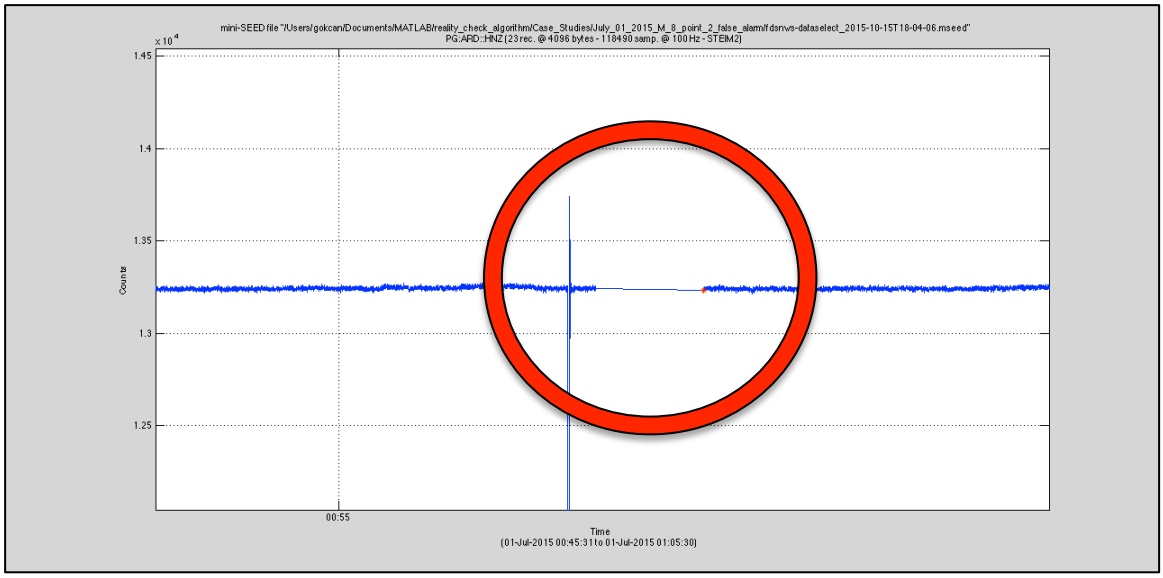

Figure 5.14: Seismic data for the calibration pulse that is used as an over-prediction example 1 from the Northern California Earthquake Data Center, zoomed-in around missing data.

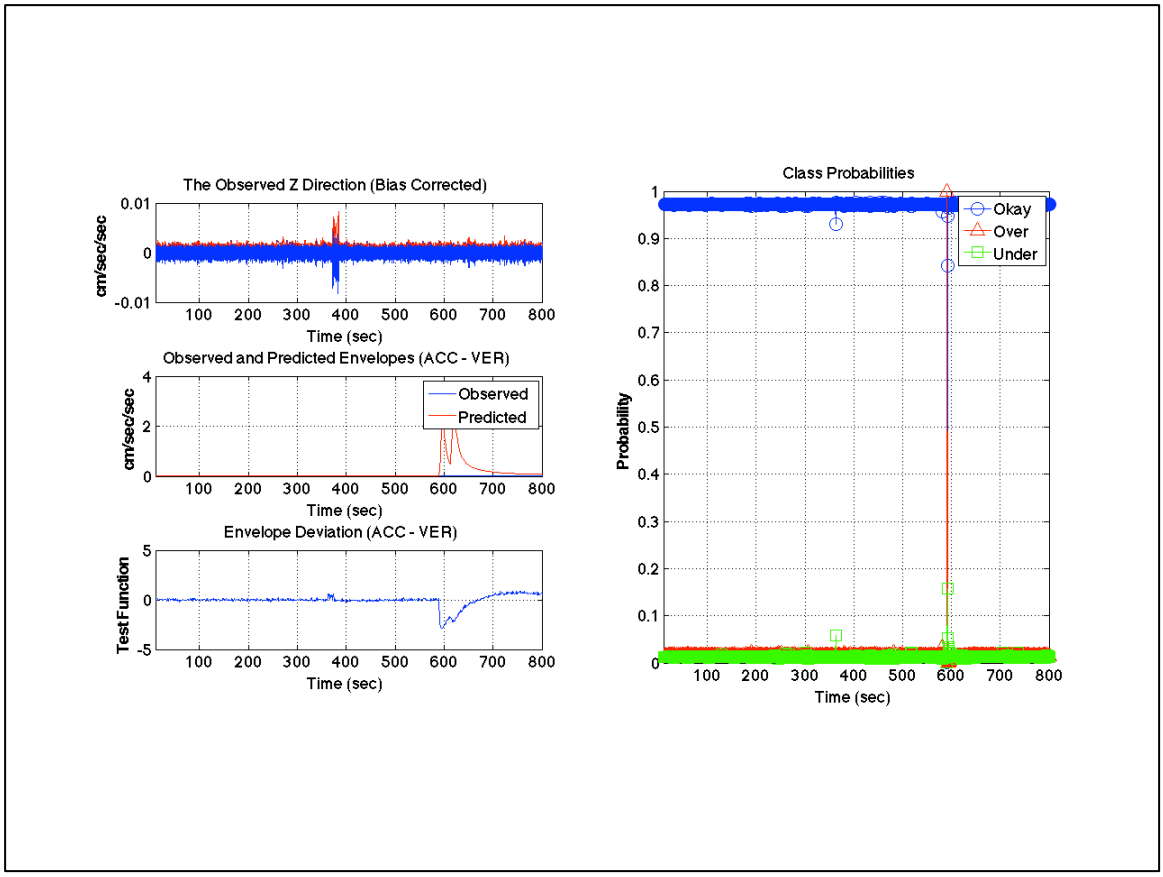

Figure 5.15: Reality Check Algorithm's performance plot for the over-prediction example 2 computed using (3.99) from acceleration input using Method II. For description of the figure, see *"General figure description - I"* given above.

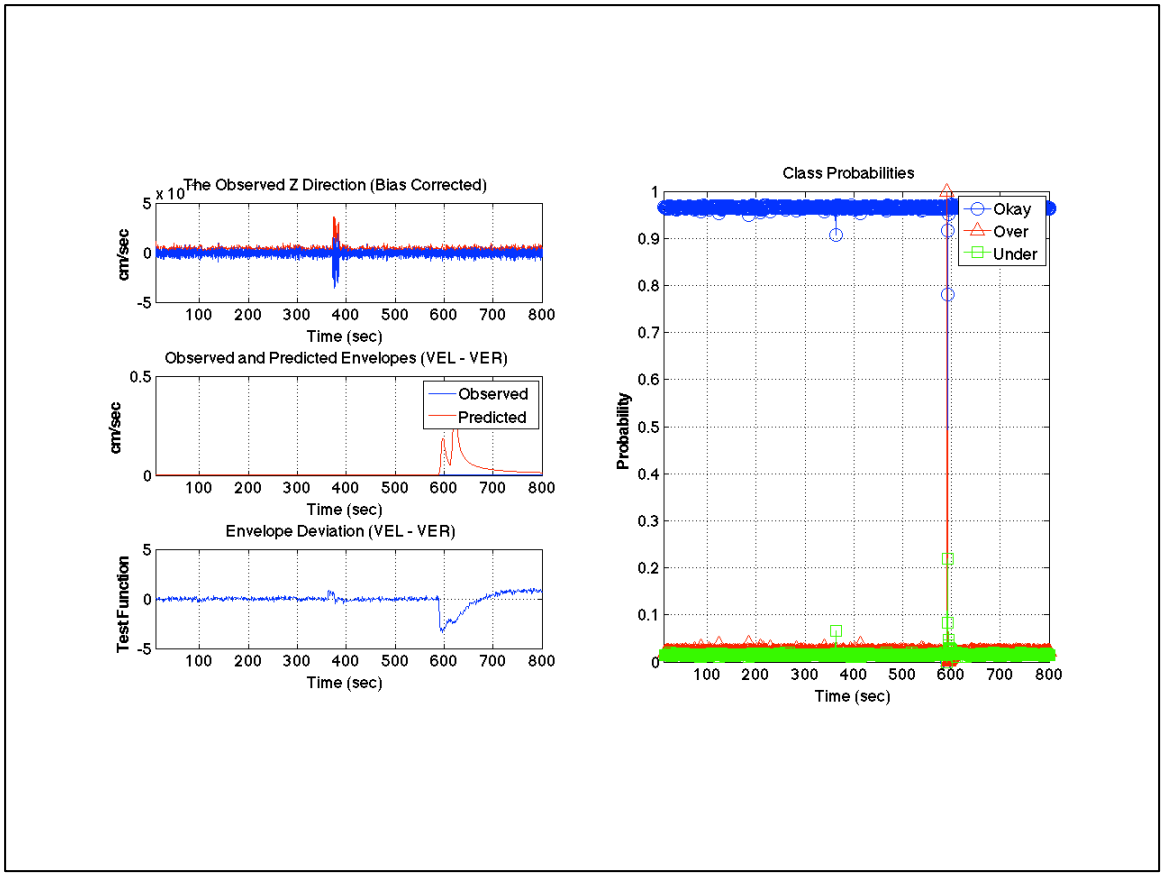

Figure 5.16: Reality Check Algorithm's performance plot for the over-prediction example 2 computed using (3.100) from velocity input using Method II. For description of the figure, see *"General figure description - I"* given above.

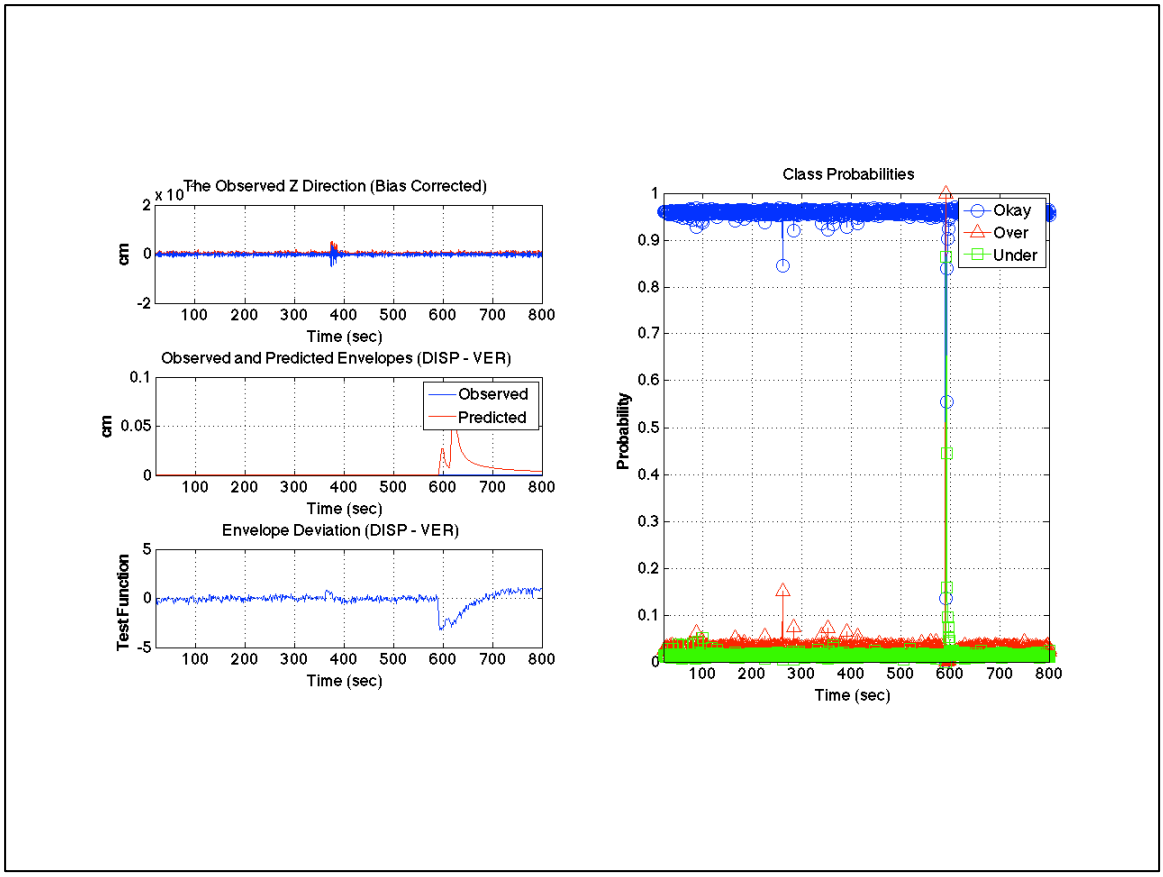

Figure 5.17: Reality Check Algorithm's performance plot for the over-prediction example 2 computed using (3.101) from displacement input using Method II. For description of the figure, see *"General figure description - I"* given above.

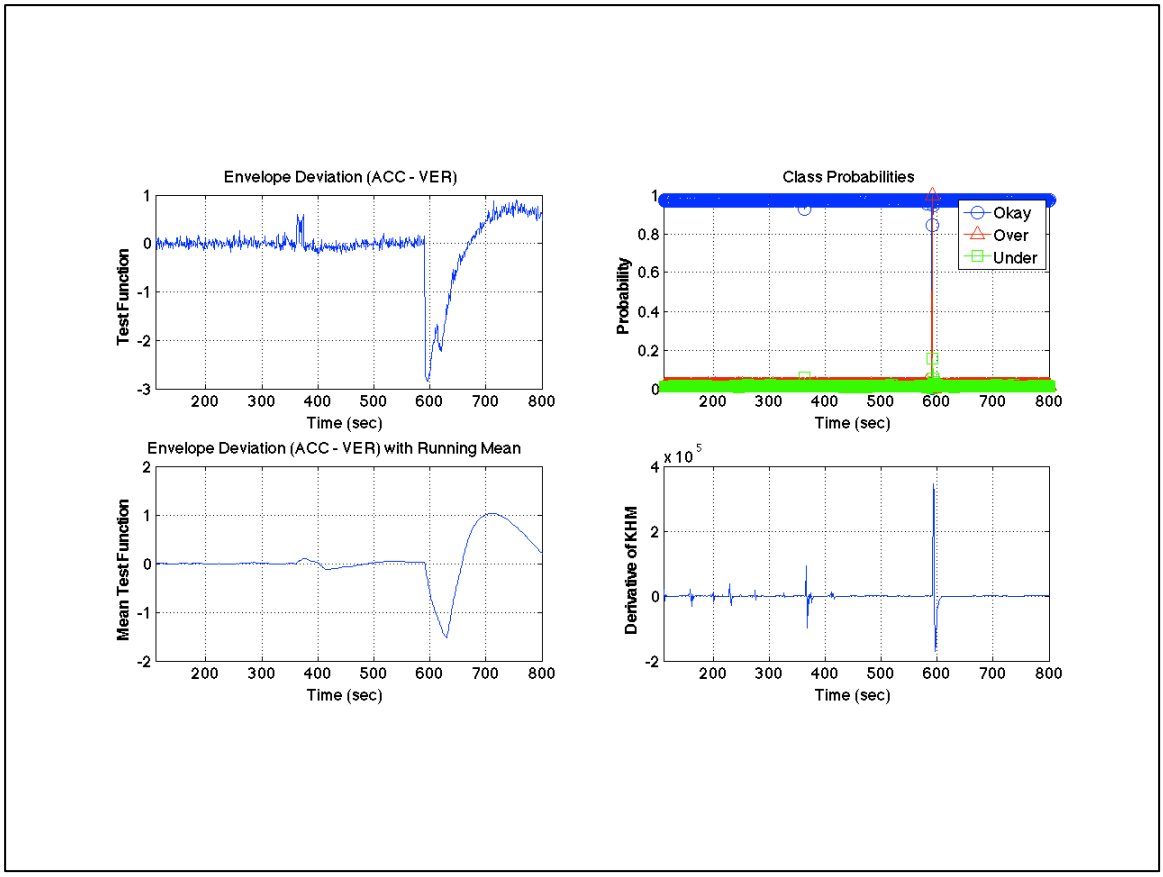

Figure 5.18: Performance of the supplementary method to RCA for the over-prediction example 2. For description of the figure, see *"General figure description - II"* given above.

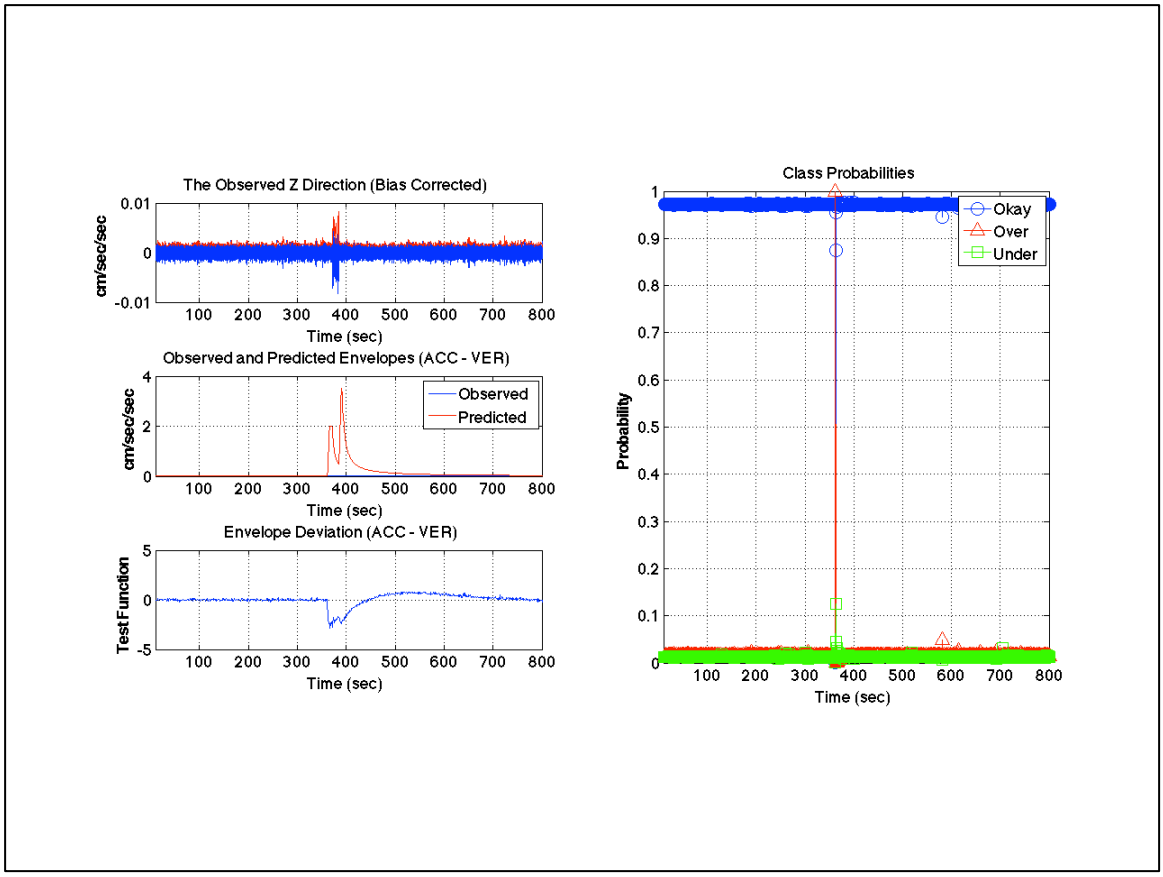

Figure 5.19: Reality Check Algorithm's performance plot for the over-prediction example 2 computed using (3.99) from acceleration input using Method II with matching arrival times of observed abnormal noise increase and predicted P-wave. For description of the figure, see *"General figure description - I"* given above.

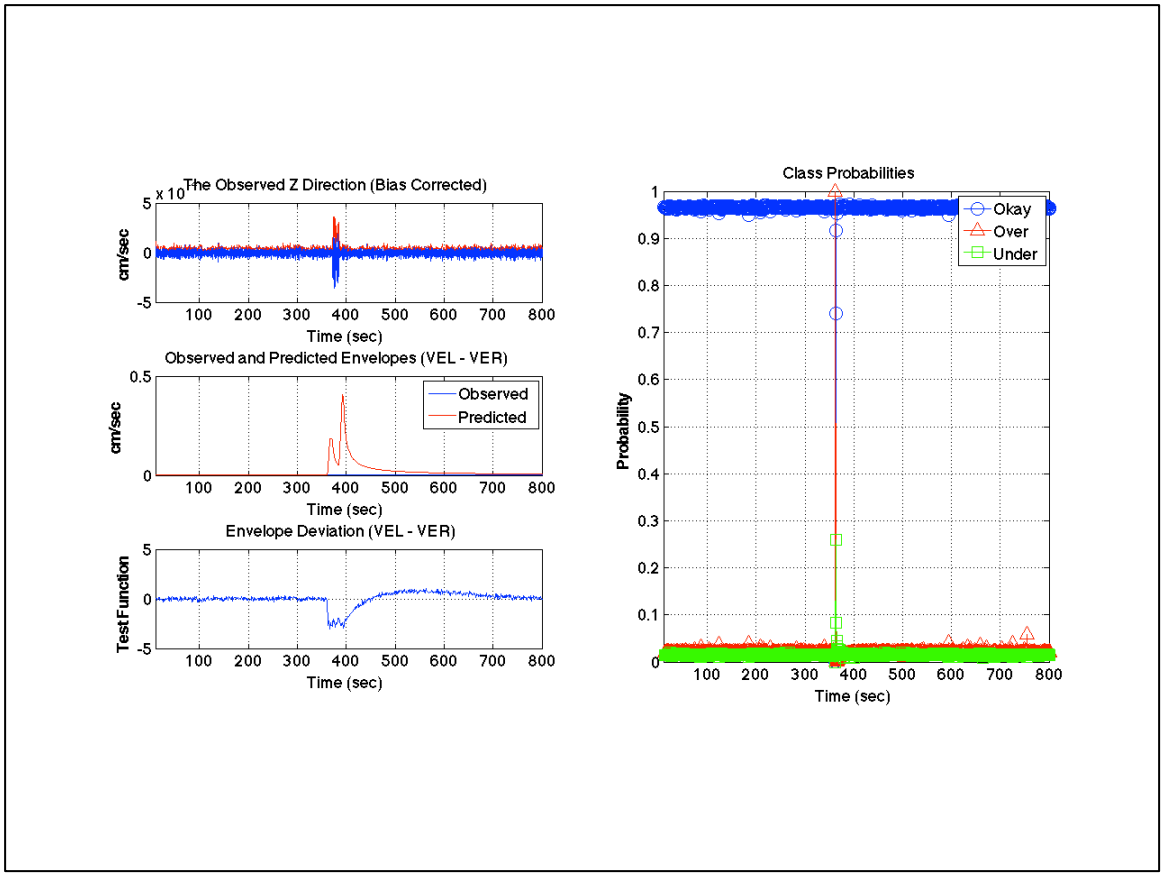

Figure 5.20: Reality Check Algorithm's performance plot for the over-prediction example 2 computed using (3.100) from velocity input using Method II with matching arrival times of observed abnormal noise increase and predicted P-wave. For description of the figure, see *"General figure description - I"* given above.

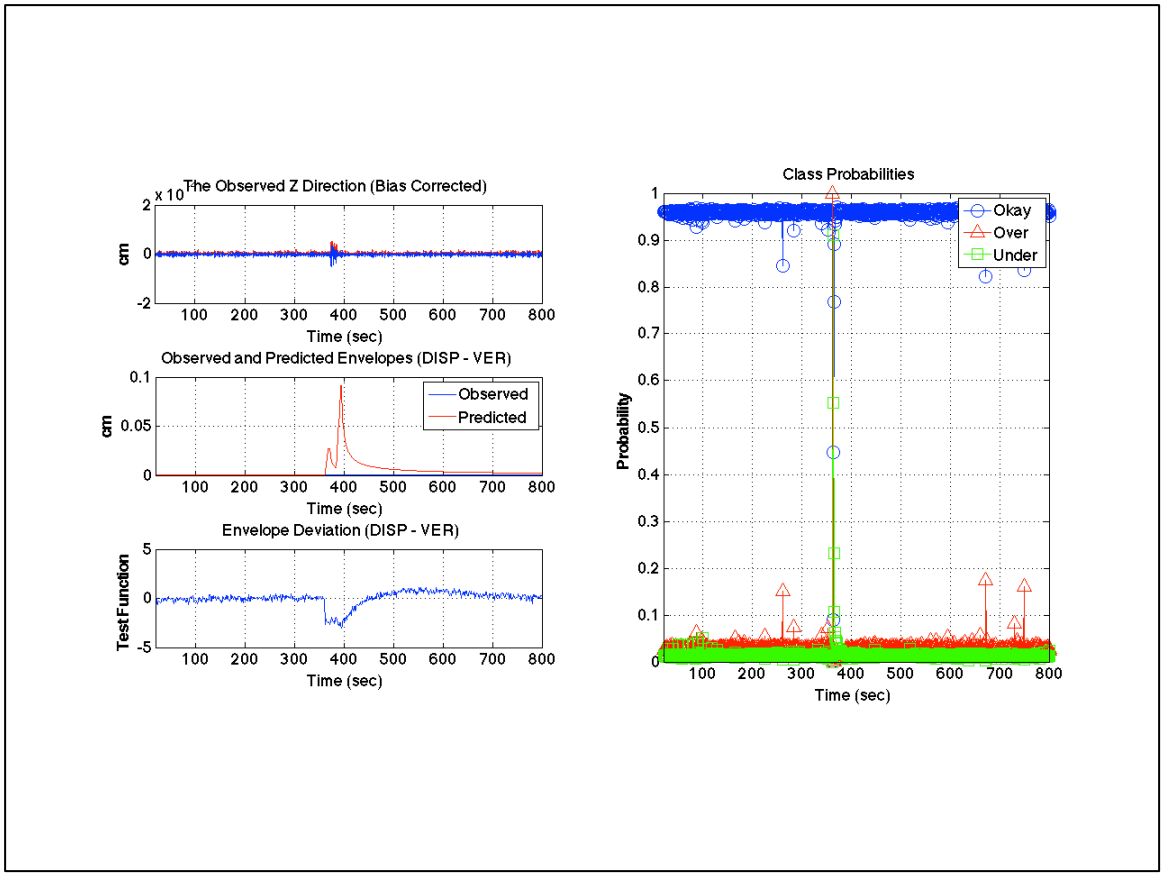

Figure 5.21: Reality Check Algorithm's performance plot for the over-prediction example 2 computed using (3.101) from displacement input using Method II with matching arrival times of observed abnormal noise increase and predicted P-wave. For description of the figure, see *"General figure description - I"* given above.

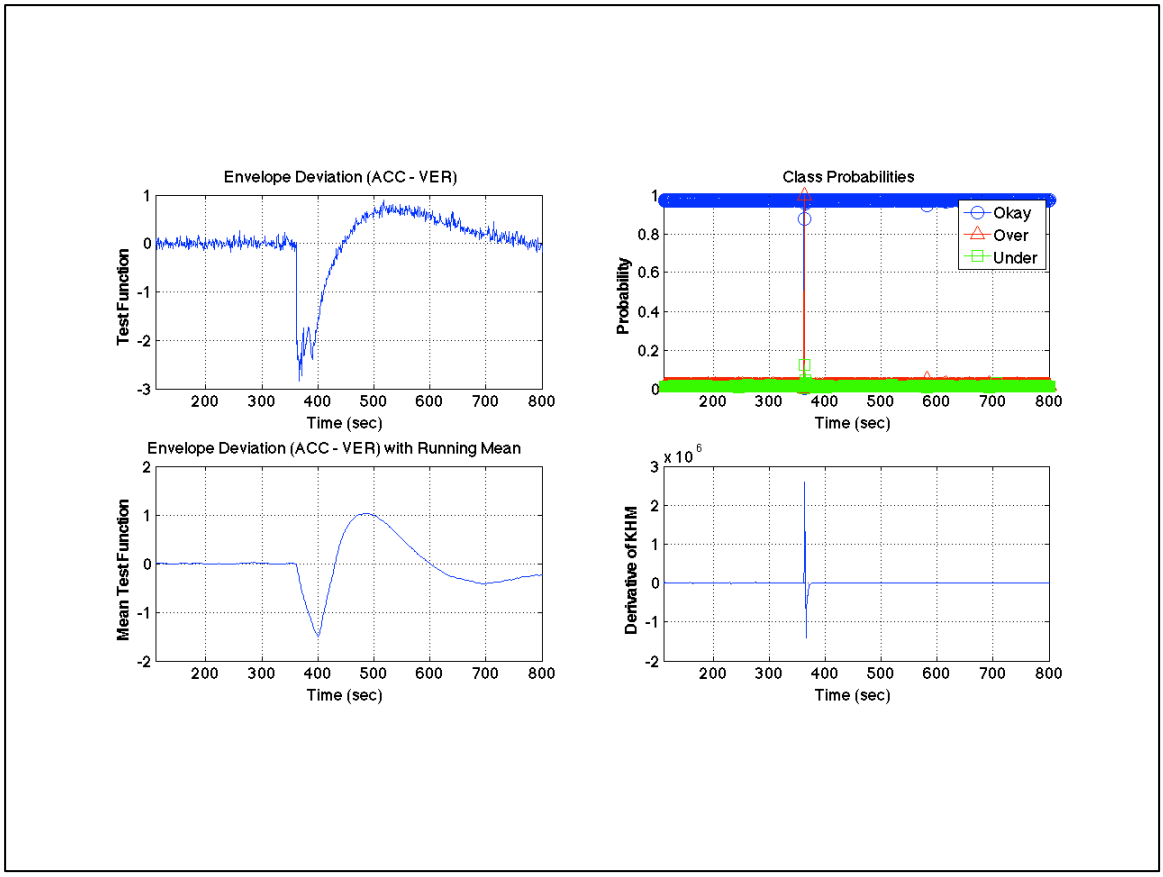

Figure 5.22: Performance of the supplementary method to RCA for the over-prediction example 2 with matching arrival times of observed abnormal noise increase and predicted P-wave. For description of the figure, see *"General figure description - II"* given above.

5.3.1 Discussion of over-prediction example 2

Let us start with Figures 5.15 to 5.18; all results, i.e., acceleration, velocity and displacement indicate over-prediction. This is not surprising because the observed value consists of only noise, which is somehow a consistent amplitude except for the temporary increase in earlier portions of the record. Then, all of a sudden, a P-wave predicted envelope enters the computation, and this leads to over-prediction indication by RCA. Moreover, the KHM result, which is seen in Figure 5.18, supports RCA in that the predicted ground motion is not in accord with what is being observed. But that is as far as the KHM results go for the time being, that is, we need to evaluate KHM results in a consistent framework so that we can classify 'something is wrong' message by KHM into either 'over-prediction' or 'under-prediction' case. An abnormal noise value in the early portion of the observed ground motion causes the under-prediction probability for acceleration and velocity results to increase a little. However, this increase does not confuse RCA, and it accurately classifies the case as okay-prediction. Looking at area of the abnormality in Figure 5.18 provides another reason why KHM needs quantification; the abnormal increase in the observed ground motion noise level creates a sudden increase in KHM values, but this increase is not as much as the one caused by the predicted P-wave that arrives sometime later. Therefore, quantification of KHM should help us determine what values indicate okay-prediction cases even though they are significantly different from the rest of the okay-prediction cases.

When we move on to Figures 5.19 to 5.22, we see that even if DM was tricked into believing there was an earthquake due to the abnormal increase in the observed ground motion noise level, RCA and KHM would correctly classify this case as over-

prediction. Similar to the previous figures of over-prediction example 2, over-prediction classification can be observed in all of the results for Method II.

Chapter 6

Case Studies – Under Predictions

In this chapter, we demonstrate the performance of RCA and KHM on the M7.2 event of Cucapah-El Mayor on April 4th, 2010 as an under-prediction case example. Although the Decision Module (DM) was not producing alerts and log files at the time, we use the location and magnitude estimate provided by the research group at Caltech who has been testing DM's off-line performance on significant earthquakes of the past. According to the log files, one of the algorithms in DM first predicted the magnitude as 5.434 and a location with latitude 31.905 and longitude -115.196. The particular seismic station, on which we demonstrate the performance of our algorithms, is GLA in network CI. Method II is first applied and the KHM technique, introduced in Chapter 4, is used to mitigate problems associated with the examples. The results are presented in Figures 6.1 to 6.5.

First, we would like to clarify a particular point: we use 10 minutes of continuous record before the catalogued origin time of this event as well as 10 minutes of continuous record starting from the catalogued origin time. In this particular case, there happens to be a smaller event in the pre-event continuous data, and because the corresponding predicted envelope consists only of noise at the time of the smaller event, RCA accurately predicts it as an under-prediction, or in other words, a missed event. So, the first under-prediction indication in the following figures, with the exception of those for displacement, around time 254th second is due to that small event which was not considered to be there for these particular simulations.

Similar to Chapter 4, we provide two general descriptions for figures of this chapter.

General figure description - I:

Left panel: first row shows the seismogram recorded by the seismic station whose identification information is given above (seismogram in blue and its envelope in red); note that although only the vertical channel is shown in the plot for demonstration purposes, both vertical and horizontal channels were used in RCA computations; the second row shows both the predicted and observed envelopes; the third row shows the test function after high-pass filtering by the values given in Chapter 2. Right panel shows the probability values for each class with a different color and marker.

General figure description - II:

Left panel: first row shows the test function after high-pass filtering by the values given in Chapter 2; second row shows the test function values after applying a running mean and a high-pass filter by the values given in Chapter 2. Right panel: first row shows the probability values computed using (3.99) from acceleration input using Method II for each class with different color and marker; second row shows the results of the supplementary technique.

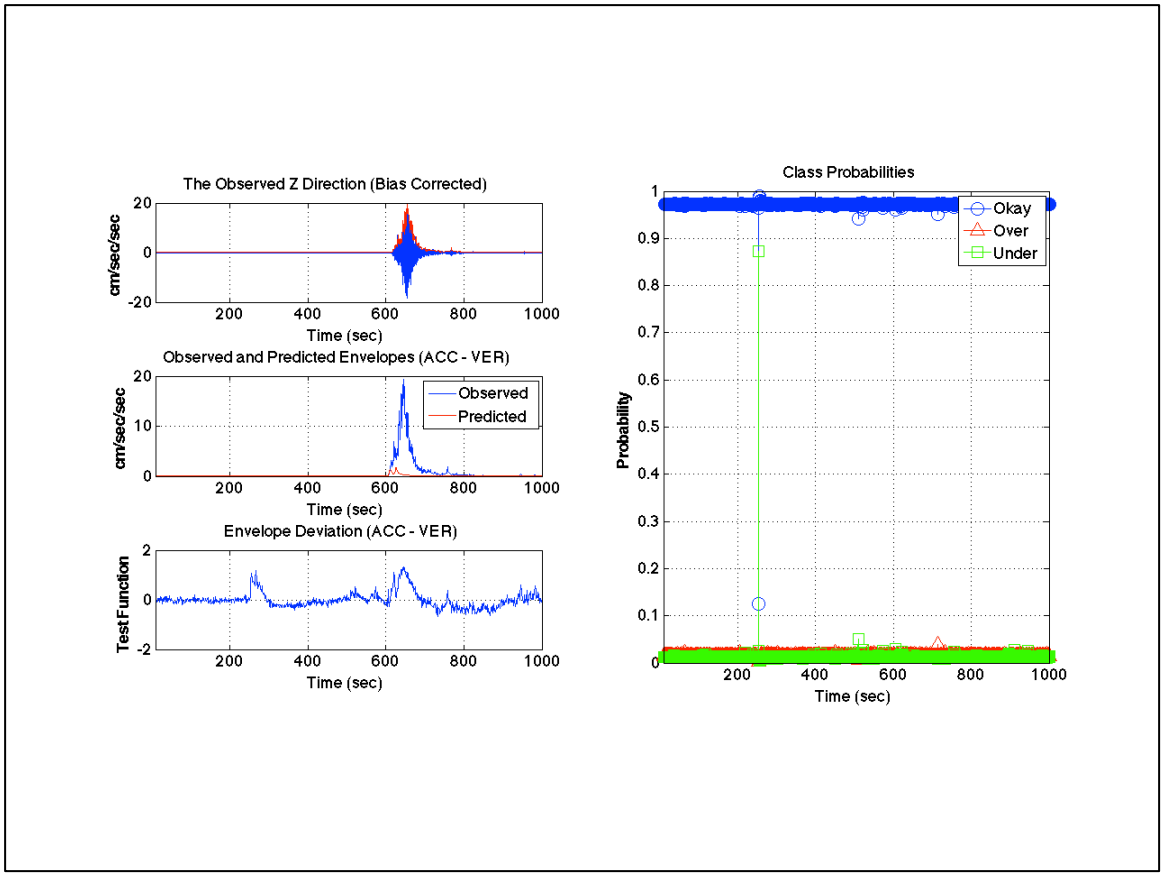

Figure 6.1: Reality Check Algorithm's performance plot for the under-prediction example computed using (3.99) from acceleration input using Method II. For description of the figure, see *"General figure description - I"* given above.

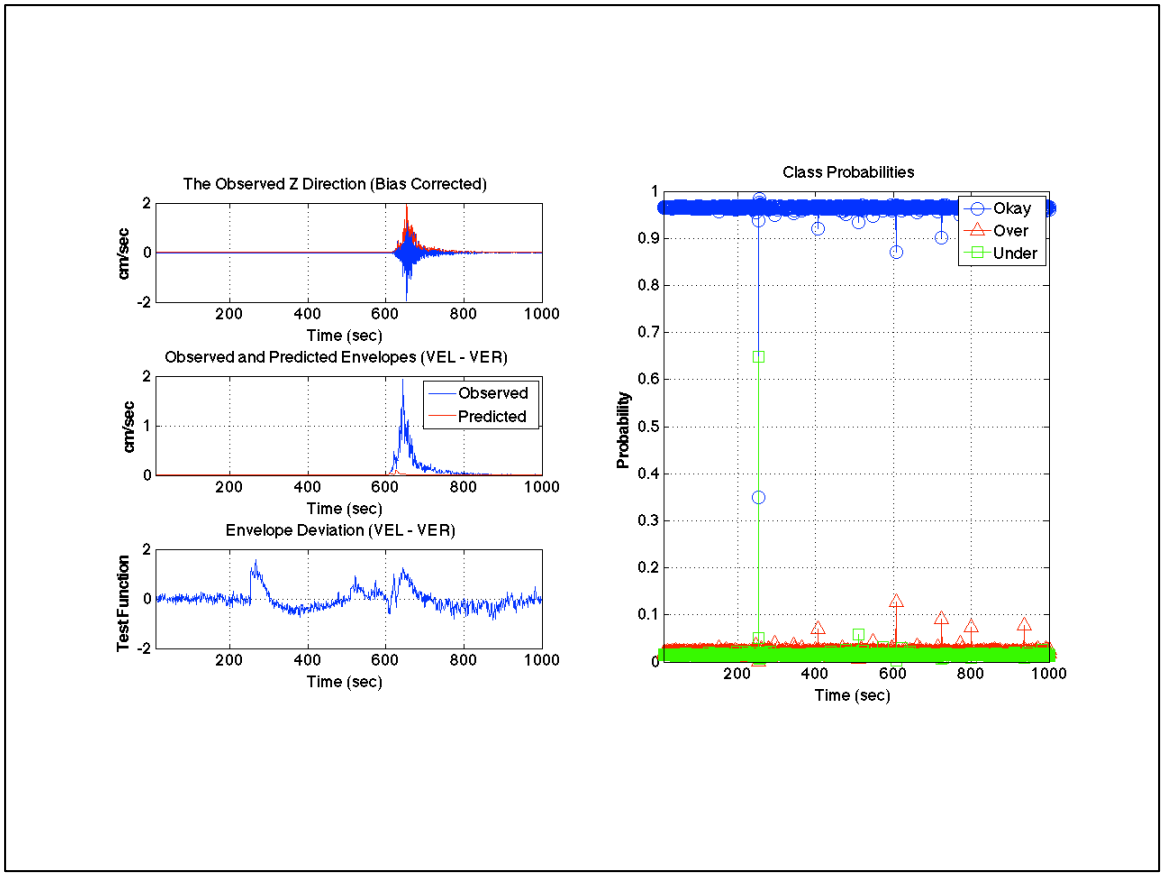

Figure 6.2: Reality Check Algorithm's performance plot for the under-prediction example computed using (3.100) from velocity input using Method II. For description of the figure, see *"General figure description - I"* given above.

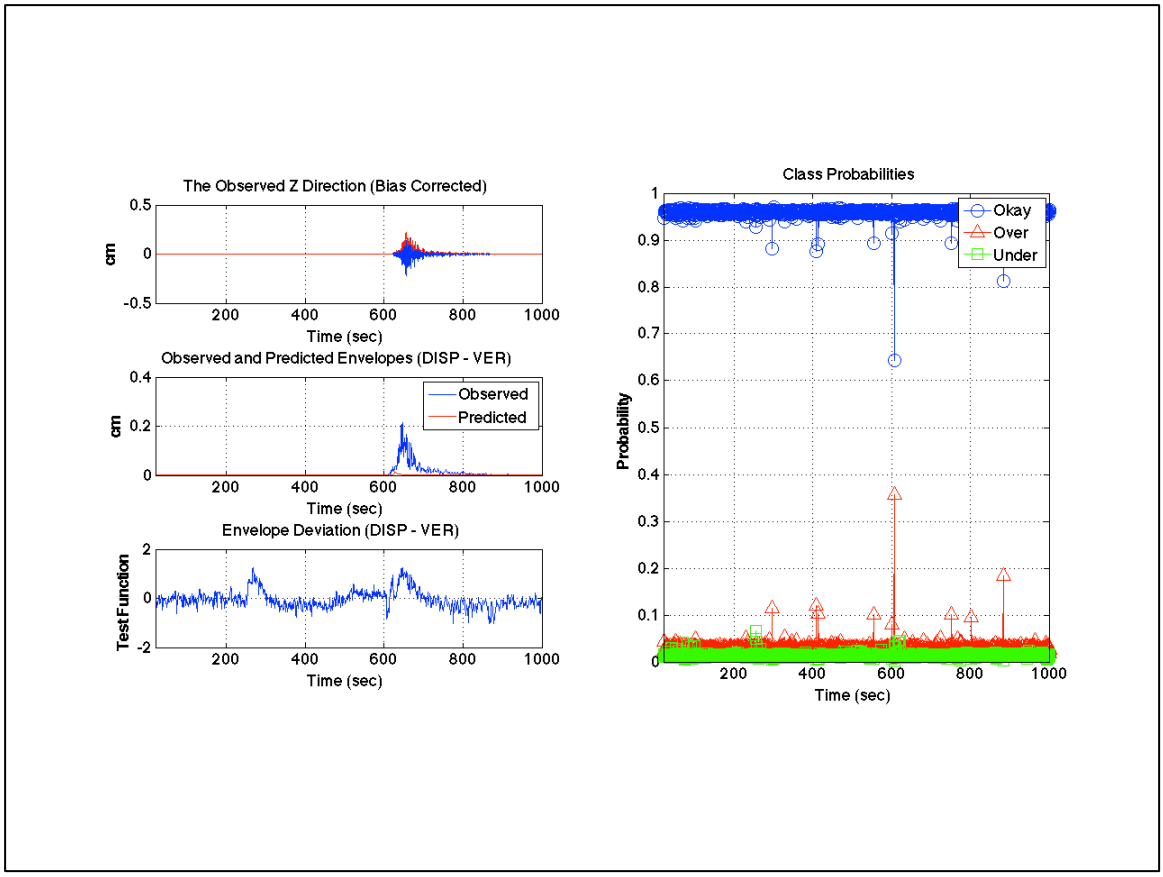

Figure 6.3: Reality Check Algorithm's performance plot for the under-prediction example computed using (3.101) from displacement input using Method II. For description of the figure, see *"General figure description - I"* given above.

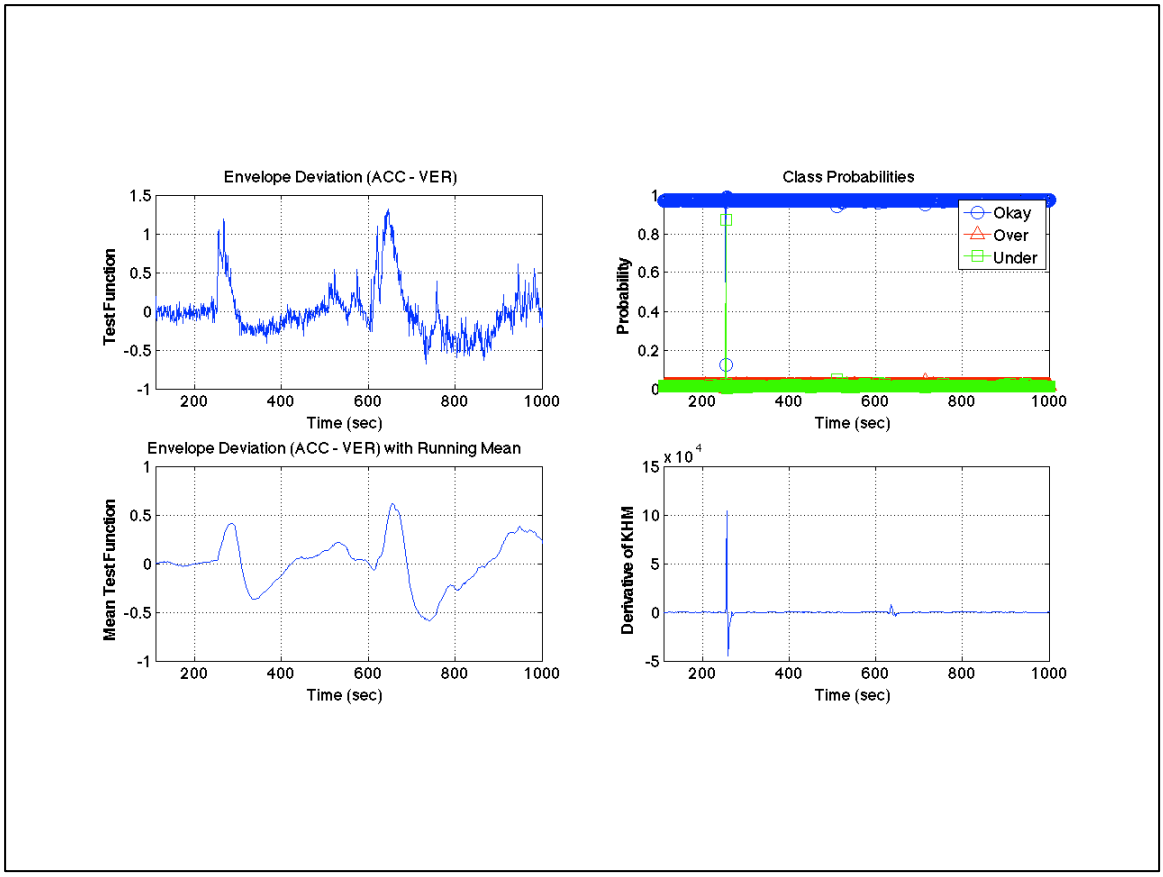

Figure 6.4: Performance of the supplementary method to RCA for the under-prediction example. For description of the figure, see *"General figure description - II"* given above.

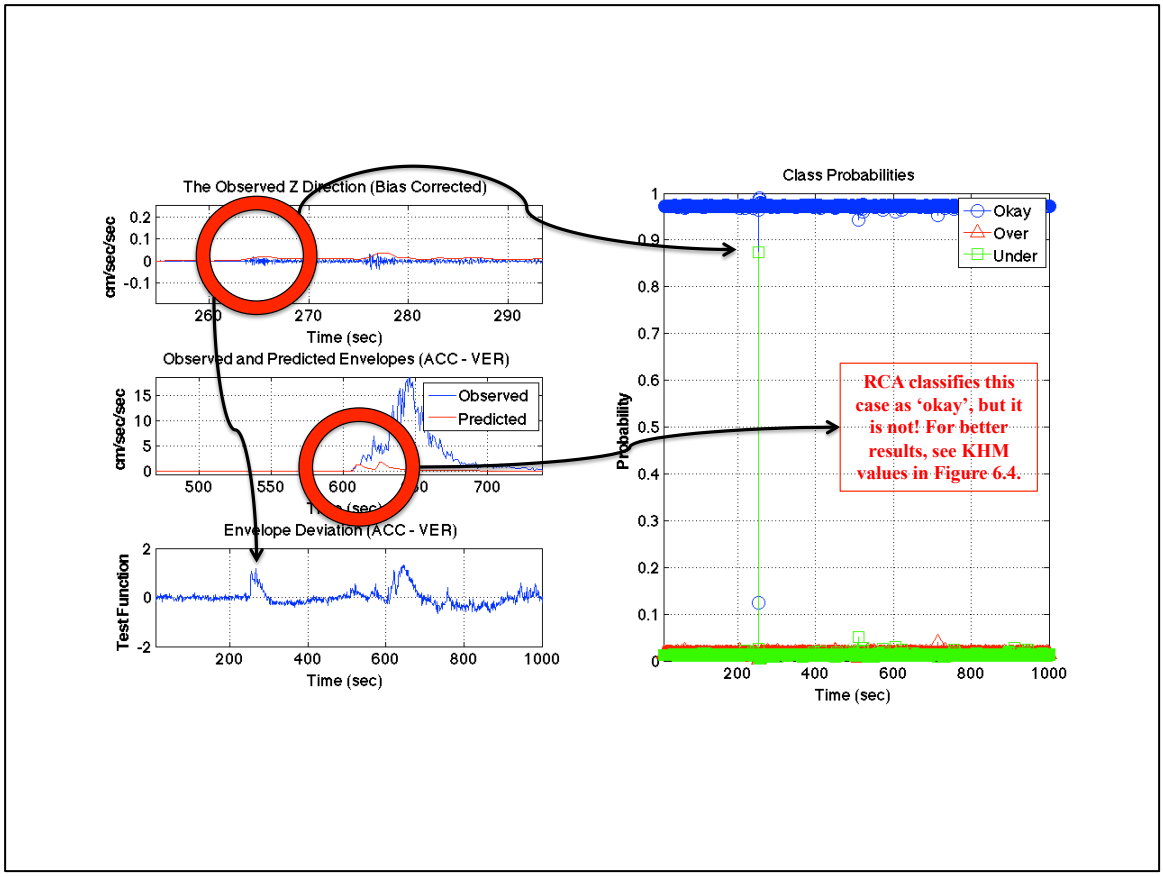

Figure 6.5: Reality Check Algorithm's performance plot for the under-prediction example computed using (3.99) from acceleration input using Method II (zoomed-in at arrival time of the small preceding event in the seismogram, and Cucapah – El Mayor earthquake in ground motion envelopes). For description of the figure, see *"General figure description - I"* given above.

6.1 Discussion of under-prediction example

Let us start with the small event preceding M7.2 earthquake of Cucapah-El Mayor in 2010. According to the catalogue, this event, whose magnitude is 3.35, has the origin time of 2010, April 4 at 22:34:50.160 UTC, which is approximately 6 (six) minutes before the M7.2 event. It is located at latitude 32.2297 and longitude -115.2952. Figure 6.5 shows a zoomed-in version of Figure 6.1 with the data associated with that small event highlighted. Note that seismograms, which are shown in the panel in the upper left, have 10 seconds of more data, i.e., envelopes and RCA computations have 10 seconds less data to get rid of transitional effects of filtering. That is why the arrival time of the small event is at the 264^{th} second in the seismogram (Figure 6.5) while the under-prediction indication on the right is at the 254^{th} second. Except for displacement, all of the results managed to accurately classify it as an under-prediction, i.e., missed alarm.

When RCA makes computations about the time of M7.2 event, there is no indication of under-prediction. This is due to the fact that P-wave of M5.434, which was predicted by one of the algorithms in DM, is comparable in amplitude to that of a M7.2 event. In this case, one would expect RCA to indicate under-prediction when the S-wave of the M7.2 comes into the calculations. That does not happen! Since, by the time the S-wave of M7.2 earthquake arrives, the average misfit between predicted and observed values is so different from that of when there is only noise envelopes, RCA is unable to detect the misfit to be an under-prediction case, it still thinks predictions are okay. They are obviously not, because the energy of an M5.434 earthquake (predicted envelopes) is significantly less than that of an M7.2 event. However, the KHM method from Chapter 4

indicated that there is something wrong about the prediction (Figure 6.4) by looking at the ‘overall’ fit between observed and predicted values.

One cannot help but notice the amplitude difference between the KHM values for the small and the big event. Although the small event has a significantly smaller magnitude than the big one, the KHM value associated with it is much bigger than that associated with the Cucapah - El Mayor earthquake. This difference is because the small event is completely missed, i.e., the assumed corresponding predicted envelope consists only of noise whereas the big event is not totally missed, it is only under-predicted. This shows another reason why we need rigorous interpretations of KHM quantities (i.e. what is the degree of under- or over- prediction associated with KHM values?).

Chapter 7

Concluding Remarks and Future Work

This work primarily examines the reliability the predictions made by an earthquake early warning system. Although a fully functioning earthquake early warning system is highly desirable in seismic regions, the presence of false and missed alarms may negatively affect the public's perception of such a system. The proposed Reality Check Algorithm aims to minimize the mistakes an earthquake early warning system might make by continuously monitoring what the system predicts and what the spatially distributed ground motion actually is. The focal point of this work has been using the ground motion envelopes created by the Virtual Seismologist (VS) (Cua 2005). Although Cua provides distribution of envelope parameters, we use the 'mean' values. Also, note that implementing the VS GMPE's, which Cua created, in real-time was not an easy task. So, we accomplish that task via the Reality Check Algorithm, which is described in this thesis.

The possible prediction scenarios that may be made by an earthquake early warning system are categorized into three discrete classes: okay-, over-, and, under-prediction classes. Okay-prediction is the ideal case where the alert sent by the system is acceptably accurate about the earthquake being experienced. Over-prediction occurs when the alert is an overestimation caused by higher amplitude levels at seismic stations; for example, a relatively short burst of unusually high level of noise being mistaken for a large magnitude earthquake. Under-prediction occurs when the system does not infer how big the earthquake actually is.

Earthquake early warning system's predictions are used to create VS predicted envelopes and the result is assigned to one of the previous three classes. The first classification method examined used a straightforward linear discriminant analysis (see Appendix C), but it does not produce probabilities on each class individually, except for assigning 100% probability for one class and zero for others. Moreover, we cannot easily aggregate its results with those of other potential probabilistic algorithms. Therefore, a Bayesian probabilistic classification was investigated next (see Chapter 3) where a probability (degree of plausibility) is computed for each class. The Bayesian approach also allows one to sequentially update the uncertainties associated with the system predictions. The probabilities that are calculated for a system can be systematically and rigorously combined in real-time. Although the results presented in this work are based on a single seismic station's computations, the Bayesian methodology's sequential updating capability allows uncertainty calculations from multiple stations to be readily combined; giving 'feedback' to the earthquake early warning system to inform it about its prediction.

After completing the classification analyses, we moved on to test the results on several examples. Although RCA performed impressively in terms of okay-predictions, it produced some discrepancies for over- and under-prediction cases; as far as RCA was concerned, an early arriving predicted P-wave of an otherwise accurately estimated earthquake could show the same probability values as a false alarm. Moreover, late arriving predicted envelopes might be considered the same as a missed earthquake. To overcome these problems an overall misfit measure was introduced that we call the Karakus-Heaton Moment for Signal Data (KHM), which checks the overall fit between

observed and predicted earthquake envelopes while being insensitive to the timing of them. Therefore, while RCA makes sure the timing of the envelopes are in agreement, which translates to origin time and location predictions, KHM assesses the degree of magnitude match between the predictions and observations. The KHM method shows good promise. In future work, the next step would be to analyze the optimal selection of the values of the KHM parameter.

We strongly believe that the current system in California could benefit greatly from an envelope-based early warning algorithm such as the one presented in this thesis. In future work, we plan to use the VS location and magnitude estimations along with a Bayesian grid search to propose a better stand-alone algorithm for earthquake early warning systems. We described RCA as a process that uses the prediction made by the Decision Module to create predicted waveform envelopes. Then, these predicted envelopes are compared with the observed ones and the classification of the prediction, i.e., okay-, over, or under-prediction, is obtained. In this way RCA works as a supplementary unit for an early warning system; it does not make assessments independently from other algorithms that are already in the system. In fact, RCA depends on other algorithms that are making predictions. In future work, however, RCA could evolve into a "stand-alone" earthquake early warning algorithm that is able make predictions of source parameters such as location, origin time, and magnitude of earthquakes independently. The Decision Module would continue to aggregate all the independently made predictions (by several algorithms) into one probability (degree of plausibility). The extended RCA would involve a Ground Motion Envelope Predictor Algorithm (GMEP) that uses a grid search method. Due to the nature of the problem, the

grid would be in a hyper-dimensional space with axes *latitude, longitude, magnitude,* and *origin time*. The spatial section of the grid would be stretched over the entire surface of California as increments of *latitude* and *longitude* coordinates. Time limitations and available computational power would impose constraints regarding the grid fineness. On the other hand, we could make use of the fact that seismic stations do not get triggered at the same time due to vibrations as a guide to give varying weights to grid solutions that favor the first triggered stations. Moreover, a statistical study of past earthquake locations could be used as "prior" knowledge in GMEP. Note that the original RCA uses a Bayesian classification scheme in which any type of prior knowledge can be aggregated easily. Note also that any GMEP calculation made in the past can be used as a prior for the current ones; this sequential updating capability is an important aspect of Bayesian inference.

Before the work presented in this thesis, there had not been any algorithm designed to check the accuracy of the California early warning system predictions in real-time. This fact may be true for other earthquake early warning systems in the world. In addition to that, we proposed a new paradigm in which our earthquake early warning algorithm depends on finding envelope fits. This is a substantial contribution to earthquake early warning systems compared with the old paradigm in which the algorithms depend on pick times and amplitudes.

Appendix A

Virtual Seismologist Envelope Equations

The following is a portion from Georgia B Cua's PhD Thesis.(Cua 2005)

The Virtual Seismologist models the observed ground motion envelope as a combination of P-wave, S-wave, and ambient noise envelopes. These envelopes are combined using the following formula:

$$E_{observed}(t) = \sqrt{E_P^{\,2}(t) + E_S^{\,2}(t) + E_{ambient}^{\,2}} + \varepsilon \qquad \text{(A.1)}$$

where

$E_{observed}(t)$ = envelope of observed ground motion
$E_P(t)$ = envelope of P-wave
$E_S(t)$ = envelope of S-wave and later-arriving phases
$E_{ambient}$ = ambient noise at the site
ε = difference between predicted and observed envelopes

Ambient noise at a site is modeled as a constant. The P- and S-wave envelopes are described by five parameters for each: a rise time (t_{rise_P},t_{rise_S}), an amplitude (A_P,A_S), a duration (Δt_P,Δt_S), and two decay (γ_P,γ_S), (τ_P,τ_S) parameters. Therefore, an observed ground motion can be described by eleven envelope parameters.

$$E_{i,j}(t)=\begin{cases} 0, t<T_i \\ \dfrac{A_{i,j}}{t_{rise_{i,j}}}(t-T_i), T_i \le t < T_i + t_{rise_{i,j}} \\ A_{i,j}, T_i + t_{rise_{i,j}} \le t < T_i + t_{rise_{i,j}} + \Delta t_{i,j} \\ A_{i,j}\dfrac{1}{\left(t-T_i-t_{rise_{i,j}}-\Delta t_{i,j}+\tau_{i,j}\right)^{\gamma_{i,j}}}, t \ge t < T_i + t_{rise_{i,j}} + \Delta t_{i,j} \end{cases} \quad \text{(A.2)}$$

where

i = P-,S-wave
T_i = P-,S-wave arrival times
j = horizontal and vertical ground motions

For further details, see (Cua 2005).

Appendix B

Multiple Window Approach

Kurtosis is usually calculated by starting from the most current value of the distribution (test function in our case). We then go back a predetermined window length of samples in time and calculate the excess kurtosis (which is kurtosis minus 3) and slide the window as more data arrive in real-time. We used a novel idea: instead of one window length, we use multiple different window lengths (ranging from 10 seconds to 100 seconds with 10-second increments, i.e., 10 different sliding windows) for kurtosis computation and linearly add the results up.

Notice in Figure B.1 that if you have only one window, you cannot tell if the outlier is at the current end or in the past because you will see a spike in kurtosis calculation for each location. A multiple windows approach guarantees that the outlier will be detected at the same time in real-time by all of the windows, and kurtosis and skewness values will add up at that location whereas any indication of the outlier in the past relative to different window lengths will be at different locations and therefore suppressed by summation of the results (Figure B.2).

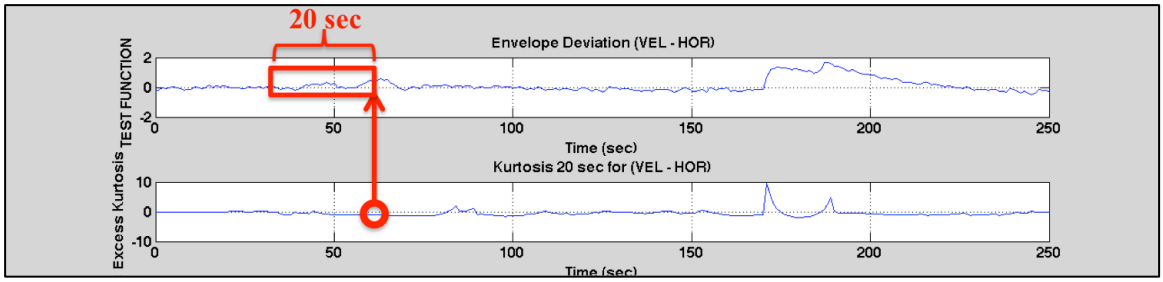

Figure B.1: Kurtosis computation example with a single window of length 20 seconds. In order to compute the kurtosis value highlighted with a circle on the bottom plot, we go to the corresponding value of the test function shown on the top of the figure. Starting from that value, we go back 20 seconds in time and use equation (2.3) on the values within this 20 seconds long window. Notice there are two spikes on the bottom of the figure indicating another outlier for kurtosis computations.

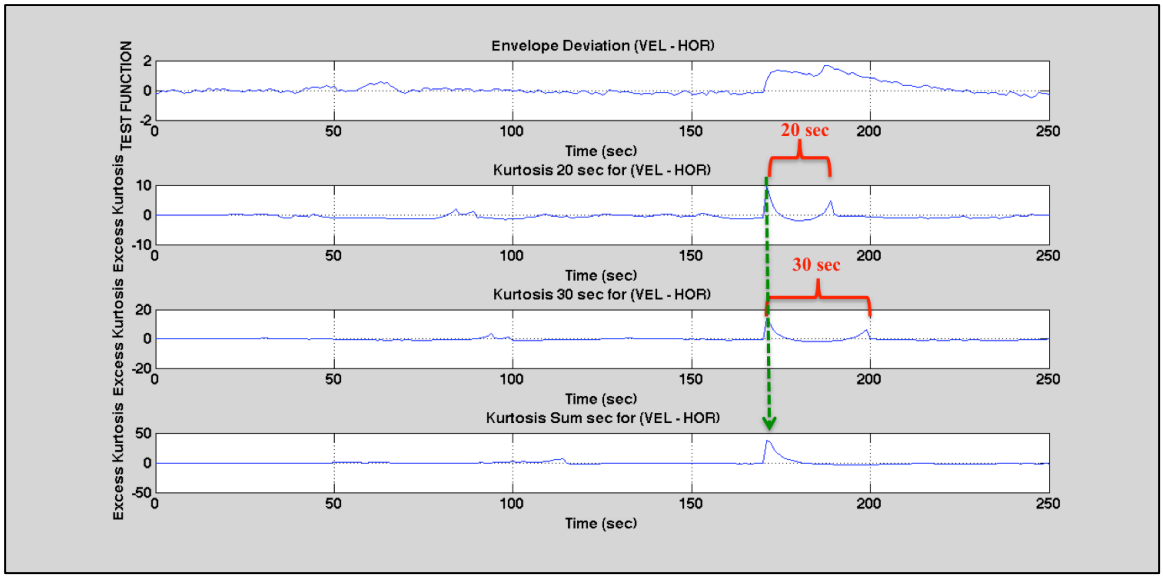

Figure B.2: Multiple windows approach for a kurtosis computation with two different window lengths and their sum. A sample test function (top), kurtosis for the test function using a 20 seconds long window (second row), kurtosis for the same test function using a 30 seconds long window (third row), and linear sum of second and third row (bottom).

The same procedure, namely summation of the running windows of differing lengths starting from the most current time and going back, is used for skewness calculation in real-time as well.

Appendix C

Non-probabilistic Classifications

C.1 Classifications: Least Squares

The following theory is based on the concepts described in Chapter 4, Linear Models for Classification in Bishop, 2006.

We start with least squares, which has a fairly straightforward implementation phase. Training runs are not computationally highly demanding and parameters can be computed using a closed form solution as shown below.

As mentioned above, we have three classes ($K=3$), and we describe each class by its own linear model

$$y_k(\mathbf{x}) = \mathbf{w}_k{}^T\mathbf{x} + w_{k0} \tag{C.1}$$

where $k=1,\cdots,K$.

Let us write these linear models in a more compact form

$$\mathbf{y}(\mathbf{x}) = \tilde{\mathbf{W}}^T\tilde{\mathbf{x}} \tag{C.2}$$

where $\tilde{\mathbf{W}}$ is the matrix of dimension $(D+1)\times K$. Column vector $\tilde{\mathbf{w}}_k = \left[w_{k0}, \mathbf{w}_k{}^T\right]^T$ comprises the columns of $\tilde{\mathbf{W}}$, and the augmented input vector is defined as $\tilde{\mathbf{x}} = \left[1, \mathbf{x}^T\right]^T$. Note that we augment the input vector with a dummy input $x_0 = 1$ for later convenience. Classification is made by choosing the maximum y_k for a given new input $\mathbf{x}$, and assigning it to C_k, that is, $\mathbf{x} \in C_j \Leftrightarrow j = \arg\max_k \left|\tilde{\mathbf{w}}_k\tilde{\mathbf{x}}\right|$.

In the least squares approach, we determine the unknown matrix of parameters, $\tilde{\mathbf{W}}$, by minimizing a sum-of-squares error function. Our training set consists of the input-output pairs $\left\{\mathbf{x}_n, \mathbf{t}_n\right\}$ where $n = 1,2,\cdots,N$. We can write our input and output values in a more compact form as $\tilde{\mathbf{X}}$ and $\mathbf{T}$, where the row vectors $\tilde{\mathbf{x}}_n^T$ and $\mathbf{t}_n^T$ are the n^{th} rows for $\tilde{\mathbf{X}}$ and $\mathbf{T}$, respectively. Note that for the purpose of having less clutter in the mathematical expressions, we can redefine our parameters as ones without a "tilde", i.e., ~ symbol without loss of generality. That is to say, from now on, unless specified otherwise, $\tilde{\mathbf{W}} \to \mathbf{W}$, $\tilde{\mathbf{X}} \to \mathbf{X}$, and $\tilde{\mathbf{x}} \to \mathbf{x}$. Then, we can write the sum-of-squares error function as

$$E_D\left(\mathbf{W}\right) = \frac{1}{2} Tr\left\{\left(\mathbf{XW} - \mathbf{T}\right)^T \left(\mathbf{XW} - \mathbf{T}\right)\right\} \tag{C.3}$$

We set the derivative of $E_D\left(\mathbf{W}\right)$, with respect to $\mathbf{W}$, to zero, and then we obtain the following closed form solution for $\mathbf{W}$

$$\mathbf{W} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{T} \tag{C.4}$$

Then, our compact discriminant function becomes

$$\mathbf{y}\left(\mathbf{x}\right) = \mathbf{W}^T\mathbf{x} \tag{C.5}$$

An input feature vector with acceleration values is in the form

$$\mathbf{x} = \begin{bmatrix} 1 \\ \frac{d}{dt}\left(\text{Kurtosis of Horizontal Acceleration}\right) \\ \frac{d}{dt}\left(\text{Kurtosis of Vertical Acceleration}\right) \\ \frac{d}{dt}\left(\text{Skewness of Horizontal Acceleration}\right) \\ \frac{d}{dt}\left(\text{Skewness of Vertical Acceleration}\right) \end{bmatrix} \tag{C.6}$$

An input feature vector in velocity or displacement would be similar to (C.6) with acceleration replaced by velocity or displacement respectively.

We set up our least squares solution such that $k = 1$ represents okay-prediction, $k = 2$ represents over-prediction, and $k = 3$ represents under-prediction classes, similar to probabilistic classifications. At the end of our computations, we obtained the following parameter values, i.e., (C.7), (C.8) and (C.9), and the confusion matrices, i.e., Table C.1 to Table C.3; Table C.1 is constructed using (C.7), Table C.2 is constructed using (C.8), and Table C.3 is constructed using (C.9).

C.2 Discussion of Results in Tables C.1 to C.3

In general, actual and predicted classes for okay-prediction class agree well (more than 94 percent) in all of the different ground motion parameters influenced by different frequency contents: acceleration, velocity, and displacement. The least squares approach shows relatively better performance when we use the first half of the data set for training, and the second half for validation. However, when we swap these data sets, that is, when we use the second half as training and the first half as validation sets, the performance decreases significantly. This fact suggests that we might want to use a more rigorous

validation scheme such as the *leave-one-out* method. However, we do not resort to that technique because when we apply a full Bayesian treatment in the upcoming sections, our approach will use only the training data and so it will not need a separate data set for validation. This is particularly useful because reserving part of the data set for validation causes a waste of valuable training data. In addition to that, cross-validation inherently means multiple training runs as opposed to a single training run that is required in a full Bayesian treatment. Moreover, the error function (C.3) is not robust to outliers and so a better alternative is desirable in order to achieve better performance.

$$\mathbf{W}_{acceleration} = \begin{bmatrix} 0.897 & 0.057 & 0.045 \\ -0.001 & 0.001 & 0.001 \\ -0.002 & 0.001 & 0.001 \\ 0.001 & -0.006 & 0.005 \\ 0.002 & -0.005 & 0.003 \end{bmatrix} \quad \text{(C.7)}$$

Table C.1: Confusion matrices for least squares classification using acceleration data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	97.7	0	2.3
	Over	38	62	0
	Under	30.8	0	69.2
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	95.6	0	4.4
	Over	18.4	81.6	0
	Under	15.2	0	84.8
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	98.2	0	1.8
	Over	63.2	36.8	0
	Under	47.2	0	52.8
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	96.9	0	3.1
	Over	40.8	59.2	0
	Under	31.2	0	68.8

$$\mathbf{W}_{velocity} = \begin{bmatrix} 0.875 & 0.079 & 0.046 \\ -0.003 & 0.002 & 0.001 \\ -0.001 & 0.0003 & 0.001 \\ 0.007 & -0.013 & 0.005 \\ -0.002 & -0.001 & 0.003 \end{bmatrix} \quad \text{(C.8)}$$

Table C.2: Confusion matrices for least squares classification using velocity data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	97.9	0.1	2
	Over	42	58	0
	Under	35.6	0	64.4
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	94.8	0.2	5
	Over	24.8	75.2	0
	Under	12.8	0	87.2
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	98.4	0	1.6
	Over	77.6	22.4	0
	Under	57.6	0	42.4
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	96.6	0.1	3.3
	Over	51.2	48.8	0
	Under	35.2	0	64.8

$$\mathbf{W}_{displacement} = \begin{bmatrix} 0.835 & 0.092 & 0.073 \\ -0.005 & 0.004 & 0.001 \\ -0.0004 & 0.0003 & 0.0001 \\ 0.014 & -0.023 & 0.008 \\ -0.004 & -0.001 & 0.005 \end{bmatrix} \quad \text{(C.9)}$$

Table C.3: Confusion matrices for least squares classification using displacement data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	98.4	0.6	1
	Over	54.4	45.6	0
	Under	45.6	0	54.4
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	97	1	2
	Over	48	52	0
	Under	23.2	0	76.8
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	98.8	0.6	0.6
	Over	72.8	27.2	0
	Under	68	0	32
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	97.9	0.8	1.3
	Over	60.4	39.6	0
	Under	45.6	0	54.4

C.3 Classifications – Linear Discriminant Analysis (LDA)

The following theory is based on the concepts described in Chapter 4 of Georgia Cua's PhD thesis (2004): Creating the Virtual Seismologist: Developments in Ground Motion Characterization and Seismic Early Warning.

In linear discriminant analysis, we define a number of groups or classes, and then find a linear combination of some input vectors (similar to the least squares input) that will maximize the separation between these groups. Unlike the least squares classification, the parameter vector is not augmented, so we do not need to augment the input feature vector either. Therefore, we determine a 4-dimensional parameter vector, $\mathbf{w}$. This goal is achieved by maximizing the *among-class* to *within-class* variance ratio. That means that the parameter vector is able to discriminate the data belonging to different classes as much as possible while reducing the deviation within a specific class.

Similar to the least squares classification, we have $K = 3$ classes with N_k observations in each class, i.e., total number of observations, $N = \sum_{k=1}^{3} N_k$. Then, we define the class means as

$$\mathbf{m}_k = \frac{1}{N_k} \sum_{n \in k}^{N} \mathbf{x}_n \tag{C.10}$$

which is simply the mean of all of the observations in a class. Using (C.10), we can define the *within-class* covariance matrix as

$$\mathbf{S}^{(k)} = \frac{1}{N_k - 1} \sum_{n \in k}^{N} \left(\mathbf{x}_n - \mathbf{m}_k \right) \left(\mathbf{x}_n - \mathbf{m}_k \right)^T \tag{C.11}$$

We need a sum of the *within-class* covariance matrices for all classes, which is defined as

$$\mathbf{S}_p = \frac{1}{N-K}\sum_{k=1}^{K}\left(N_k - 1\right)\mathbf{S}^{(k)} \tag{C.12}$$

Next, we define the *among-class* covariance matrix as

$$\mathbf{S}_a = \frac{1}{K-1}\sum_{k=1}^{K} N_k\left(\mathbf{m}_k - \mathbf{m}\right)\left(\mathbf{m}_k - \mathbf{m}\right)^T \tag{C.13}$$

where we implicitly define a general mean, $\mathbf{m}$, for the input values of all classes as

$$\mathbf{m} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n = \frac{1}{N}\sum_{k=1}^{K} N_k\mathbf{m}_k \tag{C.14}$$

As we mentioned above, we are looking for the vector $\mathbf{w}$ such that the linear combination $\mathbf{w}^T\mathbf{x}$ will assign $\mathbf{x}$ to one of the classes by reducing the variability of the data within a class and increasing separation of the data from one class to another. We can find such $\mathbf{w}$ by maximizing the *among-class* to *within-class* variance ratio:

$$\lambda = \frac{\mathbf{w}^T\mathbf{S}_a\mathbf{w}}{\mathbf{w}^T\mathbf{S}_p\mathbf{w}} \tag{C.15}$$

Now we take the derivative of (C.15) with respect to $\mathbf{w}$ and equate it to zero, i.e., $\frac{\partial\lambda}{\partial\mathbf{w}} = 0$. Then, we obtain

$$\lambda\mathbf{w}^T\mathbf{S}_p - \mathbf{w}^T\mathbf{S}_a = 0 \tag{C.16}$$

Noting the symmetry of $\mathbf{S}_a$ and $\mathbf{S}_p$, let us take the transpose of (C.16)

$$\begin{aligned} &\lambda\mathbf{S}_p\mathbf{w} - \mathbf{S}_a\mathbf{w} = 0 \\ &\qquad\Rightarrow \lambda\mathbf{S}_p\mathbf{w} = \mathbf{S}_a\mathbf{w} \end{aligned} \tag{C.17}$$

We assume $\mathbf{S}_p$ is invertible, then we obtain

$$\lambda\mathbf{w} = \mathbf{S}_p^{-1}\mathbf{S}_a\mathbf{w} \tag{C.18}$$

Note that (C.18) is an eigenvalue problem. The parameter vector $\mathbf{w}$ we are seeking is an eigenvector of $\mathbf{S}_p^{-1}\mathbf{S}_a$; the separating measure is therefore the largest eigenvalue of $\mathbf{S}_p^{-1}\mathbf{S}_a$.

Using the same class and data definitions as the least squares classification section, we followed the procedure described in this section. The following eigenvalues and eigenvectors are computed using the entire data set for training. However, additional confusion matrices are provided for cases where the data set is divided into two: a training data set and a validation data set, as done in the least squares classification section. The eigenvalues of $\mathbf{S}_p^{-1}\mathbf{S}_a$ for acceleration input are given below

$$\begin{aligned} \lambda_1 &= 1726.463 \\ \lambda_2 &= 853.553 \\ \lambda_3 &= \text{-}4.475\text{e-}13 \\ \lambda_4 &= 7.326\text{e-}13 \end{aligned} \tag{C.19}$$

Since λ_1 is the largest value, the eigenvector associated with it, $\mathbf{w}_{acceleration}$, gives the desired linear combination values for classification.

$$\mathbf{w}_{acceleration} = \begin{bmatrix} 0.027 \\ 0.015 \\ \text{-}0.822 \\ \text{-}0.568 \end{bmatrix} \tag{C.20}$$

Figure C.1 shows the histograms obtained using $\mathbf{w}_{acceleration}$ as the discriminant function combination values. The decision boundaries are located at the middle of the mean of two adjacent class histograms. Therefore, the decision boundary between okay- and over-prediction classes is located at 27.6195 and the decision boundary between okay- and under-prediction classes is located at -20.5939.

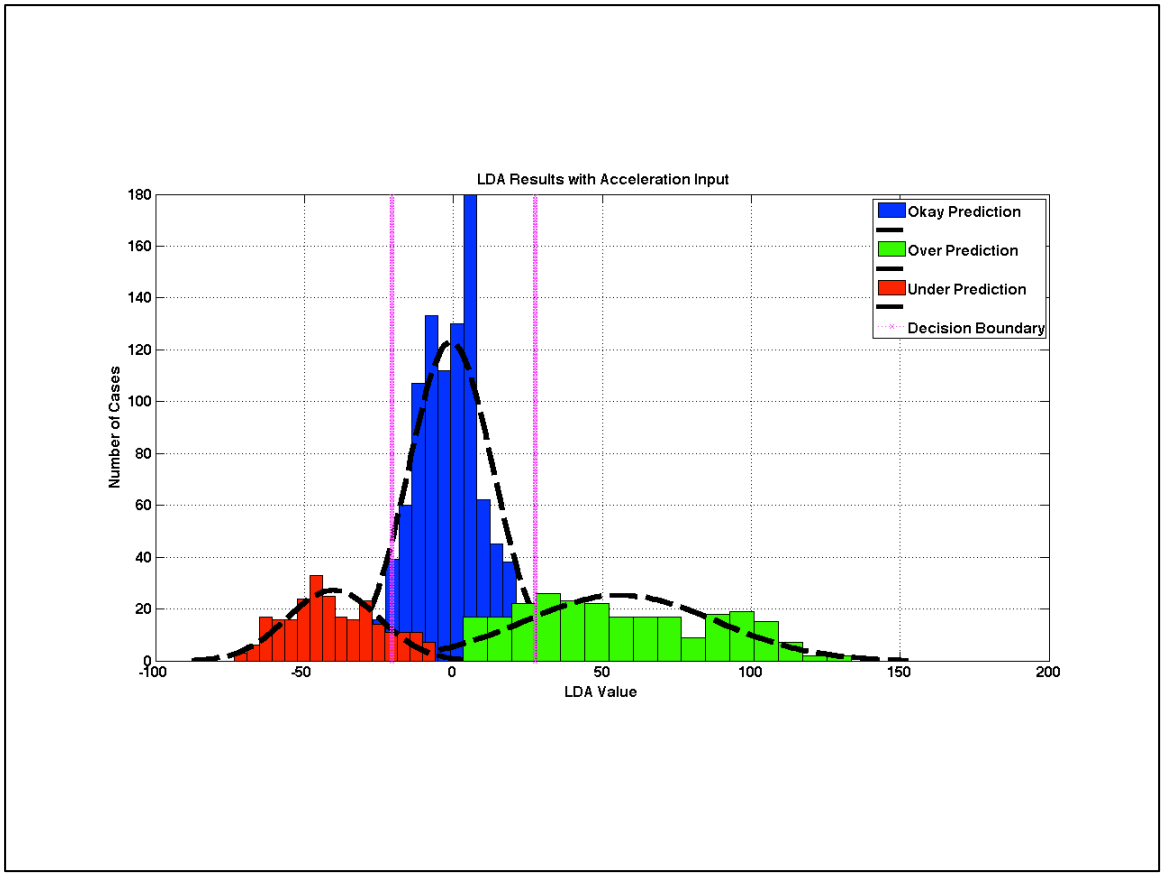

Figure C.1: Histogram for all three classes obtained using LDA with acceleration values only.

Table C.4: Confusion matrix for LDA using acceleration data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	91.6	2.7	5.7
	Over	22	78	0
	Under	14.4	0	85.6
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	90.8	2	7.2
	Over	17.6	82.4	0
	Under	12	0	88
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	82.4	4.4	13.2
	Over	28.8	71.2	0
	Under	16.8	0	83.2
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Acceleration**	Okay	Over	Under
	Okay	86.6	3.2	10.2
	Over	23.2	76.8	0
	Under	14.4	0	85.6

The eigenvalues of $\mathbf{S}_p^{-1}\mathbf{S}_a$ for velocity input are given below

$$\begin{aligned}\lambda_1 &= 1591.126\\ \lambda_2 &= 744.510\\ \lambda_3 &= \text{-}2.089\text{e-}13\\ \lambda_4 &= 4.057\text{e-}13\end{aligned} \tag{C.21}$$

Similar to acceleration results, the largest value is λ_1. The eigenvector associated with it, $\mathbf{w}_{velocity}$, gives the best linear combination.

$$\mathbf{w}_{velocity} = \begin{bmatrix} 0.0413 \\ \text{-}0.024 \\ \text{-}0.973 \\ \text{-}0.226 \end{bmatrix} \tag{C.22}$$

Figure C.2 shows the histograms obtained using $\mathbf{w}_{velocity}$ as the discriminant function combination values. The decision boundaries are located at the middle of the mean of two adjacent class histograms. Therefore, the decision boundary between okay- and over-prediction classes is located at 15.9058 and the decision boundary between okay- and under-prediction classes is located at -20.4801.

The eigenvalues of $\mathbf{S}_p^{-1}\mathbf{S}_a$ for displacement input are given below

$$\begin{aligned}\lambda_1 &= 1380.411\\ \lambda_2 &= 485.286\\ \lambda_3 &= \text{-}1.115\text{e-}13\\ \lambda_4 &= 7.361\text{e-}13\end{aligned} \tag{C.23}$$

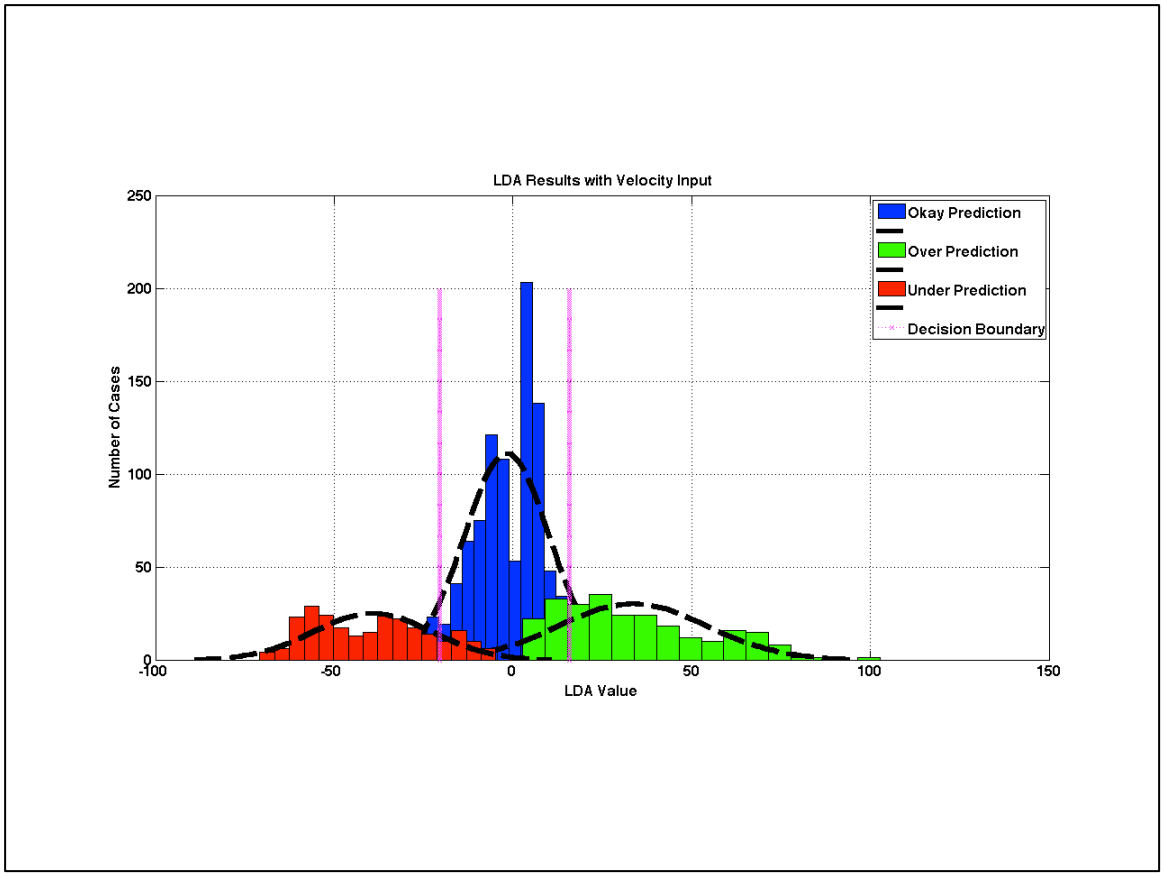

Figure C.2: Histogram for all three classes obtained using LDA with velocity values only.

Table C.5: Confusion matrix for LDA using velocity data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	90.6	3.1	6.3
	Over	22.4	77.6	0
	Under	16	0	84
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	89.6	1.6	8.8
	Over	18.4	81.6	0
	Under	12	0	88
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	75	14.2	10.8
	Over	20	80	0
	Under	24.8	0	75.2
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Velocity**	Okay	Over	Under
	Okay	82.3	7.9	9.8
	Over	19.2	80.8	0
	Under	18.4	0	81.6

The largest value is λ_1, and the eigenvector associated with it, $\mathbf{w}_{displacement}$, gives the best linear combination.

$$\mathbf{w}_{displacement} = \begin{bmatrix} -0.046 \\ -0.003 \\ 0.967 \\ 0.252 \end{bmatrix} \tag{C.24}$$

Figure C.3 shows the histograms obtained using $\mathbf{w}_{displacement}$ as the discriminant function combination values. The decision boundaries are located at the middle of the mean of two adjacent class histograms. Therefore, the decision boundary between okay- and over-prediction classes is located at -8.4600 and the decision boundary between okay- and under-prediction classes is located at -14.4728.

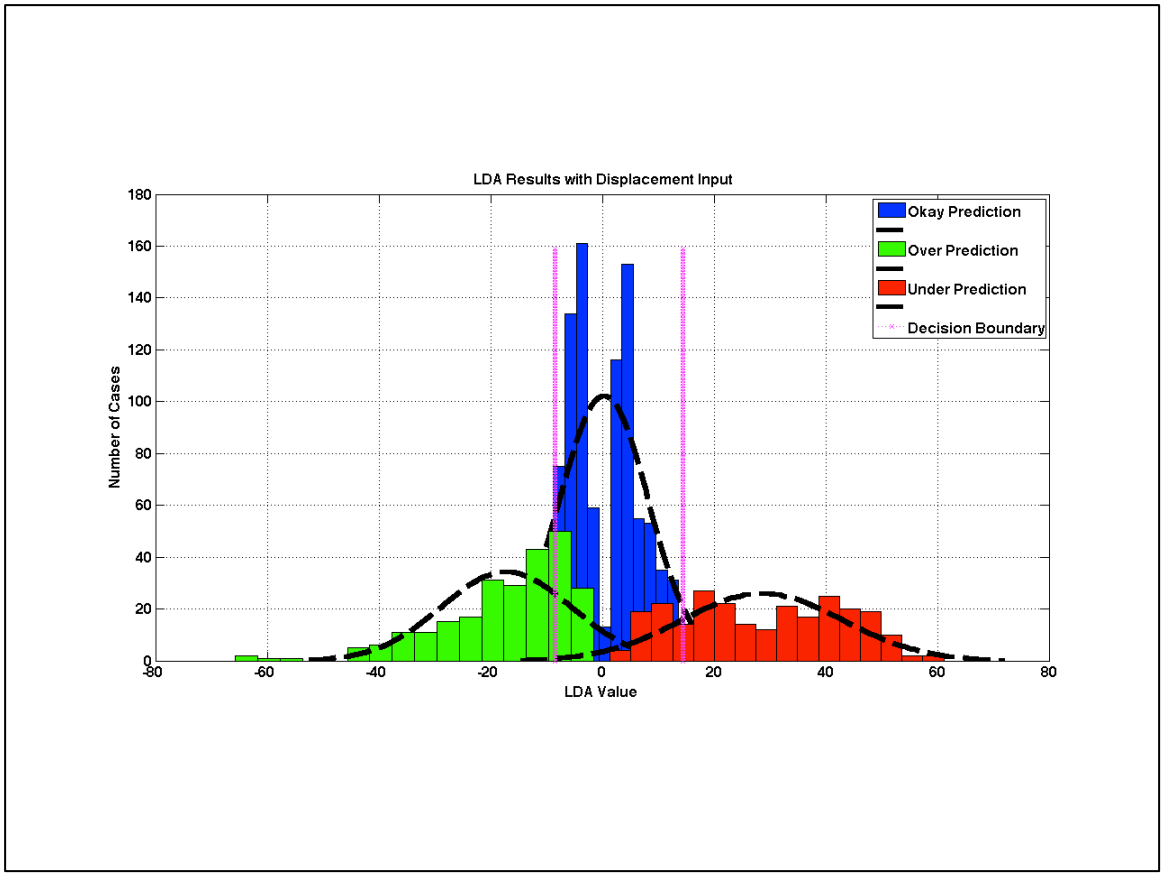

Figure C.3: Histogram for all three classes obtained using LDA with displacement values only.

Table C.6: Confusion matrix for LDA using displacement data.

ALL DATA	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	88.8	8	3.2
	Over	25.2	74.8	0
	Under	20.4	0	79.6
FIRST HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	89	4	7
	Over	25.6	74.4	0
	Under	16	0	84
SECOND HALF	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	55.8	34.8	9.4
	Over	24.8	75.2	0
	Under	34.4	0	65.6
AVERAGE OF CROSS VALIDATIONS	PREDICTED CLASSES			
ACTUAL CLASSES	**Displacement**	Okay	Over	Under
	Okay	72.4	19.4	8.2
	Over	25.2	74.8	0
	Under	25.2	0	74.8

C.4 Discussion of Results in Tables C.4 to C.6

Starting with a four-dimensional input feature vector $\mathbf{x}$ and projecting it onto a scalar value, i.e., one-dimension, as this section, results in loss of information. In fact, this reduction in dimensionality may cause significant overlap among classes (Bishop 2006). In the linear discriminant analysis, we computed such a parameter vector $\mathbf{w}$, which led to a projection that better separates classes. This enhancement in the performance is seen in the confusion matrices provided for LDA. Compared to the performance of least squares classification, all channels of ground motion input showed consistent improvement.

When we look at the individual ground motion parameters, the best performance is provided by acceleration even though the difference from the performance provided by velocity is not significantly improved. The worst performance is observed when we used the second half of the displacement data for training and tested the results using the first half for validation. Although the algorithm for this particular case performs well as far as the actual over- and under-prediction classes are concerned, more than 30 percent of okay-prediction classes are classified as over-prediction.

Bibliography

"The facilities of IRIS Data Services, and specifically the IRIS Data Management Center, were used for access to waveforms, related metadata, and/or derived products used in this study. IRIS Data Services are funded through the Seismological Facilities for the Advancement of Geoscience and EarthScope (SAGE) Proposal of the National Science Foundation under Cooperative Agreement EAR-1261681."

"http://www.mathworks.com/matlabcentral/fileexchange/28803-read-and-write-multiplexed-miniseed-file/content/rdmseed.m."

"NCEDC (2014), Northern California Earthquake Data Center. UC Berkeley Seismological Laboratory. Dataset. doi:10.7932/NCEDC.".

"rdmseed.m."

"readsac.m."

"SCEDC (2013): Southern California Earthquake Center. Caltech.Dataset. doi:10.7909/C3WD3xH1."

Allen, R. M. and H. Kanamori (2003). "The potential for earthquake early warning in southern California." Science **300**(5620): 786-789.

Beck, J. L. (2010). "Bayesian system identification based on probability logic." Structural Control and Health Monitoring **17**(7): 825-847.

Beck, J. L. and L. S. Katafygiotis (1998). "Updating models and their uncertainties. I: Bayesian statistical framework." Journal of Engineering Mechanics **124**(4): 455-461.

Beck, J. L. and K.-V. Yuen (2004). "Model selection using response measurements: Bayesian probabilistic approach." Journal of Engineering Mechanics **130**(2): 192-203.

Behr, Y., et al. (2013). "Evaluation of Real-Time and Off-Line Performance of the Virtual Seismologist Earthquake Early Warning Algorithm in Switzerland." EGU General Assembly.

Behr, Y., et al. (2015). "Anatomy of an Earthquake Early Warning (EEW) Alert: Predicting Time Delays for an End‐to‐End EEW System." Seismological Research Letters.

Bishop, C. M. (2006). Pattern recognition and machine learning, springer.

Bishop, C. M. and M. E. Tipping (2003). "Bayesian regression and classification." Nato Science Series sub Series III Computer And Systems Sciences **190**: 267-288.

Böse, M., et al. (2014). CISN ShakeAlert: An earthquake early warning demonstration system for California. Early Warning for Geological Disasters, Springer**:** 49-69.

Böse, M., et al. (2009). "Real‐time testing of the on‐site warning algorithm in southern California and its performance during the July 29 2008 Mw5. 4 Chino Hills earthquake." Geophysical Research Letters **36**(5).

Böse, M., et al. (2009). "A new trigger criterion for improved real-time performance of onsite earthquake early warning in Southern California." Bulletin of the Seismological Society of America **99**(2A): 897-905.

Böse, M., et al. (2012). "Rapid Estimation of Earthquake Source and Ground‐Motion Parameters for Earthquake Early Warning Using Data from a Single Three‐Component Broadband or Strong‐Motion Sensor." Bulletin of the Seismological Society of America **102**(2): 738-750.

Böse, M. and T. H. Heaton (2010). "Probabilistic prediction of rupture length, slip and seismic ground motions for an ongoing rupture: implications for early warning for large earthquakes." Geophysical Journal International **183**(2): 1014-1030.

Böse, M., et al. (2012). "Real-time finite fault rupture detector (FinDer) for large earthquakes." Geophysical Journal International **191**(2): 803-812.

Clayton, R. W., et al. (2012). "Community seismic network." Annals of Geophysics **54**(6).

Clinton, J. F. and T. H. Heaton (2002). "Potential advantages of a strong-motion velocity meter over a strong-motion accelerometer." Seismological Research Letters **73**(3): 332-342.

Cua, G. and T. Heaton (2008). "Characterizing average properties of southern California ground motion amplitudes and envelopes." Bull. seism. Soc. Am.

Cua, G. B. (2005). Creating the Virtual Seismologist: developments in ground motion characterization and seismic early warning, California Institute of Technology.

DeCarlo, L. T. (1997). "On the meaning and use of kurtosis." Psychological methods **2**(3): 292.

Eguchi, R. T., et al. (1994). "Real-time earthquake hazard assessment in California; the early post-earthquake damage assessment tool and the Caltech-USGS broadcast of earthquakes."

Ellsworth, W. L. and T. H. Heaton (1994). "Real-time analysis of earthquakes: Early-warning systems and rapid damage assessment." Sensors-the Journal of Applied Sensing Technology **11**(4): 27-33.

Given, D. D., et al. (2014). Technical implementation plan for the ShakeAlert production system: an Earthquake Early Warning system for the West Coast of the United States, US Geological Survey.

Hartzell, S. H. and T. H. Heaton (1985). "Teleseismic time functions for large, shallow subduction zone earthquakes." Bulletin of the Seismological Society of America **75**(4): 965-1004.

Heaton, T. H. (1990). "Evidence for and implications of self-healing pulses of slip in earthquake rupture." Physics of the Earth and Planetary Interiors **64**(1): 1-20.

Heaton, T. H. (2007). "Will performance-based earthquake engineering break the power law?" Seismological Research Letters **78**(2): 183-185.

Hoshiba, M. and S. Aoki (2015). "Numerical Shake Prediction for Earthquake Early Warning: Data Assimilation, Real‐Time Shake Mapping, and Simulation of Wave Propagation." Bulletin of the Seismological Society of America.

Hoshiba, M. and T. Ozaki (2014). Earthquake Early Warning and Tsunami Warning of the Japan Meteorological Agency, and Their Performance in the 2011 off the Pacific Coast of Tohoku Earthquake ({M} _ {{\ mathrm {w}}} 9.0). Early Warning for Geological Disasters, Springer**:** 1-28.

Housner, G. W. and T. Vreeland Jr (1965). "The analysis of stress and deformation."

Kanamori, H., et al. (1997). "Real-time seismology and earthquake hazard mitigation." Nature **390**(6659): 461-464.

Kuyuk, H., et al. (2015). "Automatic earthquake confirmation for early warning system." Geophysical Research Letters **42**(13): 5266-5273.

Langet, N., et al. (2014). "Continuous Kurtosis‐Based Migration for Seismic Event Detection and Location, with Application to Piton de la Fournaise Volcano, La Réunion." Bulletin of the Seismological Society of America **104**(1): 229-246.

Lawson, C. L. and R. J. Hanson (1974). Solving least squares problems, SIAM.

Minson, S. E., et al. (2015). "Crowdsourced earthquake early warning." Science Advances **1**(3): e1500036.

Oh, C. K., et al. (2008). "Bayesian learning using automatic relevance determination prior with an application to earthquake early warning." Journal of Engineering Mechanics.

Richter, C. F. (1958). "Elementary seismology."

Shearer, P. M. (2009). Introduction to seismology, Cambridge University Press.

Tipping, M. E. (2001). "Sparse Bayesian learning and the relevance vector machine." The journal of machine learning research **1**: 211-244.

Vincenty, T. (1975). "Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations." Survey review **23**(176): 88-93.

Wald, D. J., et al. (1999). "TriNet "ShakeMaps": Rapid generation of peak ground motion and intensity maps for earthquakes in southern California." Earthquake Spectra **15**(3): 537-555.

Yamada, M. (2007). Early warning for earthquakes with large rupture dimension, California Institute of Technology.

Yamada, M. and T. Heaton (2008). "Real-time estimation of fault rupture extent using envelopes of acceleration." Bulletin of the Seismological Society of America **98**(2): 607-619.

Yamada, M., et al. (2007). "Real-time estimation of fault rupture extent using near-source versus far-source classification." Bulletin of the Seismological Society of America **97**(6): 1890-1910.

Zhang, H. and J. Malik (2005). "Selecting shape features using multi-class relevance vector machine." EECS Department, University of California, Berkeley UCB/EECS-2005-6.