Time Series Change Detection: Algorithms for Land Cover Change

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Shyam Boriah

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor Of Philosophy

April, 2010

Acknowledgements

The graduate school experience at Minnesota has been personally rewarding and challenging, and very enjoyable thanks to a wonderful group of people whom I have gotten to know during my time here. I have been extremely fortunate to be surrounded by friends and colleagues from whom I have learned so much, and whose support has helped immensely in completing my PhD.

First, and foremost, I would like to express my gratitude to my academic advisor, Prof. Vipin Kumar. It has been an honor and a privilege to work with him and learn from him. I would also like to thank my committee members, Professors Jaideep Srivastava, Dan Boley and Joe Knight for their valuable advice on my dissertation work. Finally, I would like to thank the excellent Computer Science Department staff.

To my family, and friends around the world, I cannot do justice to your significance in a few words: you are amazing.

Dedication

To my grandparents and my parents

ABSTRACT

The climate and earth sciences have recently undergone a rapid transformation from a data-poor to a data-rich environment. In particular, climate and ecosystem related observations from remote sensors on satellites, as well as outputs of climate or earth system models from large-scale computational platforms, provide terabytes of temporal, spatial and spatio-temporal data. These massive and information-rich datasets offer huge potential for advancing the science of land cover change, climate change and anthropogenic impacts.

One important area where remote sensing data can play a key role is in the study of land cover change. Specifically, the conversion of natural land cover into human-dominated cover types continues to be a change of global proportions with many unknown environmental consequences. In addition, being able to assess the carbon risk of changes in forest cover is of critical importance for both economic and scientific reasons. In fact, changes in forests account for as much as 20% of the greenhouse gas emissions in the atmosphere, an amount second only to fossil fuel emissions.

Thus, there is a need in the earth science domain to systematically study land cover change in order to understand its impact on local climate, radiation balance, biogeochemistry, hydrology, and the diversity and abundance of terrestrial species. Land cover conversions include tree harvests in forested regions, urbanization, and agricultural intensification in former woodland and natural grassland areas. These types of conversions also have significant public policy implications due to issues such as water supply management and atmospheric CO_2 output.

In spite of the importance of this problem and the considerable advances made over the last few years in high-resolution satellite data, data mining, and online mapping tools and services, end users still lack practical tools to help them manage and transform this data into actionable knowledge of changes in forest ecosystems that can be used for decision making and policy planning purposes. In particular, previous change detection studies have primarily relied on examining differences between two or more satellite images acquired on different dates. Thus, a technological solution that detects

global land cover change using high temporal resolution time series data will represent a paradigm-shift in the field of land cover change studies.

To realize these ambitious goals, a number of computational challenges in spatio-temporal data mining need to be addressed. Specifically, analysis and discovery approaches need to be cognizant of climate and ecosystem data characteristics such as seasonality, non-stationarity/inter-region variability, multi-scale nature, spatio-temporal autocorrelation, high-dimensionality and massive data size. This dissertation, a step in that direction, translates earth science challenges to computer science problems, and provides computational solutions to address these problems. In particular, three key technical capabilities are developed: (1) Algorithms for time series change detection that are effective and can scale up to handle the large size of earth science data; (2) Change detection algorithms that can handle large numbers of missing and noisy values present in satellite data sets; and (3) Spatio-temporal analysis techniques to identify the scale and scope of disturbance events.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The climate and earth sciences have recently experienced a rapid transformation from a data-poor to a data-rich environment. In particular, climate and ecosystem related observations from remote sensors on satellites, as well as outputs of climate or earth system models from large-scale computational platforms, provide terabytes of temporal, spatial and spatio-temporal data. These massive and information rich datasets offer huge potential for advancing the science of land cover change, climate change and anthropogenic impacts.

One important area where remote sensing data can play a key role is in the study of land cover change. Specifically, the conversion of natural land cover into human-dominated cover types continues to be a change of global proportions with many unknown environmental consequences. For example, studies [36, 57] have shown that deforestation has significant implications for local weather, and in places such as the Amazon rainforest, cloudiness and rainfall are greater over cleared land than over intact forest (see Figure 1.1). In addition, being able to assess impacts of changes in forest cover on the carbon cycle is of critical importance for both economic and scientific reasons (e.g., using forests for economic carbon sink management, and studying natural and anthropogenic impacts on ecosystems).

For example, the contribution of greenhouse gases from deforestation is one of the most uncertain elements of the global carbon cycle. Without information about global deforestation patterns and fluxes, it is difficult to model the global carbon budget (the balance of exchanges of carbon between reservoirs; see Figure 1.2) and to predict the

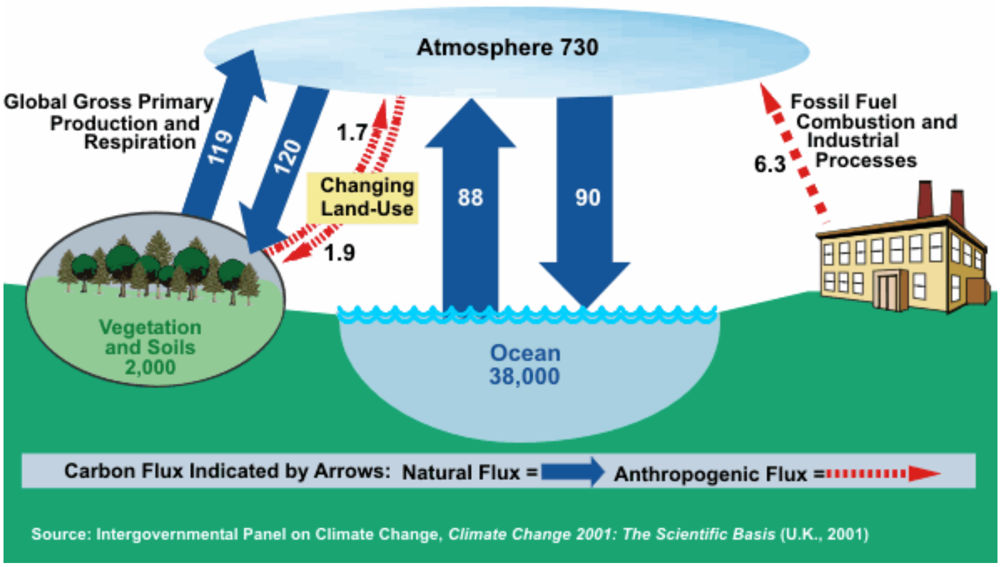
Source: NASA Earth Observatory.

Figure 1.1: Deforestation changes local weather. Cloudiness and rainfall can be greater over cleared land (image right) than over intact forest (left).

effects of climate change [94]. Recent research [102] suggests that the role forests play in regulating global climate is larger than previously thought and will likely become even more important as alternative carbon sinks become saturated while forests continue to act as sinks throughout a century of climate change. In fact, changes in forests account for as much as 20% of the greenhouse gas emissions in the atmosphere, an amount second only to fossil fuel emissions [48].

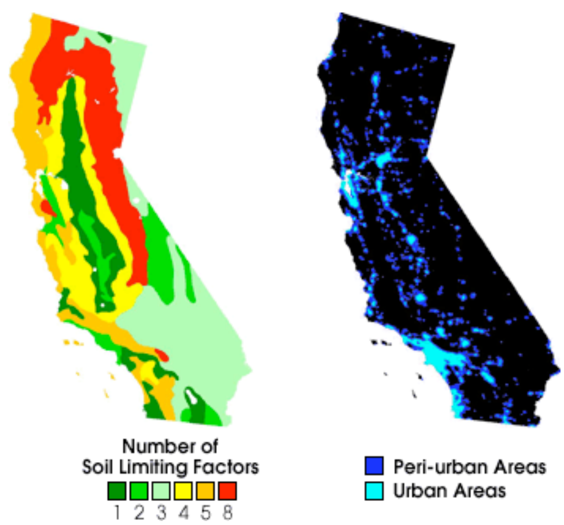
Thus, there is a need in the earth science domain to systematically study land cover change in order to understand its impact on local climate, radiation balance, biogeochemistry, hydrology, and the diversity and abundance of terrestrial species. Land cover conversions include tree harvests in forested regions, urbanization, and agricultural intensification in former woodland and natural grassland areas. These types of conversions also have significant public policy implications due to issues such as water supply management and atmospheric CO_2 output. For example, Charbonneau and Kondolf [23] found that in California between 1984 and 1990, over half of all new irrigated farmland put into production was of lesser quality than prime farmland taken out of production by urbanization (e.g., see Figures 1.3 and 1.4).

However, in spite of the importance of this problem and the considerable advances



Source: NOAA Earth System Research Laboratory.

Figure 1.2: Carbon is exchanged between the atmosphere, land and oceans.



Source: Marc Imhoff, NASA GSFC, and Flashback Imaging Corporation, Ontario, Canada.

Figure 1.3: This pair of images shows the suitability of California soils for farming on the left, and urban areas on the right. The Great Central Valley, where the state's best soils are and most of America's fresh vegetables are grown, is becoming increasingly urbanized. (1: best soil 8: worst soil).

Image Source: Google Maps.

Figure 1.4: This figure shows an example of a land cover change: conversion of farmland to housing subdivisions in Sacramento, CA. This set of pixels was detected by our algorithm (see Chapter 6 and [13] for more details). The image on the top shows an aerial photograph from early 2008, while the image on the bottom shows the same region more recently (late 2009). It is apparent that the area converted to housing has already expanded in this short time.

Domains	Algorithmic Techniques

Statistics
Signal Processing
Control Theory
Computer Graphics & Vision (curve segmentation)
Network Intrusion Detection
Fraud Detection (telecommunications, credit cards, etc.)
Health Care (Statistical Surveillance)
Industrial Processes (process control and quality control)
Land Cover Change
	Statistical Parameter Change [96] [30] [98]
	Segmentation
	    Linear Model [58] [73] [56]
	    Polynomial Model [49]
	    Wavelet Model [114]
	Predictive [45] [106]
	Subspace Approach [91]
	Anomaly Detection [19] [126] [65] [20]

Table 1.1: A sample of the domains and techniques that address the time series change detection problem. A detailed discussion of these techniques is provided in Chapter 3.

made over the last few years in high-resolution satellite data, data mining, and online mapping tools and services, end users still lack practical tools to help them manage and transform this data into actionable knowledge of changes in forest ecosystems that can be used for decision making and policy planning purposes. In particular, previous change detection studies have primarily relied on examining differences between two or more satellite images acquired on different dates [33]. These approaches have a number of limitations; for example, changes that occur outside the image acquisition windows are not mapped, it is difficult to identify when the changes occurred, information about ongoing landscape processes cannot be derived, and they are inherently unsuited for application at global scale. Furthermore, only time series approaches provide information about land cover dynamics that are necessary to quantitatively assess the carbon impact of land cover changes [102]. However, though time series change detection has been studied in a wide variety of domains and a number of techniques have been proposed (see Table 1.1), these techniques are not suitable for the land cover change detection problem primarily because, they are not scalable or they are unable to take advantage of the inherent structure present in earth science data. Thus, a technological solution that detects global land cover change using high temporal resolution time series data will represent a paradigm-shift in the field of land cover change studies.

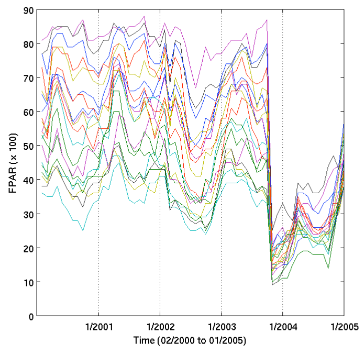
To realize these ambitious goals, a number of computational challenges in spatio-temporal data mining need to be addressed. Specifically, analysis and discovery approaches need to be cognizant of climate and ecosystem data characteristics such as seasonality, non-stationarity/inter-region variability, multi-scale nature, spatio-temporal autocorrelation, high dimensionality and massive data size.

This dissertation, a step in that direction, translates earth science challenges to computer science problems, and provides computational solutions to address these problems. In particular, three key technical capabilities are developed: (1) Algorithms for time series change detection that are effective and can scale up to handle the large size of earth science data; (2) Change detection algorithms that can handle large numbers of missing and noisy values present in satellite data sets; and (3) Spatio-temporal analysis techniques to identify the scale and scope of disturbance events.

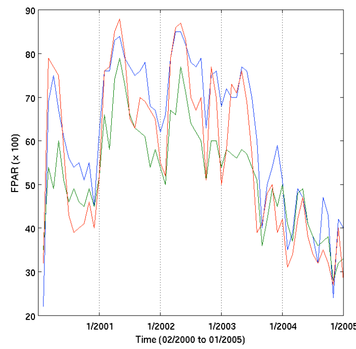
1.1 Contributions

1.1.1 Scalable Time Series Change Detection Algorithms

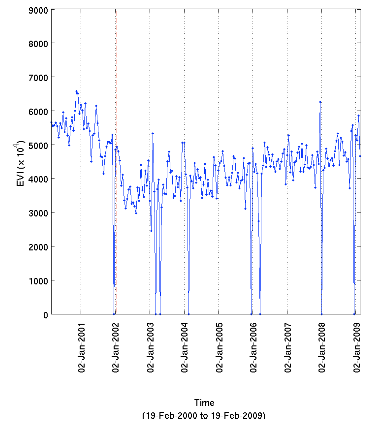
Global change detection studies using remotely sensed images are challenging because of the need to balance data set size with the high spatial and temporal resolutions required to describe fine scale landscape changes with high temporal precision. For example, a global image library based on ten years of bi-weekly 1 km MODIS images would require the storage and processing of over 230 images having 160 million pixels each (the total data size for just one variable is $\sim$75GB). In addition to the broad range of events (fires, droughts, logging, floods, insect damage, etc.) and patterns that need to be discovered, inter-region variability, natural variability (due to weather and climate variation), and the variety of vegetation types pose additional challenges. Figure 1.5 shows a few examples of the wide variety of changes. This dissertation develops algorithms that address many of the challenges, including scaling to massive global data, handling non-stationarity, and exploiting seasonality and spatio-temporal information in the data. In addition to satellite noise, another major source of variability is due to differences in local weather, vegetation type, and terrain, and geography. The algorithms developed in this dissertation are also robust to such variability. This work is unique because it is among the first high temporal resolution schemes for land cover change detection and



(a) This group of time series corresponds to a set of pixels that were burned in a forest fire in Southern California. The time period before the fire shows how much variability can be present even in a small region.



(b) This set of pixels in Western Peru was affected by a drought. Droughts often severely damage vegetation and a recovery is not seen for many years.



(c) This time series shows logging activity and corresponds to a pixel in Mendocino County, California. A logging pattern typically involves a gradual shift over a longer period (relative to fires).

Figure 1.5: Examples of different kinds of land cover changes.

because of the high accuracy with which changes are detected.

1.1.2 Techniques Robust to Missing and Noisy Data

Satellite time series data are subject to frequent contamination due to clouds, haze, pixel geometry and other factors. Contamination is particularly problematic in the tropics, where cloud cover predominates for many months of the year (Figure 5.2 illustrates this issue further). We develop robust change detection algorithms to work with limited data rather than a full time series; i.e., the algorithms are robust to missing values, albeit at the cost of losing some temporal resolution in identifying the time period of change. Figure 1.6 shows an example of a time series with a large number of missing values. This pixel was detected by our robust algorithm but not by other time series change detection algorithms. Another example with noisy values is shown in Figure 1.7.

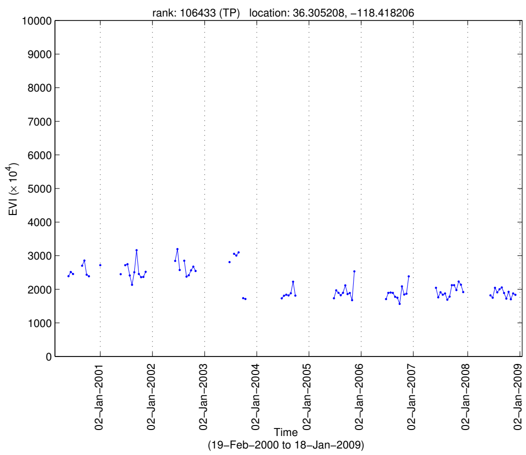


Figure 1.6: This figure shows a time series with many missing values that experienced a change in mid-2003. However, most change detection algorithms would not be able to detect this change due to the large number of missing values (shown as gaps).

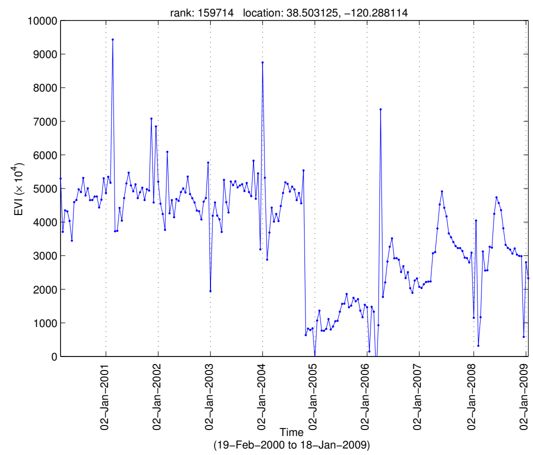


Figure 1.7: This figure shows a time series with noisy values that experienced a change in late 2004. The noisy values across the time series would throw off many change detection algorithms.

1.1.3 Spatio-temporal Analysis for Event Identification

Land cover changes often span several pixels. We develop spatio-temporal analysis techniques to identify the scale and scope of disturbance events. For example, the spatial dependence and autocorrelation properties of vegetation data can be exploited to discover that several neighboring pixels in a region are part of the same forest fire.

1.2 Scope

The general change detection problem encompasses a tremendous variety of application domains and data types. Change detection from spatio-temporal data influences domains ranging from health care to aircraft safety. In this dissertation, we restrict the scope of our evaluation to spatio-temporal data sets in the earth science domain. In particular, the focus of this work is on the land cover change detection problem (Figure 1.4 shows an illustrative example). However, potential applications of the techniques discussed in the dissertation to other domains will be presented in Chapter 9.

1.3 Outline

This dissertation is organized as follows:

Chapter 2 introduces the land cover change detection problem and discusses the necessary background. A discussion of prior work addressing this problem is also provided in this chapter.

Chapter 3 surveys previous work in the area of time series change detection. This chapter presents a unified taxonomy of techniques defined in a number of different domains and discusses the major approaches.

Chapter 4 presents novel algorithms for land cover change detection. The chapter discusses key aspects of algorithm design, the underlying assumptions and their performance implications.

Chapter 5 addresses the problem of incomplete and contaminated data that is frequently encountered with earth science data. Various approaches for dealing with missing and noisy data are discussed.

Chapter 6 provides illustrative examples of the kind of land cover changes that can be found using a time series change detection algorithm.

Chapter 7 presents a quantitative evaluation comparing the proposed algorithms, their variations, and other well-known algorithms using real-world data.

Chapter 8 discusses techniques for the problem of spatio-temporal analysis for event identification.

Chapter 9 summarizes the work in this dissertation, discusses potential applications to other domains, and considers future research directions.

Chapter 2

Land Cover Change Detection — Background and Related Work

Broadly speaking, the land cover change detection problem is one of detecting when the land cover at a given location has been converted from one type to another. Examples include the conversion of forested land to barren land (possibly due to deforestation or a fire), grasslands to golf courses and farmland to housing developments (e.g. see Figure 1.4). When the vegetation cover decreases, e.g., due to fires, droughts, insect damage, logging, etc., the change event is usually called a *disturbance*. There are a number of factors that make this a challenging problem, particularly due to the nature of the underlying data.

Much prior work in change detection has often been performed by comparing two or more images of different dates, or "snapshots," to determine the land cover differences between them, which makes them inherently unsuitable for global analysis. Hence, most studies have focused on relatively small areas or have described changes in specific categories of interest [123, 120, 68, 32]. Recent work has used time series approaches to detect land cover changes [87, 106]. Time series based change detection has significant advantages over the comparison of snapshot images of selected dates because it takes into account information about the temporal dynamics of landscape changes. Specifically, detection of changes is based on observations of the landscape over time rather than the differences between two or more images collected on different dates. Therefore additional parameters such as the rate of the change (e.g., a sudden forest fire vs. gradual logging) or the extent, speed, and pattern of regrowth can be derived and used for change detection and characterization.

The general problem of change detection in time series data has been extensively studied in the fields of statistics, signal processing and control theory. However, most of the previous work in time series change detection is not suitable for the land cover change detection problem because these techniques do not take advantage of the inherent spatio-temporal structure present in earth science data. There has been some work in time series-based land cover change detection, but this work has had limited success. For example, the change detection algorithm used to generate the Burned Area Product (a well-known MODIS data set) performs very poorly in parts of North America such as California (see Figure 4.1). In addition, such algorithms and products are geared towards specific kinds of changes (such as fires), and are not capable of detecting the broad set of changes that we seek to address (such as those due to deforestation, floods, droughts and insect infestations).

In this chapter, we will introduce the land cover change detection problem as well as the necessary background. We will also discuss related techniques from the earth science domain. Since time series change detection is an important problem in its own right, prior work for this broader problem will be discussed in Chapter 3.

2.1 Background

2.1.1 Earth Science Data

Earth science data typically consist of global snapshots of measurement values for a number of variables (e.g., temperature, pressure, and precipitation) collected for all land and sea surfaces (see Figure 2.1). These variables are either observations of different climate and ecosystem variables, e.g., precipitation, sea level pressure (SLP), sea surface temperature (SST), or the result of model predictions, such as net primary production (NPP). The data observations come from satellites, or from historical climate data, which include older satellite systems, as well as data from land-based weather stations and ocean-based ship and buoy measurements.

The earth ecience data in this dissertation consist of snapshots of measurement

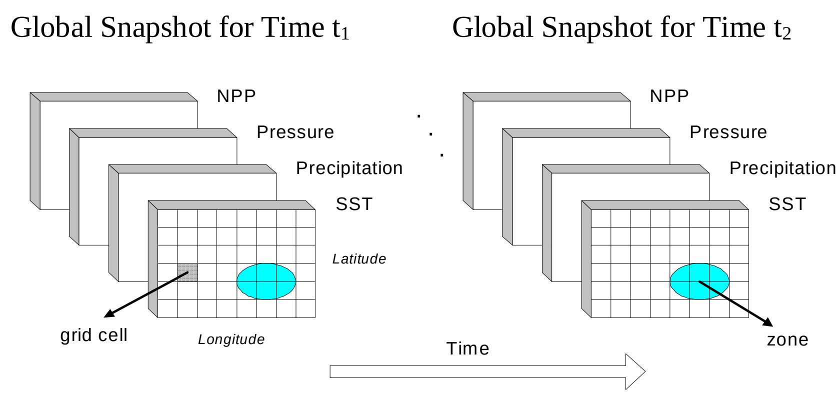


Figure 2.1: A simplified view of the problem domain.

values for a vegetation-related variable collected for all land surfaces (Figure 2.2 shows a sample of the data). The data observations come from NASA's Earth Observation System (EOS) [2] satellites and the data sets are distributed through the Land Processes Distributed Active Archive Center (LP DAAC) [3].

The specific vegetation-related variables used in this dissertation are the enhanced vegetation index (EVI) and fraction of photosynthetically active radiation (FPAR) products measured by the moderate resolution imaging spectroradiometer (MODIS). EVI is a vegetation index which essentially serves as a measure of the amount and "greenness" of vegetation at a particular location. It represents the greenness signal (area-averaged canopy photosynthetic capacity), with improved sensitivity in high biomass cover areas [63]. MODIS algorithms have been used to generate the EVI index at 250 meter spatial resolution from February 2000 to the present. In the work presented in this dissertation, the temporal coverage of the data is from the time period February 2000—January 2009. FPAR, also a vegetation index that is very similar to EVI, is at 4 km resolution and covers the time period February 2000—January 2005.

2.1.2 Notation

We introduce the following notation in order to describe the land cover change detection algorithms, the properties of the underlying data sets, and the evaluation methodology in this chapter. Let D be a data set with N land locations. Broadly speaking, there are typically two types of data that are used for land cover change detection: bitemporal

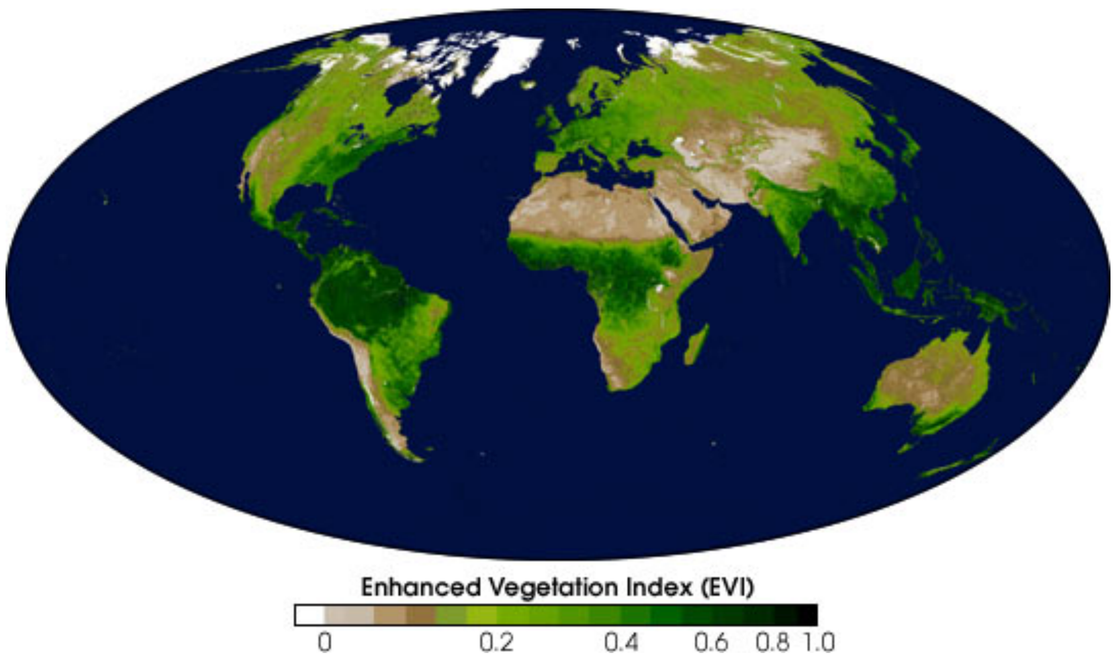


Image Courtesy: MODIS Land Group, Alfredo Huete and Kamel Didan, University of Arizona.

Figure 2.2: The above MODIS Enhanced Vegetation Index (EVI) map shows the density of plant growth over the entire globe for October 2000. Very low values of EVI (white and brown areas) correspond to barren areas of rock, sand, or snow. Moderate values (light greens) represent shrub and grassland, while high values indicate temperate and tropical rainforests (dark greens).

(snapshots from two dates) and multi-temporal (data from multiple dates) time series. Since the nature of these two data types is very different, we will use the following distinct notation for each:

Bitemporal. Bitemporal data is typically represented as images from two different dates.

t_1	Date of the first image.
t_2	Date of the second image.
I_1	An $m \times n$ matrix representing the image at time t_1. Each entry (pixel intensity) in the matrix corresponds to an individual land location, i.e., $m \times n = N$.
I_2	An $m \times n$ matrix representing the image at time t_2.

Table 2.1: Notation for bitemporal land cover change detection.

Note that I_1 and I_2 can be multispectral, in which case they are three dimensional matrices. In order to keep the notation simple, we will not explicitly specify the third dimension (corresponding to spectra), but it is understood that when comparing the ij pixels from two images, each spectrum is compared individually.

Multi-temporal/Time Series. Each land location has a time series of length T. The time series for a location corresponds to T biweekly/monthly observations at that location.

2.2 Related Work

Change detection, in general, is an area that has been extensively studied in the fields of statistics [66], signal processing [50] and control theory [80]. However, most techniques from these fields are not well-suited for the massive high-dimensional spatio-temporal data sets from earth science. This is due to limitations such as high computational complexity and the inability to take advantage of seasonality and spatio-temporal autocorrelation inherent in earth science data.

S	The number of time steps corresponding to one year of data (we also call this the season length). For biweekly data $S = 23$, and $S = 12$ for monthly data.
Y	The number of years of data in the data set. In this dissertation, we will work with complete years by truncating trailing time steps. Thus $Y = \frac{T}{S}$.
n_i	An individual location.
n_{ij}	jth month of data for the location n_i.

Table 2.2: Notation for time series change detection.

Spatio-temporal statistical modeling is a rich domain with a vast literature (the book by Banerjee et al. [8] offers an in-depth overview). This approach to data analysis has strong statistical underpinnings and offers more inferential power (in terms of quantifiable uncertainty, etc.) relative to data mining approaches. The approach explicitly models spatial and temporal effects simultaneously, and offers natural extensions to include covariates. The ability to model hierarchical structures is also powerful. However, this approach is infeasible for data sets with hundreds of thousands of points, in spite of recent work on improving scalability [44, 69, 9].

The *Geographical Information Systems (GIS)* area offers many modeling tools for spatial data. These techniques handle spatial effects and covariates very effectively. However, the methods tend to focus on a small class of problems such as the relationship between variables at different scales (e.g. city-level vs. state-level).

Signal processing approaches are typically domain-specific, focusing on problems in real-time systems, communication signals, audio signals, etc.

In this section, we will discuss land cover change detection techniques from the earth science domain. The discussion will be organized around the taxonomy in Figure 2.3. Note that we primarily focus on the change detection aspect of the domain from a computational perspective. There are many other challenges and problems (such as pixel misregistration, etc.) studied in remote sensing and other domains that are outside the scope of this dissertation, though these aspects must be addressed in practice. We also do not discuss change detection methods that are primarily manual in nature such as GIS approaches and visual interpretation. Finally, techniques for the *characterization*

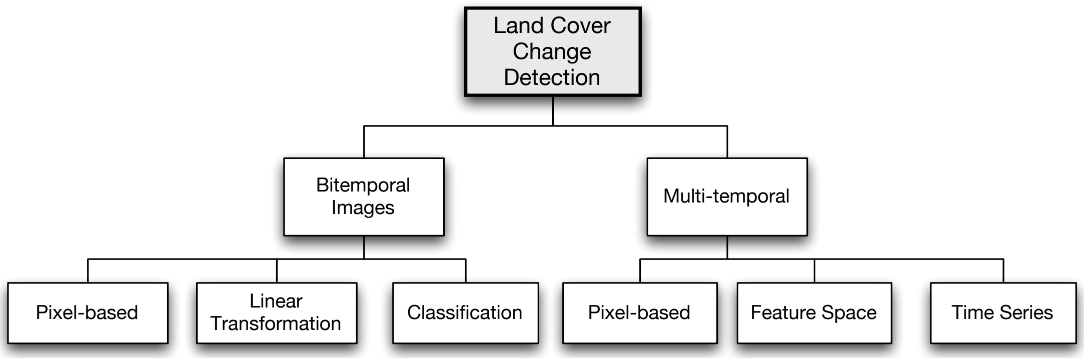


Figure 2.3: Taxonomy of techniques for land cover change detection.

problem (the problem of determining the type of land cover change) are not discussed here.

We describe each of the major approaches in the taxonomy in Figure 2.3, and provide a canonical algorithm for each approach. We also list the strengths and weaknesses of the approaches. In some cases we refer to specific examples, but in general we will not go into the details of specific techniques. There are a number of excellent surveys which discuss a large variety of techniques in great detail. We provide a summary of a few of these surveys at the end of this section.

2.2.1 Bitemporal Image Techniques

Change detection methods that are based on bitemporal images compare two snapshots I_1 and I_2 from different dates. The output is usually in the form of a *change mask* M, which is an $m \times n$ matrix with $M_{ij} = 1$ if pixel ij has changed and $M_{ij} = 0$ otherwise.

Simple Pixel-based Techniques

Simple pixel-based techniques [83, 88] are those that compare a given pixel ij at time t_1 directly with ij at t_2. These techniques compute M_{ij} by applying a function to the ij pixel of I_1 and I_2. Pixel-based techniques encompass a wide variety of methods ranging from simple differencing to more sophisticated regression approaches [72, 5, 92, 17, 90]. In the case of the simple differencing method, M_{ij} is computed as follows:

$$M_{ij} = \begin{cases} 1 & \text{if } (I_2 - I_1)_{ij} > \tau \\ 0 & \text{otherwise} \end{cases}$$

Here, τ is a user-specified threshold that denotes when a given pixel is considered to have changed. Ratioing is a similar pixel-based approach where instead of the difference, the ratio is compared against a threshold. A more sophisticated approach builds a regression model that computes a functional relationship between the bitemporal images and then compares the predicted pixel value with the pixel value from the first image. Simple pixel-based techniques can be effective for localized studies but their primary drawback is that they are extremely sensitive to noise. Their use for large regions is also limited by the issue that a global threshold is unlikely to perform well.

Linear Transformation

Linear transformation techniques [89, 83] are essentially based on encoding images as a matrix and examining structural properties of the matrix using techniques from linear algebra. The basic approach begins by representing images as columns of a matrix $\mathcal{I}$; i.e., I_1 and I_2 are converted into column vectors $\mathcal{I}_1$ and $\mathcal{I}_2$, and $\mathcal{I} \equiv [\mathcal{I}_1 \quad \mathcal{I}_2]$. The next step is to apply a linear transformation such as PCA (principal components analysis). PCA decomposes the matrix into eigenvectors and (sorted) eigenvalues. The larger eigenvalues and corresponding eigenvectors typically account for most of the variation in the matrix. The assumption is that the large eigenvalues will represent the majority unchanged portion of the image, while the smaller eigenvalues will represent the changed portions. A variation of the basic approach is to apply PCA to the *difference* image. In this case, the large eigenvalues correspond to the changed portions of the image. Other linear transformations that are frequently applied in land cover change detection include the KT transformation (also called the tasseled cap transformation), Gramm-Schmidt orthonormalization and the chi square transformation. Change detection methods based on linear transformations are much more robust than pixel-based approaches and can be applied across large regions, because they are less susceptible to noise and do not require a global threshold. However, their primary weakness is that it is difficult to determine which principal components correspond to change (versus other sources of natural variability) without manual inspection.

Classification

Change detection methods based on classification [105] train a classifier to learn models for different land cover types. The basic approach is to classify I_1 and I_2 separately, and to then compare the predicted class labels pixel-by-pixel. If the predicted class is different between I_1 and I_2 for pixel ij, then ij is considered to have changed. A variation of the basic approach is to use unsupervised classification (clustering) with multispectral data, where I_1 and I_2 are clustered separately and their cluster labels are compared [90]. A different lesser known classification approach is to try to *directly* classify the change pixels using a feature space that is a combination of features from I_1 and I_2. In this approach, the change pixels are expected to form their own classes/clusters which can be found by examining a few samples. Classification-based methods can be very powerful, especially when classifiers can be trained with high fidelity (this has been done with decision trees [43] and neural networks [42]). However, in general, because training data for large regions is manually labeled, training classifiers in this domain can be a cumbersome, time consuming and expensive process.

2.2.2 Multitemporal Techniques

Multitemporal techniques can have two views of the underlying data: (1) as a sequence of images of the same location from different dates, or (2) as a collection of time series with one time series corresponding to each pixel.

Multitemporal Pixel-based Techniques

Change vector analysis [81] is an extension of simple differencing to the multitemporal and multidimensional (corresponding to multiple spectra) space where several spectral channels are considered instead of only one. The basic approach is to construct vectors between successive time steps in the multidimensional space. The length of a given vector corresponds to the magnitude of the change between the respective time steps, while the direction corresponds to the nature of the change. While this class of techniques takes advantage of multispectral data, it suffers from the same limitations as simple differencing, viz. it requires thresholds and it is sensitive to noise.

Feature Space

Methods in this class of change detection techniques construct a high-dimensional feature space that is derived from a (possibly multispectral) time series. A classifier is then used to learn models in the high-dimensional feature space, and an approach similar to bitemporal classification is taken to detect changes. Techniques using this approach have shown recent promise [53], but training classifiers in the feature space approach can be cumbersome as in the case with bitemporal data. Additionally, because the time series is being converted to a feature space, these techniques are sensitive to the quality of the feature space being constructed, and there is potential for information loss in the transformation.

Time Series

Time series approaches detect changes primarily by using the temporal profile of the data for a particular pixel. However, some techniques also incorporate spatial information [87] into the time series change detection process, e.g. to normalize model comparisons by variation across space. There are a wide variety of time series approaches that can be taken to address the land cover change detection problem. The paper by Kucera et al. [78] discusses a CUSUM-based approach that was used for detecting burnt areas in Portugal.

Although time series approaches use the most information, and are potentially the most powerful class of techniques for change detection, their applicability for some problems is currently constrained by limitations of the underlying data. Specifically, satellite instruments which collect data at a high temporal frequency are also of relatively *coarse* spatial resolution. Therefore these techniques cannot be used to detect very small scale changes for which only a few images may be available. Additionally, time series approaches are of limited value when the time series is of short length (such as when a satellite has just been launched), and face challenges in detecting change at the endpoints of the time series.

2.2.3 Surveys

The survey by Lu et al. [85] provides a comprehensive discussion of a vast number of change detection algorithms and their applications. A discussion of eight major applications (ranging from forest fire detection to crop monitoring) of methods for land cover change detection and key references for each are presented. A detailed discussion goes into preprocessing techniques and other considerations in preparing data for change detection. The authors then categorize the literature into a detailed taxonomy (consisting of 31 categories) based on the approaches taken by change detection methods. The taxonomy also provides the level of complexity for each category. A detailed discussion of each category discusses the relative merits of the approaches. The authors also list a number of comparative studies of change detection techniques, discuss the feasibility of global scale change detection, and the complexities of threshold selection and accuracy assessment. Finally, recommendations for future work are provided.

Coppin et al. [33] provide a survey of land cover change detection that discusses the problem from three perspectives: (1) the types of variability and change events (land cover *conversion* vs. *modification*), (2) the preprocessing and data preparation process, and (3) categorizing and critically evaluating various change detection methods. They cover many of the same approaches as Lu et al. [85] but also discuss newer developments and discuss the most promising research directions. An older survey by Singh [116] provided much of the foundation in categorization of land cover change detection approaches.

Finally, Radke et al. [100] survey change detection techniques from the perspective of the image processing community. They consider a broader class of techniques in the vast image change detection literature, including the land cover change problem and other areas such as medical imaging. The generic image change detection problem is essentially to detect pixels that have changed (to produce the change mask) given a sequence of two- or three-dimensional images. Radke et al. [100] discuss some of the technical preprocessing steps such as geometric and intensity adjustment that are typically performed prior to change detection. Techniques are categorized into a number of major groups, and each category is discussed in detail. Spatial and temporal models are considered, but the approaches discussed are primarily geared towards images from cameras (image or video sequences) rather than general spatial/temporal data.

Evaluation approaches and principles are presented. A key issue that is discussed is the difficulty of obtaining comprehensive ground truth data in many change detection domains. As a result, the evaluation procedure is often qualitative.

Chapter 3

Time Series Change Detection

The last several decades have witnessed the increasing complexity of virtually all activities that make use of technology and more detailed monitoring of such activities. The problem of using the collected data to detect changes in the underlying processes has been called the change detection problem; other names include change point detection, breakpoint detection and concept drift. Other problems such as segmentation and partition regression will be discussed as subproblems of the general change detection problem. Note that some techniques, especially those from statistics, separate the *detection* (of whether a change exists) and *estimation* (of the location of the change point) tasks into separate tasks that are performed independently.

Time series data sets have always been ubiquitous; as such, the space of change detection techniques is vast and encompasses a variety of fields including statistics, signal processing and computer science. This chapter provides a discussion of major techniques for time series change detection in a unified context. We also provide a taxonomy of change detection techniques.

There have been numerous surveys and review papers that address time series change detection in the past few decades. One of the earliest surveys on change detection was performed by Shaban [113] in the form of an annotated bibliography. The problems considered are change of mean and change of regression model, covering an extensive number of papers published until 1980. A survey of frequentist and Bayesian techniques for this problem is provided by Zacks [127]. In particular, he considers the problem of testing the null hypothesis of no change against the alternative of at most one change, also called the AMOC (at most one change) setting. Zacks also considers the problem of estimating the location of change points. Wolfe and Schechtman [125] and Csörgő and Horváth [34] considered the AMOC setting, as well. They review several nonparametric procedures and also extensively evaluate the power of the tests using Monte Carlo studies. A comprehensive study of change detection techniques was performed by Krishnaiah and Miao [76]. Their paper includes a detailed study of frequentist and Bayesian change detection techniques. Some specific problems that are considered are the estimation of the intersection of two regression curves and estimation of the positions and number of change points in the multivariate setting. Lai [80] performed a detailed survey on sequential change detection techniques drawing on work in statistics as well as engineering. A recent survey was provided by Chen and Gupta [27] with a discussion that is organized around the characteristics of the procedures used, with most of the techniques based on the normal model. The authors discuss likelihood ratio and information theoretic approaches (based on AIC, etc.) for the mean change, variance change and multivariate mean change problems.

3.1 Problem Statements and Terminology

The wide variety of settings where change detection techniques are used has meant that there are many different (often opposing) application requirements. Therefore, while there is the single concept of time series change detection, there are several problems that address a number of scenarios. Below, we list some common change detection problem statements:

Offline Change Detection The goal is to detect when the characteristics of the time series has changed. Characteristics can range over a variety of properties. These could include functions such as the mean and variance or properties such as shape and trend. This is also called the *batch* or *fixed sample* setting.

Online Change Detection In this problem setting the data is coming in at a certain rate, not all the data can be stored at one time, and each instance may be examined only a fixed number of times. The goal is to detect changes with minimum delay, while minimizing false alarms, and with finite computational resources. Other terms used to describe this setting are *streaming*, *incremental* or *sequential* change detection.

Segmentation The segmentation problem is one of dividing a time series into homogeneous segments such that some predefined function (of coherence within segments and separation between segments) is optimized.

Top-k Discords The top-k discords problem is one of finding the k most unusual subsequences of a time series.

Smoothing and Forecasting The goals of time series smoothing and forecasting are essentially to remove discontinuities and predict future values, respectively.

Time Series vs. Sequences

There are subtle differences between data that is described as a time series and data described as a sequence. The term sequence is often used to refer to (typically discrete) data which has some sequential ordering. This could be, for example, a sequence of commands issued by a Linux user at the command prompt, or a protein sequence consisting of a chain of amino acids. On the other hand, a time series usually refers to *real-valued* data which are observed at a specific moment in time and thus accompanied by a *time stamp*; additionally the time stamp are usually regularly spaced (i.e. every second, etc.). Therefore, sequences can be viewed as a superset of time series. In our discussion, we will use the two interchangeably (since change detection is usually performed with real-valued data) unless the distinction is important. Note that there are a number of change detection techniques which consider a sequence of discrete values but our focus will be on *continuous* values.

Before we discuss the various approaches, we define terminology in Table 3.1 that will be used throughout this chapter.

3.2 Taxonomy

Time series change detection has been studied in a number of fields over time and there has been a large variety of techniques proposed. To help organize the discussion we will

CUSUM	Cumulative Sum
EWMA	Exponentially Weighted Moving Average
ARMA	Autoregressive Moving Average
ARIMA	Autoregressive Integrated Moving Average
LRT	Likelihood Ratio Test
MLE	Maximum Likelihood Estimation
DP	Dynamic Programming
HMM	Hidden Markov Model
PCA	Principal Components Analysis
SVD	Singular Value Decomposition

Table 3.1: Terminology used for time series change detection.

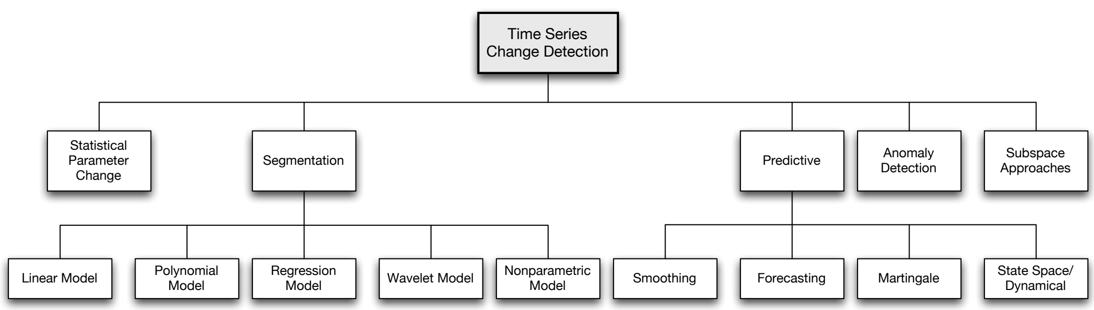


Figure 3.1: Taxonomy of time series change detection approaches.

use the taxonomy shown in Figure 3.1. There are, of course, other taxonomies that can be used to classify change detection techniques.

There are several important characteristics of change detection algorithms that cut across several of the categories in the taxonomy in Figure 3.1. We will discuss these characteristics in greater detail in subsequent sections, but briefly state them here:

- *online vs. offline*: Only some change detection techniques can be used in an online setting, while an online technique can always be used in an offline setting.

- *abrupt vs. gradual change*: In many problem settings, there tends to be a tradeoff between detecting abrupt and gradual changes. As a result, most techniques are optimized for one type of change or the other.

- *single vs. multiple change point*: Techniques may be able to detect only a single change point or multiple change points.
- *univariate vs. multivariate*: Most change detection techniques consider time series which consist of a single variable at each time step. However, there are some techniques which are able to handle a multivariate vector at each time step.
- *uniform sampling*: Most time series data sets consists of observations at equal spaced time intervals (the process is uniformly sampled). However, in some domains, such as finance, observations may be recorded only when a certain event occurs (e.g. the price of a security changes), in which case the time series is irregularly spaced.

3.3 Statistical Parameter Change Approaches

The earliest change detection techniques originated in manufacturing, where time series data was studied for process and quality control. In this setting, the time series was expected to follow a particular distribution and any significant departure from the distribution led to a shutdown of the manufacturing process so the the equipment could be corrected. In statistical terminology, the generating process has changed and therefore one is looking for a change in parameter of the distribution. We call such techniques parameter change approaches. The vast majority of work on parameter change approaches is from the statistical literature.

Parameter change detection approaches have the following characteristics:

- It is assumed that the data follows a parametric distribution.
- The sequence may contain a point where the parameter(s) of the underlying distribution change(s).
- The goal of the technique is to detect if such a change exists, and in some cases to estimate the value of the parameters before and after the change (such approaches often utilize maximum likelihood estimation (MLE) to estimate the parameters for the model).

Additionally, a characteristic that is shared by many parametric change detection techniques is that a *test procedure* and *test statistic* is defined for a time series and employed to determine whether a change point exists. In general, the hypothesis test consists of a null hypothesis that specifies no change and an alternative hypothesis that specifies some type of change, i.e.:

$$H_0 : \text{Change has not occured}$$

$$H_1 : \text{Change has occured}$$

Many of the earliest techniques [95] for change detection were parameter change approaches; these techniques were used for quality control in manufacturing processes and were studied in the area of statistical process control. The literature on parameter change approaches has since grown substantially and these techniques are now used in a variety of application areas. However, the bulk of the work shares many common characteristics [27]: they detect a single change point, assume a normal model, and use likelihood ratio tests (LRT). Maximization of the likelihood ratio is a computationally expensive procedure.

Page [95] proposed a sequential technique to detect a change in the mean. His technique uses a statistic based on cumulative sums (CUSUM), and a large volume of work continues to use CUSUM-type approaches. Chernoff and Zacks [30] used a Bayes estimator rather than an MLE-approach for the change in mean problem. Sen and Srivastava [111] proposed several techniques for detecting a change in mean based on various assumptions on the *a priori* knowledge of μ and σ^2. Picard [98] proposed offline procedures for this problem and addressed "edge" effects (this is when change points occur near the beginning or end of the time series and therefore very little data to make robust estimates); this work also included applications on oscillatory data (Canadian lynx data set) and economic data (German unemployment and IBM stock price data sets).

The change in variance problem was considered by Inclán and Tiao [66], Chen and Gupta [28] and Jandhyala et al. [67]. The CUSUM-based technique discussed in [66] detects multiple change points (using an iterative search procedure) and is applied to stock price data. The technique in [28] also detects multiple change points (using a top-down search procedure) and applied to stock price data. Their technique uses an information criterion rather than an LRT.

Hušková [64] presented test procedures for detecting changes in simple linear models. She addressed both the offline and online problems, and also applied the technique to the Nile River data set.

Sugiura and Ogden [117] proposed a variety of techniques that can detect *gradual* changes in addition to abrupt changes. The proposed techniques use LRT-type and Bayes-type tests. A data set that shows gradual change (# of vehicle accidents/year) was used to evaluate the techniques.

3.4 Segmentation Approaches

The goal of the segmentation problem is to partition the input time series into homogeneous segments (the subsequence within a segment is contiguous). Segmentation is essentially a special case of change detection since by definition, successive segments are not homogenous, which means there is likely to be a change point between the segments. The following are other problems which are variants of segmentation: partition regression, piecewise regression, segmented regression and k-segmentation. The segmentation approach to change detection is more appropriate for the *offline* setting since a global view of the time series is usually required to form segments. However, there are variations of some segmentation techniques that can be applied in the online setting.

The segmentation problem can be formulated in multiple ways, some of which may require user-defined input such as the number of segments or a threshold for *reconstruction* error. The key components of a segmentation technique are:

- A *stopping condition* which determines when new segments are no longer created. This is usually tied to the problem formulation.
- A *model* for the segments. When a model is fit to a segment it provides both a measure of homogeneity as well as a mechanism to compare segments. Note that the choice of evaluating the fit (sometimes called the cost of a segment) is critical to the output. The underlying model could be *linear*, *polynomial*, *regression*,

wavelet, or *nonparametric*. We will organize our discussion of change detection techniques around the model being used.

- A *search* strategy, such as top down, bottom up or DP, to discover the optimal segmentation. Search strategies will be discussed in detail in subsequent sections.

3.4.1 Linear Model

A *linear* model is essentially one which is based on a linear function of the data points within a segment. Examples include the mean, linear interpolation of the endpoints and the best least squares line (linear regression).

k-segmentation is a common problem formulation based on a linear model, where the goal is to partition the input into k segments, where k is defined by the user. Hawkins and Merriam [56] were among the first to address the k-segmentation problem. In their approach, the model for each segment is the mean and the cost is the sum of squared deviation from the mean. A dynamic programming search technique is presented that is optimal in the sense that it finds the k-segmentation with the smallest residual sum of squared error. Hawkins and Merriam [56] applied their technique to geological data, specifically, mechanical logs of boreholes.

Himberg et al. [58] also address the problem of k-segmentation. The proposed approach is multivariate in nature and the model for a segment is the mean vector. Although their paper discusses only the mean model, the framework is generic and other segment models can be used. The major contribution of the paper is that they extensively evaluate four search strategies (dynamic programming, top-down and two greedy) for the framework.

Keogh et al. [73] provided a survey of segmentation approaches that are based on linear models. They discuss three canonical algorithms which cover a large number of segmentation approaches: sliding window, top-down and bottom-up. Additionally, they assess the computational complexity of the canonical algorithms. An empirical comparison found the bottom-up algorithm to be generally superior and the sliding window algorithm to be the worst performer. Keogh et al. [73] also presented a new online algorithm which combines the bottom-up and sliding window algorithms. The algorithm proceeds as follows:

1. An initial buffer of some size (enough for 5-6 segments) is read from the stream.
2. The bottom-up algorithm is applied to the buffer, and the left-most segment is removed.
3. More data is read in from the stream by applying the sliding window algorithm for one segment.
4. The bottom-up algorithm is applied to the buffer, and the process repeats.

Further empirical evaluation found that the new algorithm performed similarly to the bottom-up algorithm (however, bottom-up is offline).

3.4.2 Polynomial Model

Guralnik and Srivastava [49] addressed the problem of event detection which they defined as the problem of "recognizing the change of parameters in the model, or perhaps even the change of the model itself, at unknown time(s)." Given a time series and a set of polynomial basis functions, the problem is to find a piecewise segmented model where each segment is represented by a basis function. Guralnik and Srivastava [49] presented both batch and incremental algorithms for this problem. An MLE is used to discover the split points and model selection is used to determine the best fit for a segment that can be described by the basis functions. Their technique is applied to highway traffic data for a large metropolitan area.

3.4.3 Regression Model

Segmentation using a regression model is essentially the following problem: it is assumed that the entire time series comes from the same regression model but with different parameters in different segments; the problem then is to discover the segments and estimate the parameters. Therefore, the analyst must decide what type of model to use, and in some cases how many segments are to be found. The natural measure of fit is the residual sum of squares for each segment.

Guthery [51] addresses the k-segmentation problem using a regression model. His technique is more efficient than a dynamic programming approach since it uses an efficient updating technique that does not require the regression model to be recomputed every time (thereby reducing the number of matrix inversions). The proposed approach is applied to two data sets: U.S. flour prices and the total number of telephones in the U.S.

Hawkins [54] also uses an efficient updating scheme but proposes a DP approach and a hybrid approach. The hybrid approach is a combination of bottom-up and top-down approaches which proceeds as follows: the time series is partitioned recursively as in a top-down approach, but after each iteration every pair of consecutive segments is examined to see if they should be merged. Additionally, since the technique uses MLE, this provides for selection of the number of segments if it is unknown. Another study by Hawkins [55] derives a DP-based solution for regression models from the general exponential family.

3.4.4 Wavelet and FFT Model

The fast fourier transform (FFT) and wavelets are decompositions/approximations of a time series in terms of a set of functions. FFTs are based on cosine and sine functions while wavelets utilize orthonormal functions. Wavelets are considered to be superior to FFTs, especially for functions that contain discontinuities. The *Daubechies* wavelet is a well-known class of wavelet functions that is known to have attractive properties. Wavelets are used extensively in signal processing due to their effectiveness in capturing the properties of a wide variety of functions. They have also been used in a number of data mining applications.

Ogden and Parzen [93] proposed a wavelet-based approach for smoothing a given time series. Their technique can be used for change-point detection since a jump in the wavelet coefficients usually signifies a change point.

Sharifzadeh et al. [114] presented a wavelet-based technique specifically for change-point detection that is geared toward query processing. They also discussed the notion of "degrees" of changes: if the segments are represented using polynomials and neighboring segments differ in the jth polynomial segment, the change point between the two segments is considered to be of degree j. The technique also supports query processing in that the user can get real-time responses to queries with different change thresholds once the underlying algorithm has been applied to the data set. The basic idea of the approach is to use wavelet "footprints" that can capture discontinuities in the data as well as the degree of change. The proposed technique is applied to synthetic data as well as a data set which consists of the oil and gas production rate in California oil wells.

3.4.5 Nonparametric Model

Segmentation techniques that use functions of the input time series as the model do not approximate the data with other functions. Rather, they utilize functions of time series (or subsequences thereof), such as the mean, which are *data-driven* models. We refer to such techniques as those based on nonparametric models.

Lowe [84] proposed a segmentation approach that is used to approximate curves with linear segments in the context of computer vision. The goal was to obtain simple features from pixel curves. Since a pixel curve can be viewed as a sequence, Lowe's technique is relevant in our context as well. The core component of the proposed technique is a measure of the significance of the fit of a segment to a portion of the curve. The significance is measured as a ratio of the length of the line segment to the maximum deviation of any point from the line. The algorithm proceeds by constructing a *tree* of increasingly fine approximations as follows:

1. The entire sequence is approximated with a segment and its significance is stored at the root of the tree.

2. The sequence is partitioned at the point where it has maximum deviation from the segment.

3. The process is repeated recursively with the two subsequences, until the subsequences are very short (of length 4 in [84]).

4. The final segmentation is obtained from the tree by examining the significance at each node: if the significance of either child node is greater than the parent, then the two subsegments are returned. Otherwise the single segment corresponding to the parent is returned.

3.5 Predictive Approaches

Predictive approaches to change detection are based on the assumption that one can learn a model for a portion of the input time series, and detect change based on deviation from the model. The underlying model can range from relatively simple smoothing models to more sophisticated filtering and state-space models. Due to the nature of the problem formulation, many predictive approaches are well-suited for the streaming data setting. Predictive approaches are frequently derived from *time series forecasting*, which is an important problem in time series analysis [16, 24].

3.5.1 Smoothing

Smoothing is a well-known statistical methodology that is often used to estimate missing observations and to mitigate the effects of noise. Common classes of smoothing methods are moving averages, kernel smoothers and filters. Smoothing approaches to change detection operate by building a smoothed model of previously observed data and comparing the prediction of the smoothed model against the observed data. Smoothing approaches typically have the following characteristics:

1. A combination of the current observation and previous observations is maintained as a statistic.

2. Previous observations are usually assigned a *weight* that is a function of how much time has passed since the data was observed.

3. Some smoothing approaches, such as those based on moving average models, are easy to implement and interpret.

Lucas and Saccucci [86] present a change detection procedure based on an exponentially weighted moving average smoothing model (first introduced by Roberts [103]) for data that is normally distributed. Let $\{Y_1, Y_2, Y_3, \ldots\}$ be an input time series. The technique is based on the following statistic, which is compared against control limits to determine when a change has occurred:

$$Z_i = \lambda Y_i + (1 - \lambda) Z_{i-1}, \quad 0 < \lambda \leq 1.$$

λ is a parameter which determines the weight on past observations, while Z_0 is set to the target value or an *a priori* known value. More details on how to set the parameters and control limits can be found in [86].

A key component of this study was a comparison of statistical properties between the proposed approach and the well-known CUSUM scheme. In addition to proposing a basic procedure that is sensitive to small shifts, they also presented three variations:

1. A scheme that is more sensitive to changes in the beginning of the time series (known as start-up problems in the online setting), when there are few observations. This property is known as FIR (fast initial response).

2. Some schemes suitable for small shifts are unable to handle large shifts. Lucas and Saccucci present a variation which is able to handle both small and large shifts.

3. A scheme which is robust against outliers.

Fang et al. [39] applied an EWMA-based change detection scheme in the context of land cover change detection. They apply first-order differencing to the input time series (NDVI, a measure of greenness), upon which it is assumed that the data is normally distributed. Fang et al. also provided a heuristic algorithm for estimating the change point.

The Kalman filter is a powerful and well-known technique frequently used to estimate the state of a process. A number of change detection techniques utilize Kalman filters as the underlying predictive model. Bifet and Gavaldà [11] introduce a method that combines Kalman filtering with an adaptive window scheme. The combined approach yields better performance than adaptive windows or Kalman filters alone.

3.5.2 Forecasting

Time series forecasting-based change detection approaches essentially build a forecasting model for the data; when the time series is very different from the forecast predicted by the model, a change may have occurred. Time series forecasting is highly dependent on assumptions about stationarity and modeling. A large number of forecasting techniques are based on models that combine moving average (MA) and autoregressive

(AR) processes, e.g. the autoregressive integrated moving average (ARIMA) model is a commonly used time series forecasting model.

Krishnamurthy et al. [77] studied change detection in the context of network intrusion detection. They build a model of normal network traffic and essentially look for changes inconsistent with the model. The problem under consideration involves streaming network data for which *sketches* are constructed; sketches are compact summaries of the stream. Krishnamurthy et al. [77] propose an ARIMA model which has been extended to sketches. An alarm is raised when the forecast error is above a certain threshold.

Gombay [47] proposed a change detection technique based on an AR model of order p. The technique is able to test for change in any of the p autoregressive parameters, the mean as well as the variance. The key advantage of Gombay's approach is that by estimating parameters based on the entire time series, it is much more computationally efficient than other techniques which perform estimations at all possible change points.

3.5.3 Martingale

Martingale is a statistical term which refers to a sequence in which the conditional expected value of the next observation in the sequence given all the previous observations is equal to the last observation.

Ho [60] proposed a martingale-based framework for multivariate change detection in the context of streaming *concept* change detection where the data is labeled. The core idea behind Ho's approach is testing for *exchangeability*, which specifies that switching the position of data points in a sequence does not alter the distribution; i.e. if the sequence is not exchangeable, it may contain a change point. The proposed approach begins by assigning a *strangeness* measure for each instance; this is done by using a classifier (for SVMs this could be set to the distance to the separating hyperplane). A p-value is then computed from the strangeness measure, and the martingale value is computed from the p-value. Two hypothesis tests are developed for detecting changes: the first essentially checks if the martingale for the latest observation in the sequence is above a threshold, while the second checks if the martingale *difference* between the latest observation and its prior observation are above a threshold. Note that this approach, as Ho observes, is effectively dependent on the classifier having high accuracy.

3.5.4 State-space

State-space models are those which consist of a (usually finite) set of states, such as Markov models.

Ge and Smyth [45] proposed a change detection approach which allows prior knowledge to be incorporated in a Bayesian fashion. The approach essentially combines segmentation with a hidden Markov model (HMM). The model is set up so that the segments are the states and the change points are transitions in the HMM. In their paper, the segments are modeled using regression models, but general models are possible. Prior knowledge about the duration of each state can be incorporated into the framework. The technique begins by running the EM (expectation-maximization) algorithm to obtain point estimates of the regression parameters. This is followed by running the Viterbi algorithm to obtain the change time, if any exists.

Carlin et al. [18] presented a Bayesian formulation for change detection. They present techniques for a number of problem settings including a Poisson process with a change point, changing linear models (similar to those seen in segmentation), and a Markov chain with a *change in transition matrix*. Note that this is different from the approach of Ge and Smyth [45], where the transition matrix itself does not change.

Raftery [101] also presented a Bayesian formulation, but considered two-dimensional change detection. He considered the problems of a Poisson process with a change point, and a state space model when the number of change points is not known, which is shown to perform better than a Kalman filter-based approach in some situations. He also presents the *Markov transition distribution model* for change points which is applied to image data.

Chib [31] proposed another Bayesian approach for multiple change point detection. His approach uses Markov models to specify a state variable which indicates the regime (distribution) to which the current point belongs, thereby detecting changes when the state variable switches regimes.

Firoiu and Cohen [40] proposed a segmentation technique in the context of a robot agent learning representations from sensor data. Their approach uses HMMs, where the states are artificial neural networks (ANN), and the transitions correspond to switching to different segments. It is assumed that the ANN can be used to approximate the function which generated the sensor data. The algorithm begins by creating a new

HMM state with an ANN that tries to approximate the time series; observations that are not well approximated by the ANN are set aside. The algorithm then loops between generating new states with the remaining points, finding the best segmentation using the current set of ANNs, and setting aside poorly approximated points from the best segmentation. The loop is terminated when no points are remaining. The final step involves reducing the segmentation model using an MDL criterion.

3.6 Anomaly Detection Approaches

The general problem of time series anomaly (or outlier) detection [26, 22] is to find regions of a time series that are significantly different from the rest of the time series in some sense. Anomaly detection is slightly different from change detection in that while change detection is looking for a change from one process to another, anomaly detection is looking for a *temporary* departure from normal. Nevertheless, solutions to both problems have many aspects in common. In particular, there are a number of algorithms which can be tuned to address both problems. The anomaly detection problem is also closely related to the problems of top-k discords, novelty detection, event detection and burst detection.

Parikh and Sundaresan [97] proposed a technique for *burst detection* in search query data. The query data is converted into a time series based on the frequency of arrival. Events (bursts) are detected using a two-state automaton, with states for a low arrival rate and high arrival rate. The idea is to run a given query time series through the automaton to discover the most likely model to have generated the sequence. In particular, transitions from the "low" state to the "high" state denote bursts. In addition to the above approach, the authors also provide an incremental algorithm which can be used in an online setting. Finally, the authors present schemes for ranking bursts based on qualitative and quantitative metrics.

Preston et al. [99] proposed an *event detection* approach to find significant deviations in astronomical time series (which are light curves from telescopes). The approach begins by transforming the input time series into *rank space* (by replacing continuous values with their ranks) in order to obtain a distribution with uniform noise. Then, the time series is divided into windows and the sum within each window is computed.

The key issue is assessing the statistical significance of a particular sum. Due to the computational intractability of the exact analytical approach, this issue is addressed by determining the probability distribution using a Monte Carlo method.

3.7 Subspace Approaches

Subspace approaches to time series change detection encode the input time series into a matrix structure, and then employ spectral methods such as principal components analysis and singular spectrum analysis. Changes are found by detecting changes in the matrix structure.

3.8 Other Considerations

3.8.1 Multiple Changes

For many change detection problem settings (e.g. parameter change and predictive approaches) it is more complicated to detect more than one change, while other problem settings (e.g. segmentation) lend themselves to multiple change detection naturally.

3.8.2 Multivariate Time Series

Most time series analysis, including change detection, is performed with univariate data. However, there are a number of techniques that have been proposed which can detect changes in multivariate time series data.

3.8.3 Multivariate Segmentation

When the data instances in a time series are vectors instead of real numbers, the problem is a multivariate change detection problem. This problem can be solved as a data clustering problem, with the constraint that the individual instances in a cluster must be contiguous in the time series. Therefore, clustering algorithms can be modified for this constraint and then applied to the multivariate segmentation problem. In this case, the underlying (nonparametric) model would be the clusters. Bingham et al. [12] address the problem of multivariate segmentation.

3.8.4 Multiple Time Series Change Detection

Multiple time series change detection refers to the problem of detecting changes in the context of an underlying data set. This is a different definition than the traditional definition of change where the change is in relation to the time series (and corresponding process) itself. Detecting anomalies in the context of a background data set has been studied in a survey by Cheboli et al. [26]. In the context of change detection, information from the background data set can be incorporated into certain approaches, viz. predictive and anomaly detection approaches.

3.8.5 Concept Drift

When a labeled multidimensional data stream changes behavior either in the input distribution or in the labels, this phenomenon is known as concept drift. Note that we use the generic term label to refer to a target, which may be a class label in the context of classification or a continuous value in the context of regression. Concept drift is indirectly related to change detection in that the problem is one of detecting changes in *labeled* data where changes in the input distribution would also be considered change points. This problem has been studied by Dries and Rückert [37], Severo and Gama [112] and Tsymbal [121].

3.8.6 Gradual and Abrupt Change

A gradual change is one which occurs over several time steps, while an abrupt change occurs in very few time steps. Many techniques can be tuned (by setting a parameter) to discover gradual or abrupt changes, while other techniques can only discover abrupt changes.

3.9 Search Procedures

In this section, we will discuss search procedures that are used by various change detection techniques from an algorithmic perspective. For example, techniques may perform global or local searches, greedy or exhaustive searches, etc. The type of search procedure used has an impact on the quality of the results obtained as well as on the computational cost incurred. Note that the notion of a search procedure is applicable only for some change detection algorithms. For example, most segmentation algorithms have a search component. Keogh et al. [74] performed a similar analysis in the context of segmentation algorithms.

Based on the approach taken in relation to processing the time series, algorithms can be roughly divided into the following types:

- Top-down
- Bottom-up
- Direct
- Dynamic Programming
- Sliding window
- Model-based

3.9.1 Top-down

Top-down algorithms have the following general structure:

1. Divide the time series into k (usually 2) sections at a change point.
2. Recursively process each section. (Requires a *stopping condition*.)

Top-down algorithms can only be applied in the offline setting, and are essentially greedy in nature. Unlike greedy clustering algorithms, where earlier decisions cannot be reversed, top-down change detection algorithms *can* subsequently change their choices but a poor choice early-on may still lead to a sub-optimal final output. For example, this can be done by maintaining a list of change points and dropping "poor" change points from the list if better ones are found ([84]).

3.9.2 Bottom-up

Bottom-up algorithms have the following general structure:

1. Begin by dividing the time series into the smallest possible homogeneous sections.

2. Combine small sections of the time series that are similar. This requires a method to *compare* sections and a method to *combine* sections.

3. Stop combining when the sections are no longer similar. This requires a *stopping condition*.

4. The remaining time steps consist of change points.

Bottom-up algorithms can be applied only in the offline setting, and are also greedy in nature. However, unlike the case of top-down algorithms, bottom-up algorithms cannot reverse earlier decisions since the change points are not discovered until the final stage.

3.9.3 Direct

These algorithms have the following general structure:

1. Small sections of the time series are compared with their immediate neighbors, or a statistic such as a running mean.

2. If the difference is above a certain threshold, the sections contain a change point.

Direct algorithms can be applied in both the online and offline settings, and are exhaustive in the sense that every possible solution is searched. However, due to the heavy dependence on the threshold, direct algorithms may not perform effectively in scenarios where it is difficult to select a good threshold value.

3.9.4 Dynamic Programming

If a change detection problem can be expressed in the DP framework, then an optimal solution can usually be found in quadratic time relative to the length of the time series. Specifically, the problem needs to be such that it can be broken into subproblems, and an optimal overall solution can be constructed from optimal solutions to the subproblems. There are a number of change detection algorithms which use DP as the search procedure. The primary benefit comes from the fact that the DP approach is much more efficient than a brute-force search, which can take exponential time. Algorithms that use DP search are offline.

3.9.5 Sliding Window

Sliding window algorithms have the following general structure:

1. Maintain a window of observations (requires some notion of *history*).
2. Look ahead until an observation that is significantly different from the window is encountered.

Sliding window algorithms are usually designed for the online setting, and the search is not exhaustive by nature since only a fixed history is maintained.

3.9.6 Model-based

Model-based algorithms have the following general structure:

1. Construct a model for a section (usually the beginning) of the time series.
2. When time steps outside this section are sufficiently different (i.e. it is not plausible these values could be generated by the model), these time steps are considered change points.

Model-based algorithms are like sliding window algorithms and are designed for the online setting. These algorithms usually have a *forgetting factor* which determines how older observations are remembered. In fact, sliding window algorithms are a special case of model-based algorithms in which the forgetting factor is a step function. While an exhaustive search is not performed, depending on the quality of the model and update mechanism, it is possible for model-based algorithms to find the optimal solution.

Chapter 4

Scalable Algorithms for Change Detection

The general time series change detection problem has a number of requirements, the most common of which are high accuracy and early detection. There are a large number of (sometimes opposing) additional requirements that can be taken into account, but most change detection approaches attempt to optimize over the requirements imposed by a specific domain; e.g. the CUSUM algorithm was first designed in the context of industrial process control. The earth science domain imposes a number of challenges on change detection algorithms, including non-stationarity, nonlinear processes, multi-scale nature, and low-frequency variability. Additional challenges in global land cover change studies include the massive data size, high degree of geographic/inter-region variation, noisy/missing data, disparate land cover types, and the large variety of changes that can occur.

For ecosystem data, change detection is the process of identifying changes in the cover type and/or human use of the Earth. There is a large body of research in change detection using remotely sensed image data (see Chapter 2). Most studies have focused on relatively small areas or have described changes in specific categories of interest [32, 68, 120, 123] because they are inherently unsuitable for global analysis. Change detection has most often been performed by comparing two or more images of different dates, or snapshots, to determine the land cover differences between them. Recent work has recommended the use of high temporal resolution (bi-weekly) time series data for land cover change detection in applications where high spatial resolution images are impractical, e.g., large study areas [75]. Time series based change detection has significant advantages over the comparison of snapshot images of selected dates because it takes into account information about the temporal dynamics of landscape changes. Detection of changes is based on the pattern of spectral response of the landscape over time rather than the differences between two or more images collected on different dates. Therefore additional parameters such as the rate of the change (e.g. a sudden forest fire vs. gradual logging), the extent, speed, and pattern of regrowth can be derived. There has been some work in time series-based land cover change detection, but this work has had limited success. For example, the change detection algorithm [106] used to generate the Burned Area Product (a well-known MODIS data set) performs very poorly in parts of North America such as California as illustrated in Figure 4.1. In addition, such algorithms and products are geared towards specific kinds of changes (such as fires), and are not capable of detecting the broad set of changes that can potentially be addressed (such as those due to deforestation, floods, droughts and insect infestations).

The general problem of change detection in time series data has been extensively studied in the fields of statistics, signal processing and control theory. Most of the previous work in time series change detection (discussed in Chapter 3) is not suitable for our land cover change detection problem for several reasons including: (1) the techniques are computationally expensive, making them infeasible for large scale high-resolution earth science data sets; For example, a global image library based on ten years of bi-weekly 1 km MODIS images would require the storage and processing of over 230 images having 160 million pixels each. (2) the utility of the techniques are limited by the significant data issues (non-stationarity, noisy/missing data, etc.); (3) the techniques are unable to take advantage of the inherent structure present in earth science data.

In this chapter, we will present scalable algorithms for the time series change detection problem that address many of these challenges. We discuss robust algorithms separately in Chapter 5. Specifically, we present novel techniques that we have developed, including a segmentation-based technique and a predictive technique (Sections 4.3, 4.4 and 4.5). We also present an anomaly detection-based approach published in

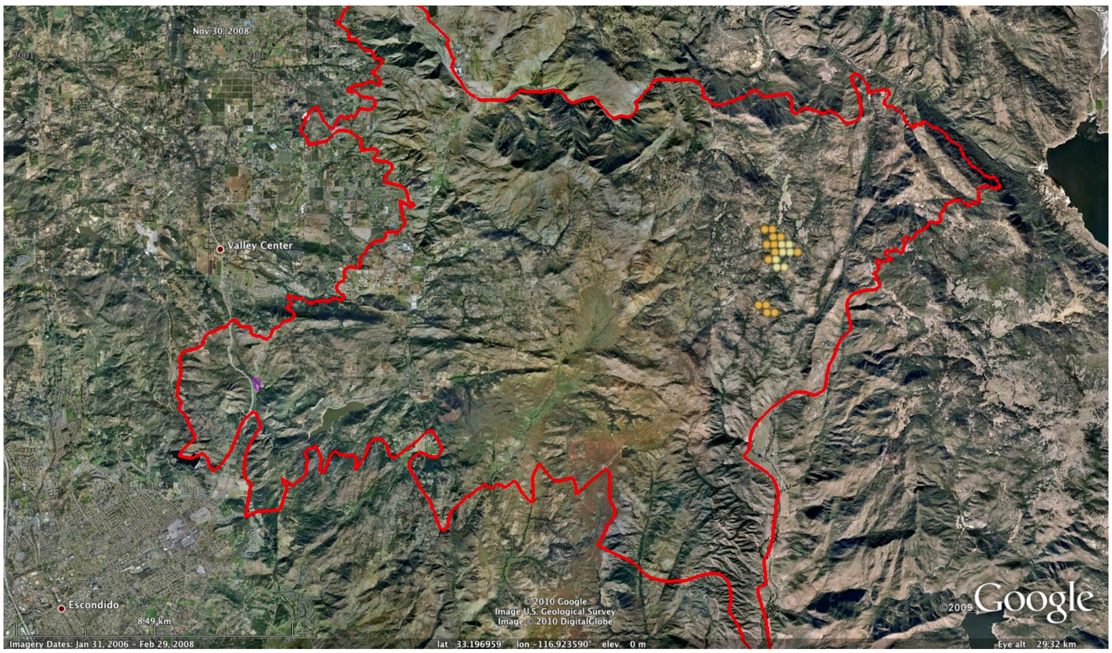


Figure 4.1: This figure illustrates the poor coverage of the Burned Area product in California. The figure is a screenshot from Google Earth that shows the boundary of a fire near San Diego in 2003 (red line), and the pixels detected by the Burned Area product (circles).

the earth science domain and a well-known statistical parameter change algorithm that we have adapted to the change scoring framework (Sections 4.1 and 4.2). All of the techniques are also discussed in the context of the taxonomy in Chapter 3. We will follow the notation defined in Table 2.2 to describe the techniques in this chapter. The techniques described in this chapter will be experimentally evaluated with real-world ground truth data in Chapter 7.

4.1 Lunetta et al. Scheme

Recently, a change detection study that uses MODIS data was performed by Lunetta et al. and published in the earth science literature [87]. This comprehensive study consists of a novel data cleaning method (a smoothing approach; see Section 5.3), a change detection method, as well as an extensive discussion of the change patterns in the region of interest. The authors performed a case study applying their change detection technique to a region known as the Albemarle-Pamlico Estuary System on the Virginia–North Carolina border. For our study, this change detection method is most relevant since it is one of the few change detection methods that has been applied to high-resolution MODIS data. Although Lunetta et al. do not work with EVI data in their paper, they do work with a closely related variable called the Normalized Difference Vegetation Index (NDVI). NDVI, like EVI, is also a measure of the vegetation level at a given land location, the main difference being that EVI is designed to enhance the vegetation signal with improved sensitivity in high biomass (high vegetation density) regions [63]. The time series change detection algorithm is a combination of an anomaly detection approach (Section 3.6) using a background multiple time series data set (Section 3.8.4). We define the following terminology prior to discussing Lunetta et al.'s approach.

Definition 4.1.1 (z-score)**.** The z-score is a measure of the deviation between a given data point and a known distribution; the z-score is also known as the standard score or the normal score. Traditionally, the z-score is computed with reference to a Gaussian distribution and is defined as

$$z = \frac{x - \mu}{\sigma}$$

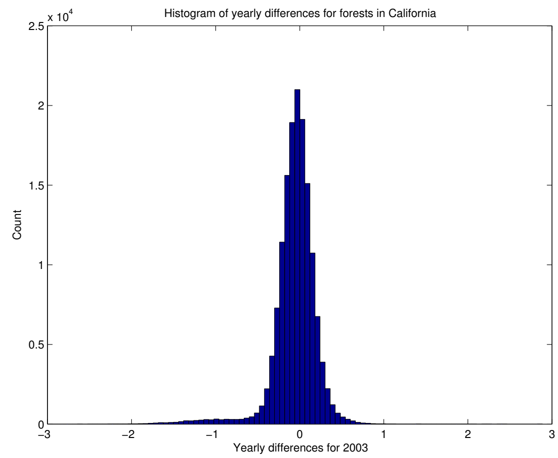


Figure 4.2: This histogram shows the distribution of integrated annual differences for a data set that consists of forests in California (this data set, called DS1, will be introduced in Chapter 7).

where x is the data point, and μ and σ are the mean and standard deviation of the Gaussian distribution, respectively.

The proposed technique begins by computing annual sums. The difference between consecutive years is then computed; this is equivalent to applying first-order differencing [25] to the time series of annual sums. The resulting differences are assumed to follow an approximately normal distribution with $\mu = 0$, which represents that no change has occurred. This is justifiable in that most locations do not exhibit change and therefore on average the change in annual sum between consecutive years is expected to be 0. Figure 4.2 shows that this is indeed the case in practice for a sample real data set.

To detect whether a change has occurred between year t_1 and year t_2, the z-score of the difference of annual sums is computed. The computation of the z-score is one of the critical aspects of the algorithm. When computing the z-score, Lunetta et al. define the standard deviation across *all* the spatial neighbors of the pixel for that time window in the data set. This innovative mechanism exploits information across space from the background data set in the change detection process; therefore, the scheme is a *spatio-temporal* change detection algorithm. If the z-score is above a threshold τ, a change is considered to have occurred between t_1 and t_2. The specific steps in Lunetta et al.'s algorithm for land cover change detection are as follows:

1. For each location n_i, the annual EVI sum is computed for each year of data. Let $\{a_{i1}, \ldots, a_{iY}\}$ correspond to this list of annual sums, where $a_{i1} = \sum_{j=1}^{12} n_{ij}$, etc.

2. The difference between the annual sum for a given year and the previous year is computed, i.e., $\{a_{i2} - a_{i1}, a_{i3} - a_{i2}, \ldots, a_{iY} - a_{iY-1}\}$. Let $d_{ij} = a_{ij+1} - a_{ij}$.

3. The z-score is computed for each of the $Y-1$ values in $\{d_{i1}, d_{i2}, \ldots, d_{iY-1}\}$. This is done for each d_{ij} by subtracting the mean (set to 0) and dividing by the standard deviation ($= st.\ dev.\{d_{1j}, d_{2j}, \ldots, d_{Nj}\}$). Note that the standard deviation is computed across *space*. Let $\{z_{i1}, z_{i2}, \ldots, z_{iY-1}\}$ correspond to this list of z-scores.

4. If z_{ij} is above a certain threshold, then location n_i is considered to have a change between the consecutive years corresponding to j and $j + 1$.

In this dissertation we have adapted Lunetta et al.'s algorithm in order to apply it to the current change detection setting. Essentially, after applying Lunetta et al.'s algorithm to the data set, we are left with a list of z-scores for each location that corresponds to each pair of consecutive years. Since a single change score is assigned to each location in our problem setting, we adapt Lunetta et al.'s algorithm by taking the minimum for each location's list of z-scores to be the change score for the location, i.e.,

$$change\ score(n_i) = \min\{z_{i1}, z_{i2}, \ldots, z_{iY-1}\}.$$

In subsequent discussions, we will refer to the modified Lunetta scheme described above as the LUNETTA scheme. The LUNETTA algorithm has a number of key strengths:

- The algorithm incorporates the spatial neighborhood of the pixel into the change detection process using information from the other time series in the data set. This is advantageous in many respects; one significant advantage is that it offers a mechanism that can account for time periods that have high natural variability or variability induced by noise. Thus, annual differences that are artificially high for a particular time window because of reasons other than land cover change will be discounted by the algorithm, leading to a lower change score.

- The change score is tied to the difference in the integrated annual vegetation and therefore has a physical interpretation. This is helpful to domain scientists when studying the degree of change in terms of the amount of vegetation lost or gained.

- The algorithm has the ability to detect *multiple* changes across time. For land cover data sets that cover longer time periods, it is common for a pixel to undergo several land cover changes although we do not consider this setting in this dissertation.
- The algorithm can detect change in both positive and negative directions. In terms of the domain, this means that both *gain* and *loss* in vegetation can be detected.
- The computational complexity is low and the algorithm can be easily applied to massive data sets.

The LUNETTA algorithm has several limitations, some of which can potentially be addressed by future extensions to the algorithm:

- The algorithm incorporates the spatial neighborhood of a pixel into the change detection process by normalizing annual differences using the standard deviation. However, the spatial neighborhood must be coherent for this process to function appropriately. This is not the case, for example, when multiple land cover types are in the data set. An in-depth discussion of this aspect of the algorithm follows in Section 4.1.1.
- Since the algorithm collapses the time series into yearly sums, the estimated time of change is accurate only up to the year of change. Obtaining a finer estimate of the time of change using, for example, sliding windows, would significantly increase the computation complexity of the algorithm.
- Although the algorithm does account for noise to some extent, it does not offer a principled mechanism for dealing with missing data. In particular, it is not clear that an input time series with many missing values can be reliably processed.

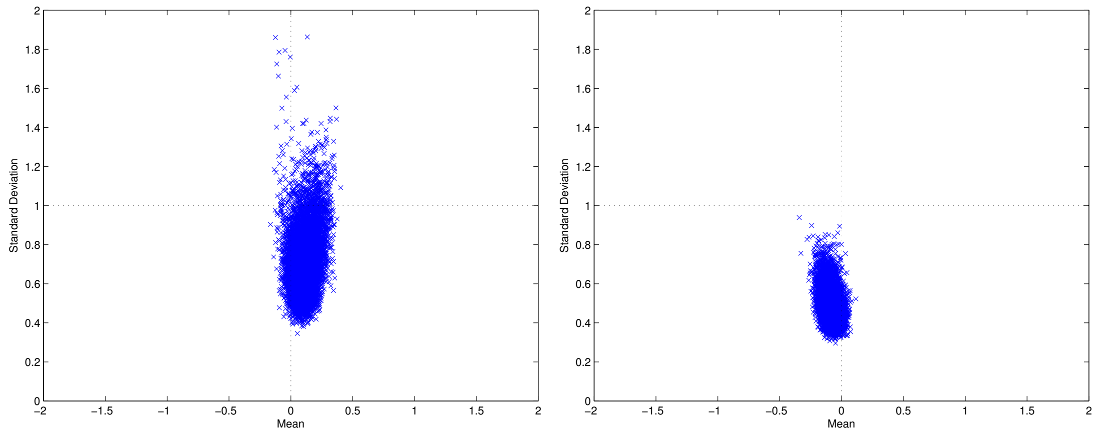
4.1.1 Normalization Step

For a given pixel, the LUNETTA algorithm normalizes each annual difference (for discussion say $a_{i2} - a_{i1}$) by the standard deviation of all neighbors across space ($st.\ dev.\{a_{\cdot 2} - a_{\cdot 1}\}$). The idea behind normalization is to account for time periods for which there is high variability across space. For example, variability in temperature and precipitation during a particular year may lead to higher change scores even though no real land cover change has occurred; by dividing by the standard deviation across space the higher change scores are normalized by the `LUNETTA` algorithm.

However, there are two situations where the normalization step can lead to undesirable performance: (1) There are multiple land cover types in the data set. Each of these land cover types will have very different time series structures which implies that the standard deviation will not reflect the true variability within a given annual window; (2) The change detection problem is focused on detecting forest fires. In this situation (even though all pixels are of one type), if there is a year which has an unusually large number of fires, the normalization step has the unwanted consequence of reducing the change score.

The first issue is illustrated in Figure 4.3, which shows that the standard deviation can have different distributions based on the spatial context under consideration. Each of these distributions was generated by randomly sampling 100 pixels (all pixels vs. only forest pixels in California), then computing the standard deviation of the sample; 10,000 samples were taken in each case. The distribution of the mean is close to 0 in both cases. However, it can be observed that the distribution of the standard deviation on the right is *tighter* and centered around a different value (since it corresponds to one land cover type), while the distribution on the left corresponds to all land cover types. This limitation can be addressed by extending the algorithm to deal with multiple land cover types by introducing the notion of time series models (see Figure 6.1 for an example of cluster-based models). Instead of including the entire spatial neighborhood of a given time series, such an extension would only include *relevant* spatial neighbors by ensuring that they have a similar model.

The second issue relates to the problem of detecting forest fires. The time series pattern for a forest fire typically involves a sharp and sustained drop at the time of the fire (see Figure 4.7 for an example). Therefore, in a time series data set of forest pixels, there are likely to be several time series with such sharp drops, and the number of such time series will be higher in years when a large number of pixels are burned. Table 4.1 and Figure 4.4 show the number of pixels burned in each year from 2001 to 2008 on the `DS1` data set (this data set is introduced in Chapter 7); also shown is the standard



(a) All land cover types.

(b) Forests only.

Figure 4.3: Distribution of the standard deviation and mean of annual differences ($d_{i\cdot}$ in Lunetta et al.'s scheme) of all land cover types (left) and forests (right) in California for the year 2004.

deviation of the annual differences corresponding to each year. The data indicates that the standard deviation of annual differences is higher for time periods when a greater number of pixels are burned. For these years (especially 2008), the change scores will be diminished compared to a year such as 2006. This means that if pixel n_i has a fire in 2006 and pixel n_j has a fire in 2008 and they have *exactly* the same time series, pixel n_i will receive a higher change score than pixel n_j. Thus, we observe that the normalization step can lead to a suboptimal change score when the problem at hand is one of detecting forest fires.

4.1.2 Variation of the Scheme

We define a variation of the LUNETTA algorithm described above, which we will call LUNETTA_NO_NORM. This variation is specifically designed to overcome the limitations discussed in Section 4.1.1. This is accomplished by removing the normalization step from the change detection process. Thus, in the notation used to described the basic Lunetta et al. approach, the LUNETTA_NO_NORM change score is defined as

$$change\ score(n_i) = \min\{d_{i1}, d_{i2}, \ldots, d_{iY-1}\}.$$

Year	# of fire pixels	Standard Deviation
2001	1142	0.20
2002	2407	0.32
2003	4946	0.27
2004	661	0.27
2005	192	0.28
2006	278	0.21
2007	1935	0.29
2008	6778	0.37

Table 4.1: Standard deviation of integrated annual differences on `DS1`.

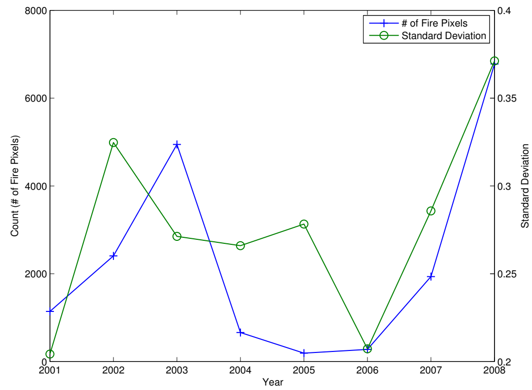


Figure 4.4: A visual representation of the data in Table 4.1.

4.2 CUSUM

CUSUM is a well-known change detection algorithm that was originally developed in the domain of process control. It is one of the earliest change detection algorithms developed, proposed by Page in 1954 [95], and has spawned an entire class of algorithms to study a variety of problems, such as change in variance [66], etc. This family of algorithms, categorized as statistical parameter change approaches, is discussed in Section 3.3.

The main intuition behind CUSUM was inspired by problems in manufacturing, where an industrial process may deteriorate to a point where the quality is no longer acceptable, but such shifts may happen gradually or abruptly. One of the defining features of CUSUM is its ability to detect *small* and *gradual* changes in the process. The basic CUSUM scheme has an expected value μ for the process. It then compares the *deviation* of every observation to the expected value, and maintains a running *statistic* (the cumulative sum) CS of deviations from the expected value. If there is no change in the process, CS is expected to be approximately 0. The scheme also defines *control limits*, UCL and LCL,[1] which denote the maximum allowable deviation before a process is considered to be out of control, i.e. if $CS \notin [LCL, UCL]$ there is a change in the process.

Before we present the CUSUM approach, we define the following terminology:

[1] LCL = lower control limit. UCL = upper control limit.

Definition 4.2.1 (In-control Mean)**.** Let $\{t_1, t_2, \ldots\}$ denote the sequence of observations of a process. The in-control mean μ is the expected value of the process in the sense that, while each $t_i = \mu + \epsilon$, there is a subsequence such that

$$\text{mean}\{t_i, t_{i+1}, \ldots, t_j\} \sim \mu.$$

Note that the in-control mean can be set to a new value at any time step r during the course of an online process. In this case, the old observations are discarded and the restarted process is treated as a new time series where t_1 corresponds to the observation at time r.

Definition 4.2.2 (CUSUM Statistic)**.** Given a time series $\{t_1, t_2, \ldots\}$ and an in-control mean μ, the CUSUM statistic CS_k is defined as

$$CS_k = \sum_{i=0}^{k} (t_i - \mu).$$

Given an input time series $\{t_1, t_2, \ldots, t_T\}$, the basic CUSUM algorithm proceeds as follows:

1. The basic algorithm sets μ to an *a priori* known value, or to the first value t_1 of the time series.

2. Compute the CUSUM statistic CS_k for $k = 1 \ldots T$.

3. Set the change score to the minimum CUSUM statistic (to detect negative changes),

   $$\textit{change score} = \min\{CS_1, CS_2, \ldots, CS_T\}.$$

   To detect positive changes, *change score* $= \max\{CS_1, CS_2, \ldots, CS_T\}$.

Figure 4.5 illustrates how the basic CUSUM algorithm works with a simulated time series with $T = 200$. The input time series $\{t_1, t_2, \ldots, t_{200}\}$ consists of two randomly generated Gaussians distributed as follows:

$$t_i \sim \begin{cases} \mathcal{N}(0, 0.25) & 1 \leq i \leq 100 \\ \mathcal{N}(0.25, 0.25) & 101 \leq i \leq 200. \end{cases}$$

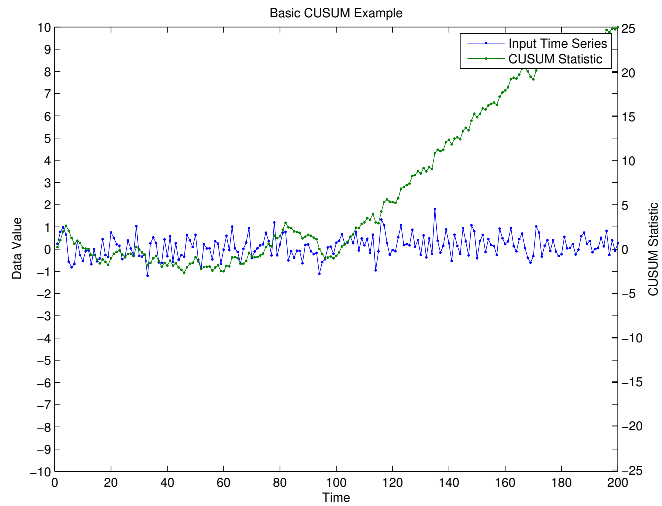


Figure 4.5: Basic CUSUM algorithm using $\mu = 0$. The input time series *has* a change point at time step 100 (the mean of the generating process changes from 0 to 0.25). Note that the y axis on the right corresponds to the CUSUM statistic and is not of the same scale as the one on the left.

In other words, the mean of the generating process changes from 0 to 0.25 after time step 100. The algorithm is run with in-control mean $\mu = 0$ (the variance is constant at 0.25). Figure 4.5 shows the CUSUM statistic generated in addition to the input time series. We observe that the statistic is mostly around zero until the change point, time step 100, after which the CUSUM statistic starts to rapidly grow. This example illustrates one of CUSUM's greatest strengths, which is the ability to detect very small shifts. In this example, the change in mean is visually almost imperceptible, and the new mean (0.25) is well within the normal range of values for the process before time step $i = 100$. Yet the algorithm shows a remarkable ability to correctly detect the change with minimal delay.

In the example in Figure 4.5, we set the in-control mean to 0 since the distribution of the input data was known *a priori*. However, let us say we do not know the input distribution *a priori*, and μ is set to the first value of the time series, i.e., $\mu = t_1$. Let us also consider an input time series with *no* change point, i.e.,

$$t_i \sim \mathcal{N}(0, 0.25) \quad 1 \leq i \leq 200.$$

Figure 4.6 shows the results for this case. It can be observed that the CUSUM statistic

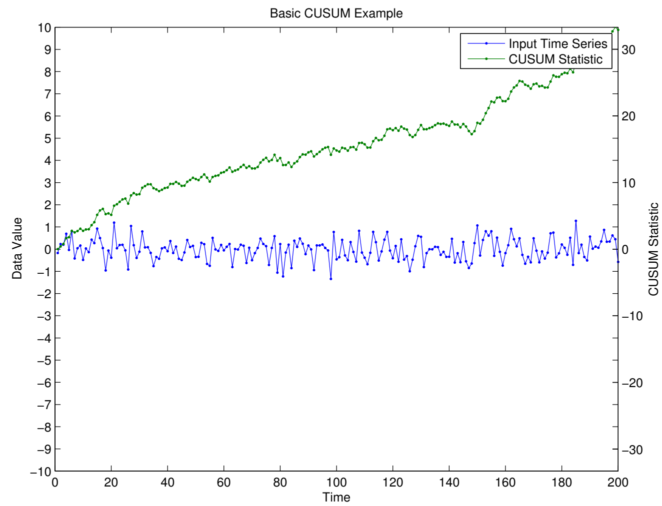


Figure 4.6: Basic CUSUM algorithm using $\mu = t_1$. The input time series has *no* change point. Note that the y axis on the right right corresponds to the CUSUM statistic and is not of the same scale as the one on the left.

CS_k increases at an almost constant rate from the beginning of the time series. This behavior continues through the end of the time series even though there is no change of mean in the generating process. Thus, we see that the success of the algorithm in detecting out of control processes depends critically on the value of μ since the CUSUM statistic will grow (positively or negatively) in a monotonic fashion if μ does not reflect the expected behavior of the time series. However, in general, one does not know the in-control mean *a priori* in the land cover change problem setting.

To address the issue of selecting an appropriate in-control mean μ, we have developed an adaptation of the basic CUSUM algorithm for the current problem setting, which we will call CUSUM_MEAN. CUSUM_MEAN proceeds as follows. Let $\{t_1, t_2, \ldots, t_T\}$ denote the input time series and let S correspond to the number of observations per year.

1. Set the in-control mean μ to the mean of the first annual cycle of the time series,

$$\mu = \text{mean}(t_1, t_2, \ldots, t_S).$$

2. Compute the CUSUM statistic CS_k for $k = 1 \ldots T$.

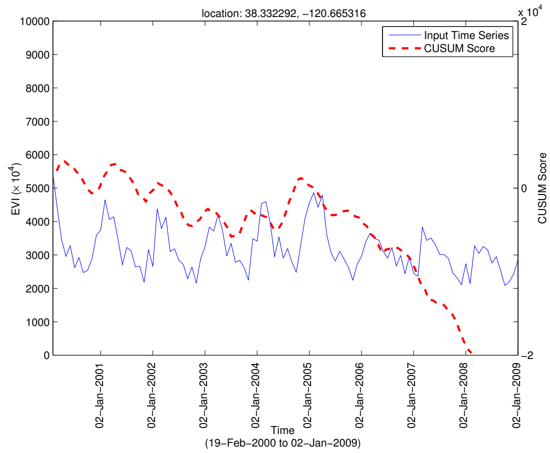


Figure 4.7: Example of how CUSUM_MEAN works on a sample time series.

3. Set the change score to the minimum CUSUM statistic (to detect negative changes),

$$\textit{change score} = \min\{CS_1, CS_2, \ldots, CS_T\}.$$

To detect positive changes, *change score* $= \max\{CS_1, CS_2, \ldots, CS_T\}$.

Figure 4.7 illustrates how CUSUM_MEAN works on a time series which has an actual land cover change. It can be observed that the CUSUM statistic stays within a certain range centered around 0, and begins to drop monotonically soon after the change has occurred. Therefore, because of the more appropriate choice of μ, the CUSUM_MEAN scheme is better suited for the current land cover change problem than the basic CUSUM algorithm. We will explore the performance of CUSUM_MEAN further in the quantitative evaluation performed in Chapter 7.

We discuss some of the strengths of the CUSUM approach below:

- One of the most significant strengths of CUSUM-based approaches is the ability to detect small and gradual shifts (as seen in Figure 4.5). This is a type of change that many time series change detection algorithms are incapable of detecting.
- The algorithm can detect the time of change close to the actual change with accuracy up to the temporal resolution of the input data set, although postprocessing may be required to obtain the most accurate estimate.

- CUSUM-based approaches are a well-studied family of algorithms, with a rich range of variations and strong process control and statistical underpinnings. Therefore, the literature offers a strong base upon which to build extensions in several directions.
- CUSUM's simple scoring model also means that it is extremely fast and scalable.

Some of the limitations of the CUSUM approach are discussed below:

- The algorithm is sensitive to noise and missing data (this behavior will be shown in Chapter 7).
- For data with periodic cycles, where the CUSUM statistic itself may oscillate between a range, the algorithm is sensitive to slight pattern changes in the periodic cycle that do not represent real changes (this behavior will also be shown in Chapter 7).
- The algorithm is constrained in its ability to detect multiple changes and changes in both directions simultaneously for a given problem. This is because if the algorithm tries to detect changes in both directions, then large positive and negative change scores will both need to be examined, complicating the scoring process. The multiple change scenario is also complicated to address given that the CUSUM statistic typically grows rapidly after a shift in the process has occurred, and any extension seeking to detect multiple changes would need to reset both the CS and μ to appropriate values after each change.
- CUSUM_MEAN's change scores are easy to interpret visually, but it is difficult to attach a physical interpretation from a domain perspective.

4.3 Departure from Model

Large time series data sets frequently consist of instances belonging to several *classes* of data. These classes may be supplied *a priori* (supervised) or may be discovered by forming homogeneous groups via a technique such as clustering (unsupervised). Models describing each class can be constructed. A change detection approach can then take advantage of such models by comparing a given time series to its nearest model to detect deviations from the model. This scheme is a combination of an anomaly detection approach (Section 3.6) using a background multiple time series data set (Section 3.8.4). Before we describe the approach we define the following terms:

Definition 4.3.1 (Homogeneity)**.** A group of time series is said to be homogeneous if their similarity to each other or a central point is higher than a given threshold. The similarity function used to measure homogeneity can include a combination of spatial proximity, time series similarity, and similarity of other features such as land cover types.

Definition 4.3.2 (Time Series Model)**.** A time series model describes the expected behavior of a group of homogeneous time series. For example, the model can be a compact summary such as the *centroid* of a time series cluster. A model can also include additional information such as confidence intervals for each time step.

Definition 4.3.3 (Conformity)**.** Given a time series subsequence $\{t_i, t_{i+1}, \ldots, t_j\}$ and a time series model $\mathcal{M}$, a conformity metric measures the extent to which the subsequence conforms to the expected behavior described by the model.

Definition 4.3.4 (Deviance)**.** Given a time series subsequence $\{t_i, t_{i+1}, \ldots, t_j\}$ and a time series model $\mathcal{M}$, a deviance metric measures the extent to which the subsequence disagrees with the model's expected behavior.

Based on the framework described above, we define a meta-algorithm for time series change detection. The algorithm is defined broadly such that depending on the problem at hand, the key components can be optimized in multiple ways. The algorithm proceeds as follows:

1. Build time series *models* (using spatial proximity or clustering) for many classes of data.

2. Assign each time series to one of the models, based on some notion of *conformity*. Compute a *metric* C which measures the extent to which the time series conforms to the model for a portion of the data.

3. Compute another metric D which measures the extent of deviation.

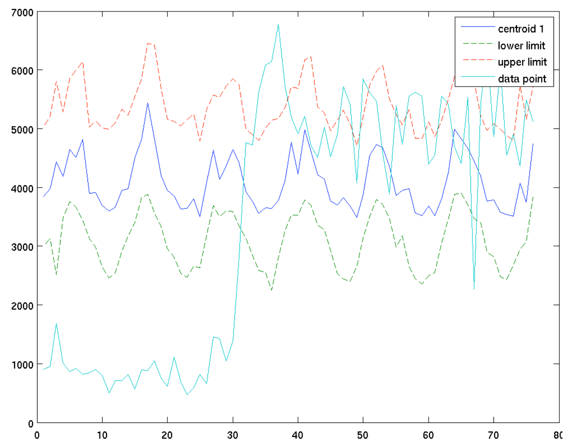


Figure 4.8: This figure shows an example of a change point that was discovered by the confidence intervals around cluster centroids methodology.

4. Combine the C and D metrics to assign a *change score* to the time series.

We devised a clustering-based methodology that is a specific instantiation of the above approach. This scheme is based on the assumption that most of the data can be grouped into well-defined clusters and that the majority of data does not exhibit a change. Preliminary results of this algorithm were presented in [14].

The algorithm works as follows. The data are clustered into k major classes using a clustering algorithm. For each cluster, an upper and lower interval is constructed based on the 5th and 95th percentile, i.e. for every month, the percentiles are computed from all the data points in the cluster. Data points are characterized as outliers based on their relationship to these intervals, using a numerical score which signifies the number of months that a data point lies outside the interval.

Figure 4.8 shows an example of a change point that was discovered by this scheme. The figure shows how a point that is partially within the confidence interval of a cluster and partially outside is likely to be a change point. One of the strengths of this scheme is that it is robust to noisy observations that are interspersed throughout the time series. The primary drawback of this scheme is that there is a heavy dependence on the quality of the clustering (i.e. the fidelity of the models in capturing many classes of data). If there is a group of points which have similar time series but are not in a cluster by themselves, then these points may be discovered as change points. For example, Figure

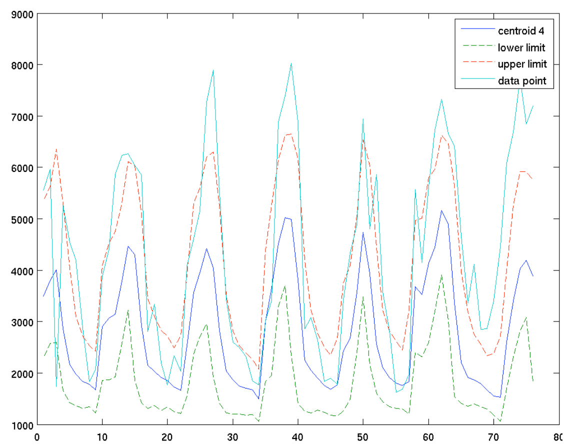


Figure 4.9: This figure shows an example of a point that was discovered by the confidence intervals around cluster centroids methodology. It is evident from the time series that this point should not be considered a change point.

4.9 shows a time series that does not contain a change, but is likely to belong to a class of points that was not found as a cluster.

This algorithm has high computational complexity, primarily because the initial cluster/model generation step is typically an expensive step. In addition, the clusters must be finely tuned with a postprocessing step to ensure noise clusters are removed. Therefore, we did not include this algorithm in the experimental evaluation in Chapter 7.

4.4 Recursive Merging Algorithm

The recursive merging algorithm follows a segmentation approach to the time series change detection problem. Segmentation approaches, which try to find homogeneous segments of the time series, are discussed in Section 3.4. A preliminary qualitative evaluation of this algorithm was presented in [13]. The algorithm operates under the assumption (standard for segmentation) that a given time series can be partitioned into homogeneous segments and that boundaries between segments represent change points. We first outline the base algorithm (called RM0) from [13], and then present various design choices that can be altered to create variations of the algorithm.

We designed the RM0 technique based on some key characteristics of the underlying data set: (1) most land locations do not exhibit a change; given the large coverage of land cover data sets, it is fairly obvious that only a small fraction of points will actually exhibit a change; and (2) the major mode of behavior in the vegetation signal is seasonality, i.e., the natural seasonal growing cycle is a dominant characteristic of a time series and this intrinsic seasonality should not itself be called a change. The main idea behind the recursive merging algorithm is to exploit seasonality in order to distinguish between points that have had a land cover change and those that have not. In particular, if a given location has not had a land cover change, then we expect the seasonal cycles to look very similar going from one year to the next; if this is not the case, then based on the extent to which the seasons are different one can assign a change score to a land location. The time series for each location is processed as follows:

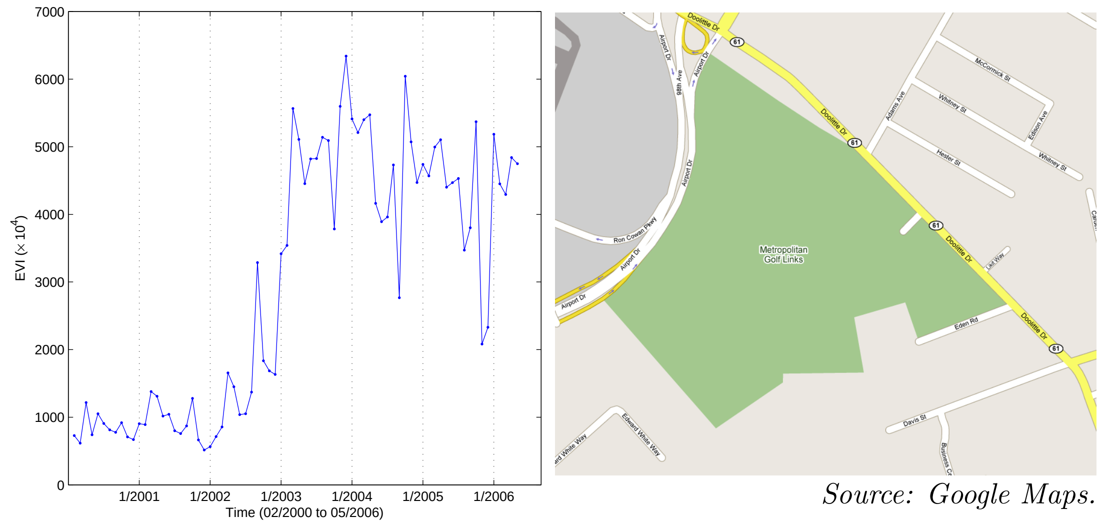
1. The two most similar consecutive annual cycles are merged, and the distance is stored. Let $\{b_{i1}, \ldots, b_{iY}\}$ correspond to the list of annual cycles, where $b_{i1} = [n_{i1}\ n_{i2}\ \ldots\ n_{i12}]$, etc. Suppose b_{i1} and b_{i2} are the two most similar annual cycles; then, at the end of this step what is left is a list with 1 less element, $\{\frac{b_{i1}+b_{i1}}{2}, b_{i3} \ldots, b_{iY}\}$, along with the distance $s_1 = dist(b_{i1}, b_{i2})$.

2. Step 1 is applied recursively until one annual cycle is left remaining. This results in a list of distances of length $Y-1$, $\{s_1, \ldots, s_{Y-1}\}$. Note that the order in which items are inserted into this list is not related to the order of the annual cycles, it is merely the order in which seasonal cycles were merged.

3. The change score for this location is based on whether any of the observed distances are extreme. This can be quantified by computing the quantity $\frac{\max\{s_1,\ldots,s_{Y-1}\}}{\min\{s_1,\ldots,s_{Y-1}\}}$.

Note that it is possible that $\min\{s_1, \ldots, s_{Y-1}\}$ might be 0, in which case a small value ϵ can be added to this quantity.

Figure 4.10 shows an example of a land cover change pattern that was detected by the RM0 algorithm and is typically of interest to Earth Scientists. The time series shows an abrupt jump in EVI in 2003. The location of the pixel corresponds to a new golf course, which was in fact opened in 2003. For this time series, the algorithm merges segments before 2003 (they are the most similar) followed by those after 2003. When



Source: Google Maps.

Figure 4.10: This figure shows an example of a change point in the San Francisco Bay Area. When the geocoordinates of this pixel were entered into a mapping service, we found that it corresponds to a golf course in Oakland, CA (right pane). This golf course was built in 2003, which corresponds to the time step at which the time series exhibits a change.

the last two segment models around 2003 are compared, their relative distance is very high giving this time series a high ranking.

The `RM0` technique bears some resemblance to the bottom-up segmentation algorithm discussed by Keogh et al. [74] in the context of linear approximations of time series. The goal of linear approximation algorithms is to provide the best segmentation using linear models to support query processing, similarity search, compression, etc. Figure 4.11 shows the first few steps of the bottom-up segmentation algorithm. In the first step, the original time series of length n is replaced with an initial approximation of length $n/2$. Then, the cost of merging each pair of segments is computed and the pair with the lowest cost is merged. This procedure is repeated until a user-supplied threshold for the cost of merging is reached. The similarity between `RM0` and bottom-up segmentation lies in the procedure used to merge segments: they are both based on averaging the models of the segments. The algorithms have a number of key differences including different segment models (nonparametric vs. linear), and model comparison functions (L_1 distance vs. approximation error of combined model).

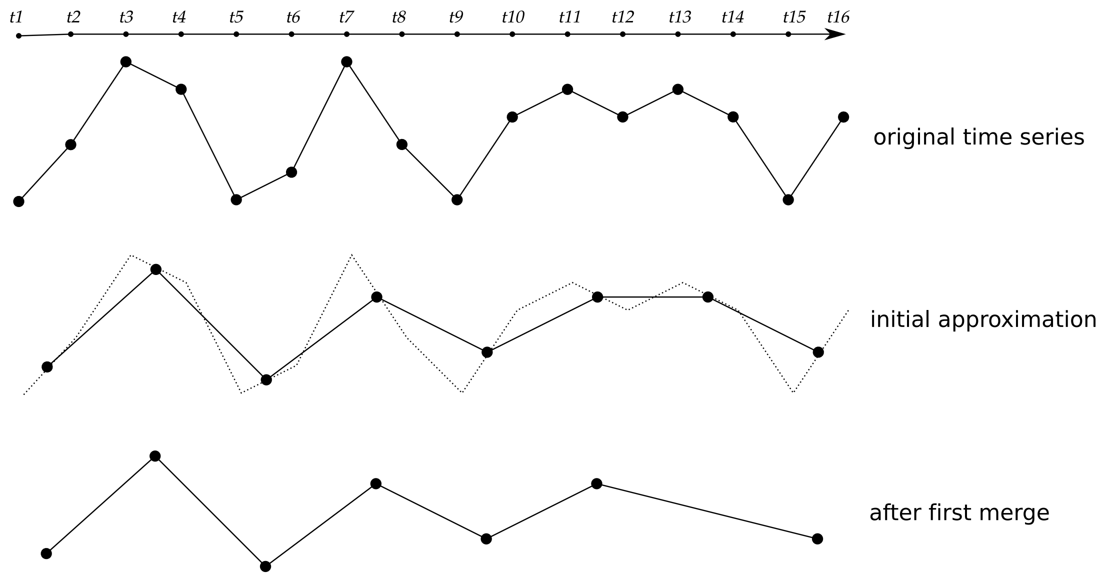


Figure 4.11: This figure shows the first few steps of the bottom-up segmentation algorithm [74] applied to a time series.

Strengths of RM0:

- The recursive merging algorithm has a built-in notion of *variability modeling*. In particular, the final score is computed taking the ratio of the maximum and the minimum distance between models. The intuition here is that when an input time series is highly variable in nature, large changes between neighboring values are unlikely to represent a real change in the process. This is a defining characteristic of RM0 as it addresses one of the key challenges in the domain: detecting change in spite of highly variable time series data.
- One of the most powerful features of the RM0 approach is its robustness to noisy and missing data. We will discuss these aspects further in Chapters 5 and 7.
- The approach is able to detect changes in both directions. Figure 4.10 shows an example of an increase in vegetation. Examples with a decrease are shown in later chapters.
- The algorithm is scalable to large global scale data sets.

A few of the limitations of the RM0 algorithm are discussed below:

- One of the drawbacks of the recursive merging approach is that, like other segmentation algorithms, the change point may not be identified accurately. This is a side effect of the segmentation process: by combining entire subsequences/segments, it is possible that the specific point of change occurs in the middle of the subsequence and thus may not be identifiable once segments are combined.
- The approach does not handle multiple change points directly. It is likely to assign a high score to a time series with multiple change points, but an extended algorithm is needed to handle multiple change points in a principled fashion.
- It is difficult to attach a physical interpretation to the change scores assigned by RM0.

As in the case of other segmentation approaches, the recursive merging algorithm has a number of key components. We discuss some of these components below:

How is a subsequence/segment modeled? The basic RM0 algorithm uses a non-parametric model represented by a vector of periodic means based on the underlying data. Other potential models include linear and polynomial models, or any model that supports the distance and merge operations.

How are two models compared? The L_1 distance (also called Manhattan distance) is used to compute the distance between two segments. This measure is called the L_1 distance because the distance is equal to the L_1 norm of the difference between vectors. The L_1 distance was chosen because it is less sensitive to noisy values than other distance measures. We will evaluate other distance measures in Chapter 7.

What is the search strategy? The algorithm follows a bottom-up search strategy, merging neighboring segments in an iterative fashion. Although the algorithm is conceptually recursive, it is also *tail recursive*, and thus can be implemented as an iterative algorithm.

How is the combination of two models computed? The combination of two segments is computed by taking the mean of the models. Other variations will be explored and evaluated in Chapter 7.

How is the final score assigned? The change score assigned to a time series is computed as the ratio of the maximum and minimum difference between models. This step also implicitly models a notion of *variability* of the time series.

4.5 Yearly Delta Algorithm

The yearly delta algorithm takes a predictive approach to the time series change detection problem. It essentially looks for discrepant subsequences in relation to a model of previous observations in the time series; this is related to EWMA-based algorithms and similar approaches discussed in 3.5.1. We define the following terms in the context of the yearly delta algorithm.

Definition 4.5.1 (Time Series Projection Model)**.** A time series projection model $\mathcal{M}$ describes and predicts the expected behavior of a future time series *subsequence*, i.e., given the time steps $\{i, i+1, \ldots, j\}$, $\mathcal{M}$ generates the projected values $\{\hat{t}_i, \hat{t}_{i+1}, \ldots, \hat{t}_j\}$. The model is trained on a window of previous observations $\{t_1, t_2, \ldots, t_{i-1}\}$. A weight vector $\mathbf{w}$ determines both the forgetting factor and the weights on the historical observations; $\mathbf{w}$ is defined such that $\sum \mathbf{w} = K$, where K is constant across the data set.

The basic yearly delta approach is as follows:

1. Build an initial projection *model* based on a short window at the beginning of the time series.
2. *Compare* a subsequent window to the model's projection and store the deviation.
3. *Grow* the model by incorporating the subsequent values.
4. Repeat steps 2 and 3, until the end of the time series is reached.
5. The *change score* is the maximum deviation observed.

In this dissertation, we will first present a specific instance of this algorithm that we call `YD0`; we also explore some variations of `YD0` in Chapter 7. We discuss some of the key components of the yearly delta approach below, as well as specific design decisions of the `YD0` algorithm.

How to build the model? The model is essentially a summary of the values that have been observed until the current point in the time series. The model can be parametric or nonparametric. `YD0` uses the mean as model.

How the model is grown? What is the forgetting factor? A key aspect of growing the model is the forgetting factor, i.e., how much importance does the algorithm attach to more recent values in relation to older values. `YD0` uses a step function as the forgetting factor, attaching equal importance to a recent window of values and discarding older values. We will examine variations of `YD0` which consider different forgetting factors in Chapter 7.

How to compare a subsequence to the model? Depending on how the model is maintained, there are different ways in which a subsequence can be compared to the model. This step is of critical importance since it is the metric used to measure deviation from the model that is assigned as the final change score. `YD0` uses the difference of the mean of the subsequence with the model as the metric to measure deviation.

We discuss some of the strengths of the `YD0` algorithm below:

- The algorithm can detect the time of change close to the actual change with accuracy up to the temporal resolution of the input data set, although postprocessing may be required to obtain the most accurate estimate.
- A powerful feature of the `YD0` approach is its robustness to missing data. We will discuss robust extensions to the algorithm in Chapter 5.
- The change score is tied to the difference in the integrated annual vegetation and therefore has a physical interpretation which is helpful to domain scientists.
- The algorithm is scalable to large global scale data sets.

Figure 4.12 illustrates how `YD0` works on a sample time series. Although the time series has noisy observations, and an inconsistent annual periodic cycle, the scheme assigns the minimum change score where the true change has occurred.

We discuss some of the limitations of the `YD0` algorithm below:

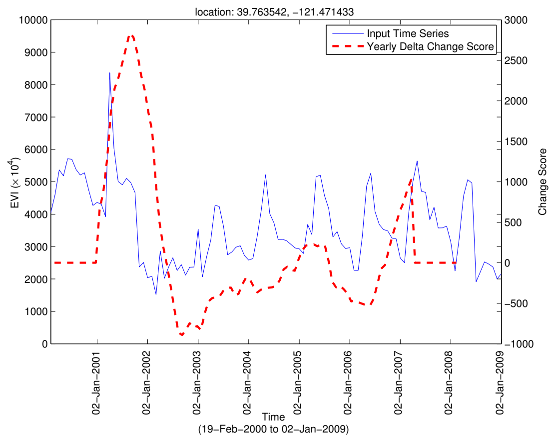


Figure 4.12: This plot shows how YD0 works on a sample time series. The time series has noisy observations, and an inconsistent annual periodic cycle, but the scheme assigns the minimum change score where the true change has occurred. This figure also illustrates one of the drawbacks of the scheme: if the changes being detected and a noisy observation are in the same direction, the scheme will confuse the two.

- YD0 is sensitive to symmetric noise (noise that is not primarily skewed in the positive or negative direction). In particular, when changes are being detected in both directions, the scheme can confuse high scores due to real change with those due to noise. This behavior can be observed in Figure 4.12.

- Like CUSUM, this approach is constrained in its ability to detect multiple changes and changes in both directions.

Chapter 5

Missing and Noisy Data

The analysis of real world data frequently involves considering how to deal with *missing* and *noisy* data. There are several factors that lead to contaminated observations in a data set, from the data generation/collection stage to the data transmission stage to the data storage stage. For instance, in earth science, these factors can include sensor error, environmental interference, etc.

The distinction between missing and noisy data is important: *missing* data refers to observations that are absent in the data set (usually at the desired level of quality) and are indicated with a special value; on the other hand *noisy* data are observations that are spurious in that they are too disparate from the true value to be useful without correction. Missing data are also called incomplete or unknown data. Noisy observations are also called outliers, anomalies, or spurious values. This important distinction plays a role in the design of approaches to handling missing/noisy data: missing data is clearly identified in the data set, while noisy data must be detected before its effect can be mitigated.

Time series data sets in earth science are often contaminated with noisy and missing values. The source of contamination for remote sensing data can be from atmospheric interference (due to clouds, aerosols, etc.), geometric distortion and other factors [108]. For example, Figure 5.1 shows the 250m EVI time series for a pixel near the city of Rancho Cucamonga, California; there are several missing values that occur both contiguously and sporadically. In addition, as indicated by the arrows in the figure, there are a number of values that are obviously noisy.

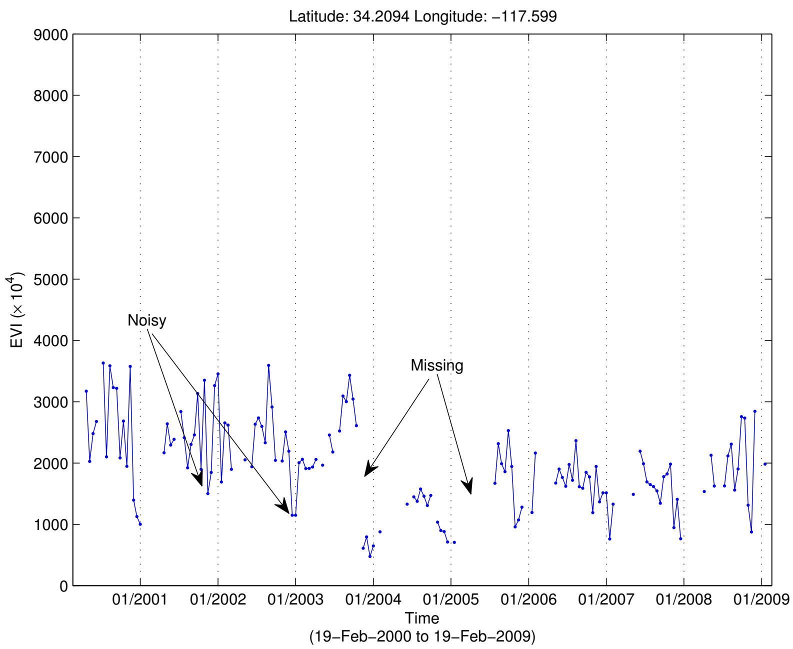


Figure 5.1: An example of a time series for a pixel near Rancho Cucamonga, California.

The problem of contaminated data does not affect all regions in a uniform fashion. In northern latitudes, most missing values tend to occur in the winter when snow on the ground interferes with the satellite's measurement of the green light being reflected. On the other hand, tropical regions (particularly the region surrounding the equator between 10°S and 10°N) frequently experience persistent cloud cover throughout the year. As a result, the problem of contaminated data is much more severe in the tropics. For example, Figure 5.2 shows the percentage of valid high quality data for each month in California and in the Democratic Republic of the Congo (DR Congo). It can be observed that while 70–85% of the data for California is of high quality for most months, there are several months for which only 10–20% of the data is of high quality for DR Congo.

The original vegetation index data product contains quality information for every pixel across all observations in time. This quality tag is based on sources of contamination that the satellite sensors can detect (e.g. clouds). When these sources of contamination are present for a particular observation, the quality tag marks the observation as uncertain (there are several degrees of uncertainty). However, there are frequently noisy observations for which there is no uncertainty according to the quality tag. These

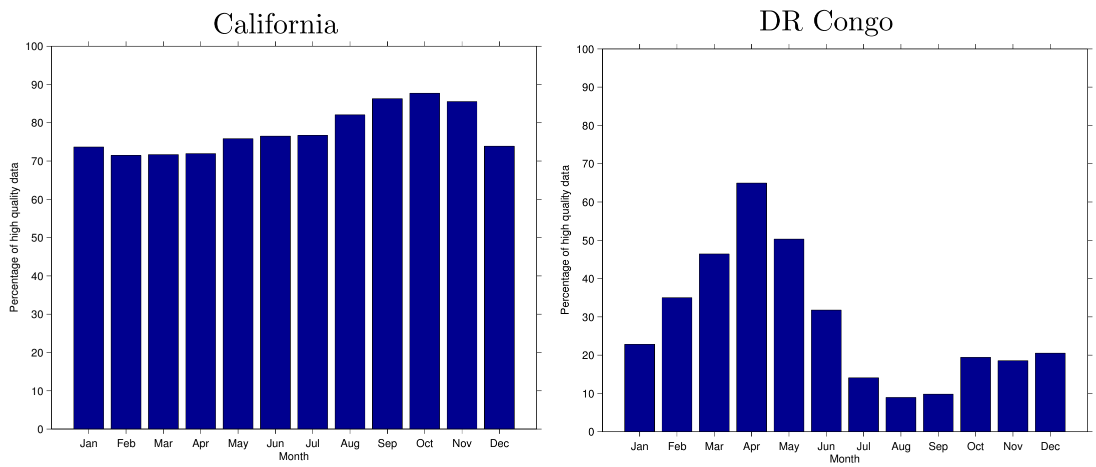


Figure 5.2: Percentage of high quality data in California (left) and DR Congo (right). Data is EVI index for the tile that includes the corresponding region. Percentages are averages for each month from Feb 2000 to Feb 2009.

observations are found when the satellite sensor is unable to detect the contaminating source, or other measurement/transmission errors.

In this chapter, we will outline four distinct approaches for dealing with missing/noisy data. We will introduce generic techniques for all approaches, but the discussion will focus primarily on techniques for time series data.

Estimate missing observations In this approach, the goal is to estimate missing observations using information that is available in the remaining data set. Some techniques following this approach also compute the uncertainty about the estimated observation.

Noise extraction This approach assumes that the input data set has a signal with some amount of noise. The goal is then to reduce the effect of noise. Given an input time series, some techniques simply detect individual observations that are noisy, while others produce an entire time series for which the effect of noise has been reduced.

Smoothing approaches This approach addresses both problems, missing and noisy data simultaneously. The assumption here is that *all* observations have some amount of noise. Therefore, the goal is to reduce the amount of noise in the overall data set and fill-in any missing values, thereby “smoothing” the data.

Robust algorithms In this approach, the input data is assumed to contain some missing and noisy data. The goal is to construct robust algorithms that can be used for analysis despite the presence of contaminated observations. In contrast to the other approaches, this approach is not a preprocessing step.

In this dissertation we evaluate the robust algorithms presented in this chapter, but we do not evaluate the other three approaches outlined.

5.1 Estimating Missing Observations

One approach for addressing missing data is to simply discard the data instances for which there are missing attributes. However, this procedure may introduce bias into the data set if the missing values do not occur at random; additionally, there may not be enough data remaining if most of the data instances have even one or two missing values. Accordingly, there is a large literature on performing statistical analysis in the presence of missing data (the book by Little and Rubin [82] provides a broad overview). The goal of techniques in this category is to estimate missing observations using information that is available in the remaining data set. Some types of data, such as those produced from population surveys and longitudinal studies, are especially prone to missing values and the respective fields have considered the missing value problem extensively.

A key question when performing statistical analysis in the presence of missing data is whether there is a pattern to the missing values, i.e. are the values missing at random, or are the values missing in a systematic fashion across attributes or instances (many records are missing the same attribute/many records are missing most attributes)? The process which caused missing observations is known as the *mechanism* that led to missing data (e.g. many respondents may not answer a question about income in a survey). Ignoring the missing data mechanism when data is not missing at random could lead to bias in the inference. Therefore, close attention is paid to the mechanism that caused the missing values.

In this section we will discuss two types of techniques for estimating missing observations: interpolation and imputation. Broadly speaking, *interpolation* techniques are computationally inexpensive and attempt to obtain a good fit/shape to the estimated data; *imputation* techniques, on the other hand, are more "principled" in that they pay attention to aspects such as the distribution of the data to ensure that the imputed value is plausible, and the uncertainty of estimation is computed. As a result, imputation techniques are much more computationally expensive than interpolation techniques.

5.1.1 Interpolation

Interpolation is typically applied to curves in 2-dimensional data and is thus particularly well-suited for time series data. Given a data set with missing values, interpolation techniques return the exact value where the data is known and an *interpolated* value where the data is missing. The following are well-known interpolation techniques:

Nearest neighbor interpolation This simple technique returns the nearest available neighbor for a given value. The advantage of this technique is that it is very fast, but the estimates are likely to be inaccurate for time series which exhibit periodic behavior.

Linear interpolation The linear interpolation technique fits a line between the available points. Missing values are then estimated by computing the value of the interpolant at the missing time point. For time series data, linear interpolation is slightly more expensive than nearest neighbor interpolation but is likely to provide a more accurate estimate.

Polynomial interpolation Polynomial interpolation attempts to fit a polynomial that passes through the available data. Given a time series with T data points, a unique polynomial (called the interpolating polynomial) of degree $\leq T-1$ exists that passes through all T points. Polynomials have some advantages (including differentiability and smoothness), but high-degree polynomials are very oscillatory in nature (known as Runge's phenomenon) and they are relatively expensive to compute.

Spline interpolation Spline interpolation uses a piecewise polynomial which is fit across the available data points (called knots). The piecewise polynomial can be linear or quadratic, but cubic splines are most commonly used in practice. Splines have some of the advantages of both polynomial and linear interpolation. They are smooth like polynomials, but because they are piecewise like linear interpolants, they do not exhibit the oscillatory behavior of polynomials. Splines see tremendous practical use in the fields of signal processing, image processing, computer graphics and computer aided geometric design (CAGD). There are multiple spline techniques (e.g. B-splines, cubic Hermite splines, etc.) that offer relative advantages in computational efficiency and numerical stability. The book by Schumaker [109] provides a comprehensive overview of splines.

5.1.2 Imputation

There are two very commonly used approaches for imputing missing data: multiple imputation, and imputing using maximum likelihood estimation (MLE). A broad survey of the area was provided by Horton and Kleinman [62].

Multiple imputation Multiple imputation is a Monte Carlo-based technique that essentially assumes a (user-defined) probability model (usually a regression model) for the data and runs repeated simulations to estimate a missing value. The book by Rubin [107] discusses the general topic of multiple imputation, while Honaker and King [61] present an approach in the context of time series data.

Maximum likelihood estimation Imputation using maximum likelihood estimation assumes that the data set under consideration consists of covariates (which may have missing values) in addition to a dependent variable. The goal is to estimate the distribution of the covariates, thereby obtaining estimates of the missing values. The distribution is typically estimated using the EM (expectation-maximization) algorithm [35]. A time series data set from earth science (where data is seasonal), can be transformed into one suitable for MLE imputation by treating each month (for monthly data) as a variable; i.e. upon transformation the data set will have 12 variables. The imputation can be performed in a sequence of steps, treating one variable as the target and the remaining eleven as predictors. Another approach is to impose a time series model (such as an ARMA model) and to then perform ML estimation on this model [70].

There are a number of imputation techniques that have been proposed in the time series forecasting literature [24]. The general problem of time series forecasting focuses on building a model and extrapolating future time steps. This problem setup can be extended to estimate missing values in the middle as well as at the end of a time series. Golyandina and Osipov [46] propose a technique based on singular spectrum analysis (SSA) to estimate missing data.

5.2 Removing Noisy Data

Noise removal is an important topic of study in several real-world domains, especially when data is measured using sensors. The general approach is to assume that the input data set has a signal with some amount of noise. The goal of noise extraction techniques is to reduce the effect of noise so that its effect on subsequent data analysis is mitigated. In this section, we will describe approaches relevant for vegetation index time series data. Anomaly detection [22], which deals with techniques to detect "unusual" observations, is a broader topic that addresses noise as one type of anomaly (while noise is a nuisance, other types of anomalies are actually observations of interest). The dissertation by Chandola [20] discusses approaches to the broader problems of anomaly detection for symbolic sequences and time series data. The signal processing domain has also extensively considered the noise reduction problem [122] but these are primarily domain-specific and not generally applicable to time series data.

In this section, we describe two major classes of techniques for addressing the problem of removing noise from time series data: (i) noise removal, and (ii) noise reduction. One of the biggest disadvantages of applying noise reduction techniques is that the change point may be lost or its sharpness diminished because its value differs significantly from neighboring values. Therefore, for such techniques to be effective in the change detection setting, they must be cognizant that change points may appear to be very similar to noise observations but should not be treated as such.

5.2.1 Noise Removal

Noise removal techniques detect and remove noisy observations from a given input time series. The resulting time series then has missing values in place of noisy observations.

These techniques operate under the assumption that noisy values are so markedly different from the other observations that they have no useful information.

In the sequence anomaly detection literature [20], this problem is formulated as one of detecting an anomalous subsequence within a time series (also known as *discord detection*). The main disadvantage of the noise removal approach is that since noisy observations are completely discarded, one must subsequently estimate their values to obtain a complete time series.

5.2.2 Noise Reduction

In contrast to noise removal techniques, noise reduction techniques assume that even noisy observations in a time series may contain some useful information. In particular, noise reduction techniques can take advantage of the *context* in which a noisy observation occurs and then appropriately utilize the actual value of the observation.

In a recent paper, Hird and McDermid [59] performed a thorough comparative empirical evaluation of six noise reduction techniques for NDVI (a vegetation index) time series. The authors first grouped the time series into six clusters based on natural regions of the study area (Alberta, Canada in this case). The cluster centroids were then used as the representative *noise-free* time series for each natural region. Six state-of-the-art noise reduction techniques were studied. Five out of the six were specifically designed for NDVI, though all were previously used in studies with NDVI data. Different levels of noise (10%, 40%, and 70%) were introduced. The evaluation criteria were RMSE (root mean square error), and a set of domain specific metrics (e.g. length of growing season, amplitude, etc.). The baseline approach of “do nothing” was also compared with the six techniques. The results indicated that the asymmetric Gaussian and double logistic function techniques performed best overall. Hird and McDermid [59] also provided a discussion of factors affecting performance, including the ability to account for negatively biased noise and differences based on the natural regions. Interestingly, it was found that applying noise reduction when the level of noise is low can actually *degrade* performance.

Based on the study by Hird and McDermid [59], it is clear that the choice of noise reduction must be made carefully, with attention paid to the level of noise and the underlying type of land cover.

5.3 Smoothing Approaches

Smoothing approaches address both the missing and noisy data problems simultaneously. These techniques assume that all observations have some amount of noise. Therefore, the goal is to reduce the amount of noise in the overall data set and fill-in any missing values, thereby smoothing the data.

There are a large number of techniques that can be described as the *function fitting* class of algorithms. These techniques try to fit well-known functions to the input data, and use the function to produce a smoother time series that has less noise and is free of missing values. We discuss a number of techniques that have been proposed primarily for ecosystem data. Roerink et al. [104] and Lunetta et al. [87] proposed function fitting techniques based on the Fast Fourier Transform (FFT). Chen et al. [29] describe an approach based on polynomial functions that applied to NDVI data using a least squares fit. An approach by Viovy et al. [124] uses sliding linear models as the smoothing function; this approach also takes into account the periodic nature of ecosystem data. Techniques for more general data that treat the smoothing problem as one of estimating a random process using a state space model were proposed by Shumway and Stoffer [115] and Jong [71].

5.4 Robust Algorithms

In this dissertation, we follow the approach of designing robust algorithms to address the problem of noisy and missing data. In this section, we propose extensions to two of our change detection algorithms which make them robust in the presence of noise and missing data. We then perform an experimental evaluation and provide a discussion of the factors affecting performance. There are a number of advantages of following the approach of making algorithms robust:

1. Since each time series does not need to be preprocessed, as is the case with the previous approaches, this approach is very computationally efficient.

2. There is no risk of the change point being discarded as noise.

3. One does not need to optimize the approach to account for regional differences,

which can be cumbersome for global studies.

4. All the available information is used, unlike some approaches which may discard noisy observations that are still useful.

The robust algorithms approach essentially assumes that the input data has noise and missing values at varying levels and that the change detection algorithm must account for this in the process of detecting changes. In principle, to detect significant changes such as forest fires, one may need only a few months of valid data per year. Depending on the approach taken, some algorithms (e.g. CUSUM) may not have natural extensions that make them robust. In addition, algorithms must have a notion of "too much" noise/missing data, when there is insufficient information to detect a change. The primary drawback of this approach is that it complicates the interpretation of the change events detected, when there are numerous missing/noisy values.

5.4.1 Robust Recursive Merging Algorithm

In this section, we describe an extension of `RM0` (described in Section 4.4) that is robust to missing data. The key insight in the robust `RM0` extension is that yearly windows that are similar will remain similar even in the presence of missing values (the algorithm is indifferent to patterns of missingness). The algorithm also makes an assumption that if there is noise, it is uniformly distributed across the time series (i.e. there is no pattern to where such values occur).

The time series for each location is processed as follows:

1. Step 1 remains the same as in the case of absence of missing values. Suppose b_{i1} and b_{i2} are the two most similar annual cycles and each has one missing value. If there are missing values, there are two cases possible: (1) both b_{i1} and b_{i2} have missing values in the same time step (say time step f), and (2) b_{i1} and b_{i2} have missing values in different time steps (say time steps f and g respectively). In the first case, the merged annual cycle $\frac{b_{i1}+b_{i2}}{2}$ also has a missing value in the time step, while the other values are averaged. In the second case, the merged annual cycle $\frac{b_{i1}+b_{i2}}{2}$ is given the value of the annual cycle that is present; time step g from b_{i2} and time step f from b_{i1}.

2. The recursive step remains the same.

3. The change score for this location is based on whether any of the observed distances are extreme. This can be quantified by computing the quantity $\frac{\max\{s_1,\ldots,s_{Y-1}\}}{\min\{s_1,\ldots,s_{Y-1}\}}$.

The algorithm does not explicitly address noisy observations. This issue is dealt with by the implicit variance modeling aspect of the algorithm: the assumption that noise is uniformly distributed means that the variance of each yearly subsequence increases by a similar factor, proportional to the amount of noise. Therefore, even though the distance comparison between annual cycles is impacted by the presence of noise, this increase in distance is normalized by variance modeling.

5.4.2 Robust Yearly Delta Algorithm

The robust yearly delta algorithm is an extension to the YD0 algorithm described in Section 4.5. The robust extension proceeds as follows:

1. Build an initial projection model based on a short window at the beginning of the time series. If there are missing values, these are stored with a special indicator.

2. Compare a subsequent window $\{t_i, t_{i+1}, \ldots, t_j\}$ to the model's projection $\{\hat{t}_i, \hat{t}_{i+1}, \ldots, \hat{t}_j\}$ and store the deviation. The deviation is computed only in terms of the values that are present in *both* $\{t_i, t_{i+1}, \ldots, t_j\}$ and $\{\hat{t}_i, \hat{t}_{i+1}, \ldots, \hat{t}_j\}$. Since YD0 uses the difference of the mean as the deviation function, the value is normalized to account for missing values.

3. *Grow* the model by incorporating the subsequent values. If any of the subsequent values are missing, they are stored with a special indicator.

4. Repeat steps 2 and 3, until the end of the time series is reached.

5. The *change score* is the maximum deviation observed.

The key operation in the robust yearly delta algorithm is in how a subsequence is compared to the projection when some of the values are missing. By performing this operation only on those values that are present, the deviation function is not biased by the missing data. Additionally, since `YD0` uses the difference of the mean as the deviation function, the value is normalized to account for missing values. The robust yearly delta algorithm does not account for noise; although this does not significantly affect its performance for some types of changes (like forest fires), it is one of the constraints that limits its ability to detect multiple types of changes.

Chapter 6

Evaluation I: Illustrative Examples

In this chapter, we provide several illustrative examples of the kinds of land cover changes that can be detected by the `RM0` algorithm. These examples illustrate its ability to handle a wide variety of changes. We focus this analysis on two separate data sets: the first data set includes only the San Francisco Bay Area, while the second includes the entire state of California. Due to high population growth in recent years, California has experienced many land use changes, such as the urbanization of farmland and desert converted to farmland. The state also has many forest fires each year that can lead to temporary changes in the land cover.

6.1 Data

We preprocessed the data to eliminate poor-quality measurements in order to simplify evaluation. Data cleaning was done by performing the following steps:

1. The MODIS data sets are tagged with a quality assurance (QA) flag which is used to describe atmospheric and sensor conditions when the measurement was taken. We used the QA flag to remove all measurements that were tagged as being of low quality. Another filtering step recommended by earth science domain experts was the removal of measurements of EVI above 0.9.

2. We also discarded any locations that contained missing data. Therefore, the data for a location is retained only if the entire time series is available with no missing values and no low quality data.

The final quality-filtered EVI data set for the San Francisco Bay Area contained 180,400 locations (covering a region of 100 miles $\times$ 50 miles), the entire California data set contained over 5 million locations (covering a region of 800 miles $\times$ 200 miles), and the data set for forest locations in the state of California contained 380,285 locations. The length of the time series for all three data sets is 76, corresponding to 6 years and 4 months of monthly data from February 2000 through May 2006.

6.2 Analysis of the San Francisco Bay Area

We initially performed a clustering of the EVI data for the San Francisco Bay Area region in order to observe the high-level characteristics of the vegetation data. We used the k-means algorithm to cluster the EVI data to produce a minimum number of distinct centroids (in this case five centroids were found to have the optimal SSE). Figure 6.1 shows the cluster centroids for the Bay Area EVI data. What we observed was that most of the data was predominantly from 5 classes of vegetation thereby implying that for the vast majority of the points in the data set, there is no change in land cover. Additionally, when the clusters are viewed on a map and compared to satellite imagery, the clusters corresponded to the actual vegetation in the region very closely.

We then applied the recursive merging change detection algorithm to the Bay Area EVI data set. The output from the algorithm is a list of change scores, one for each location. Figure 6.2 shows a histogram of the change scores obtained. We observe that the distribution of the scores conforms to our expectation based on the clustering results that most of the locations in the data set do not exhibit a change. We manually examined the top 31 points which had a change score greater than a threshold of 8. Of these, 22 points were found to correspond to interesting land cover changes and others corresponded to changing crop patterns in farmland. A majority of the points with change scores less that 8 (but greater than 4) were from farmland. We briefly discuss a few of the interesting land cover changes discovered:

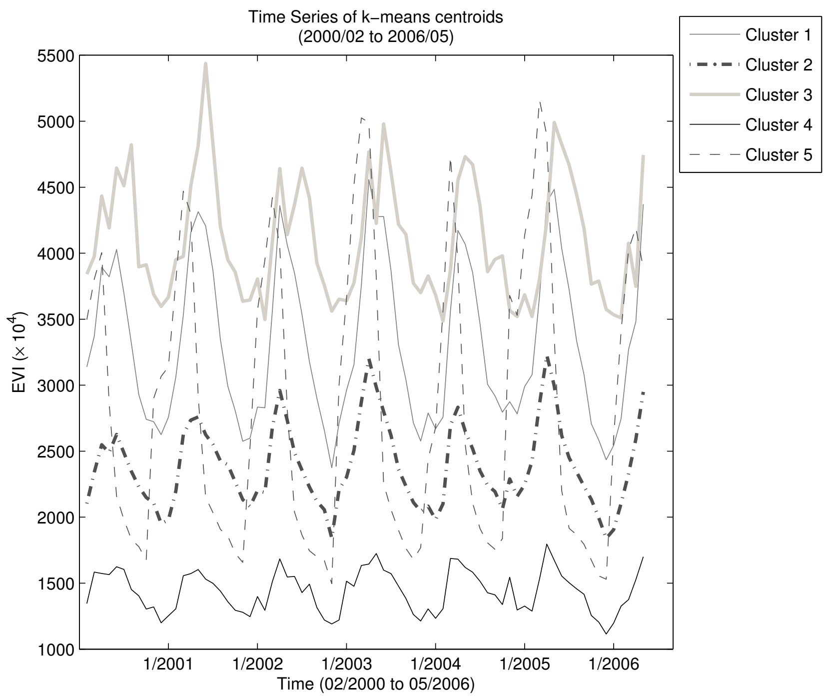


Figure 6.1: k-means cluster centroids for the Bay Area EVI data set. The land cover type corresponding to each cluster is as follows: Cluster 1 is high seasonal biomass density, moderate interannual variability (shrub cover); cluster 2 is moderate annual biomass density, moderate interannual variability (grass cover); cluster 3 is high annual biomass density, low interannual variability (evergreen tree cover); cluster 4 is low annual biomass density, low interannual variability (urbanized cover); cluster 5 is high seasonal biomass density, high interannual variability (agricultural cover).

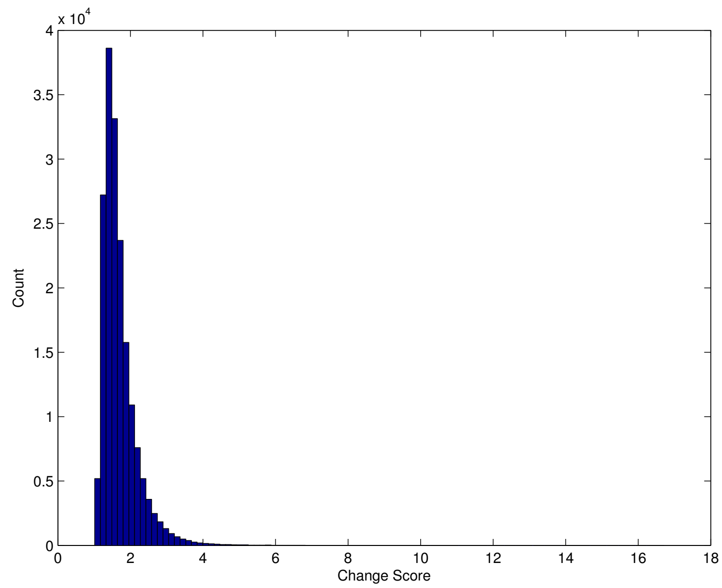
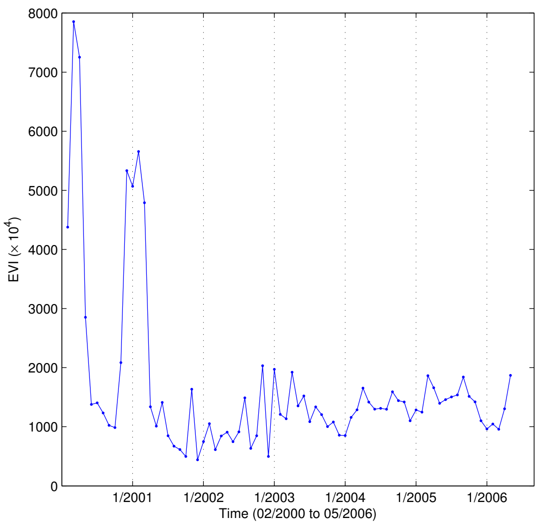


Figure 6.2: Histogram of change scores produced by the recursive merging algorithm for the Bay Area data set.

1. *Barren land to golf course.* Figure 4.10 shows the time series and map for a location where a golf course was built. Specifically, this location corresponds to the Metropolitan Golf Links in Oakland, CA which was built in 2003 on a site which was previously a disposal site for dredged material from San Francisco Bay. The time series for the location clearly shows the low level of vegetation at the location prior to 2003, after which the vegetation is relatively uniform, consistent with what is expected at a golf course that is watered throughout the year.

2. *Construction of a housing subdivision.* Figure 6.3 shows the time series and map for a location where a subdivision is under construction in Hayward, CA on a site which was previously vegetated, with the level of vegetation suggesting that this land may have been grassland prior to the construction. Interestingly, the construction of housing can actually lead to the increase in vegetation at a location such as this one since after the construction period, the area is typically planted with lawns and trees. In such situations, land cover change detection techniques that examine only the beginning and end of a time series may fail to detect this



Source: Google Maps.

Figure 6.3: Construction of a subdivision in Hayward.

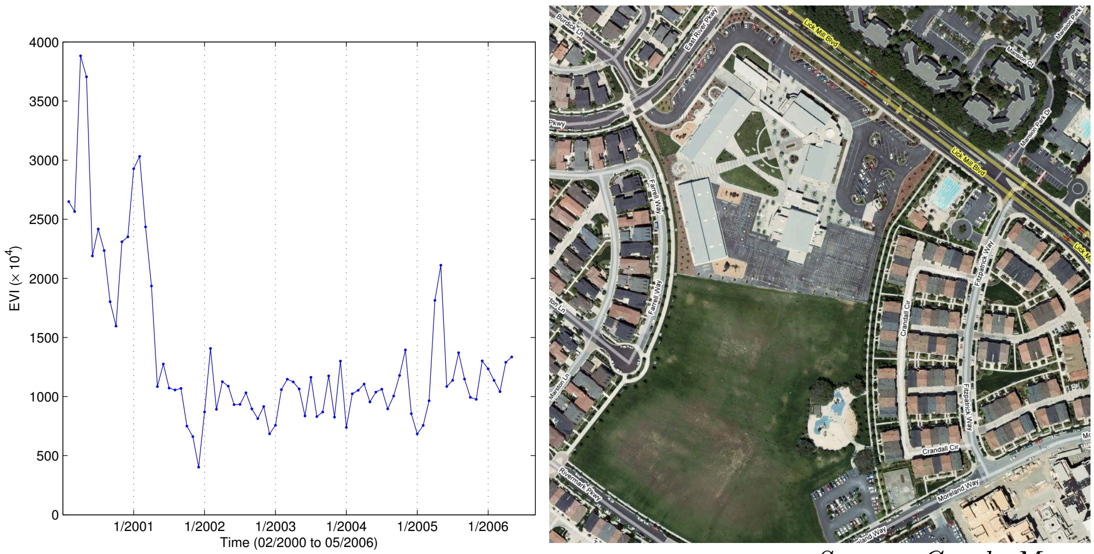
type of change.

3. *Construction of a shopping center.* Figure 6.4 shows the time series and map for a location where a shopping center has been built on a site which was previously vegetated in San Jose, CA. The year of construction for this shopping center was 2002, which is also when the change in vegetation-level occurs in the time series.

4. *Construction of a golf course.* Figure 6.5 shows the time series and map for a location where a golf course was built. This is the location of the Golf Club at Boulder Ridge in San Jose, CA. This golf course was constructed in 2001, and this is clearly seen in the time series for the location.

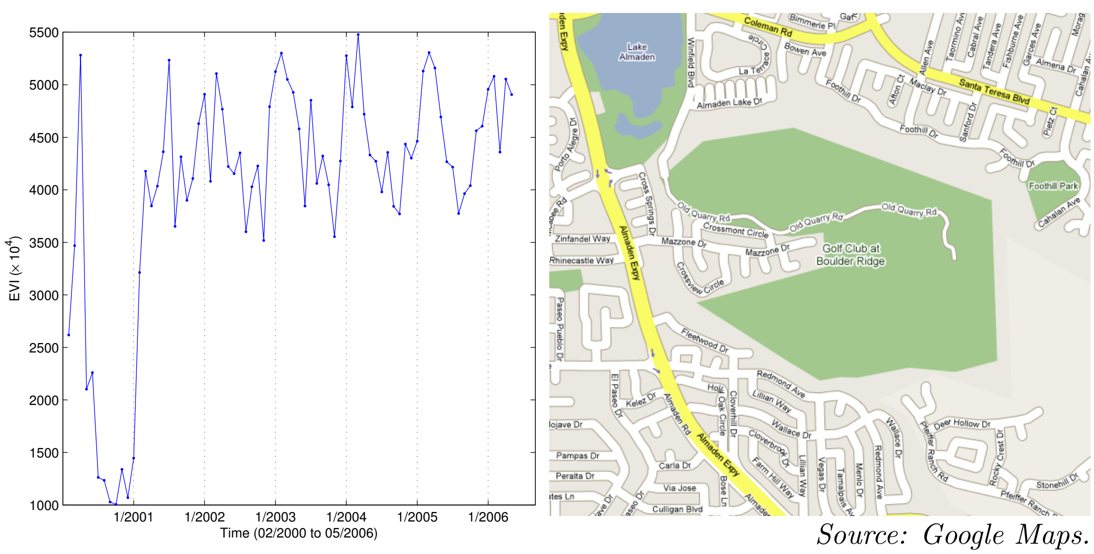
5. *Construction of a shopping center.* Figure 6.6 shows the time series and map for a location where a shopping center has been built on a site which was previously vegetated. This location corresponds to the Pacific Commons shopping center in Fremont, CA.

These results show the effectiveness of the recursive merging algorithm on the Bay Area data set. Specifically, in examining the top ranked change locations from the algorithm, we observe that a variety of interesting land cover changes are detected. The



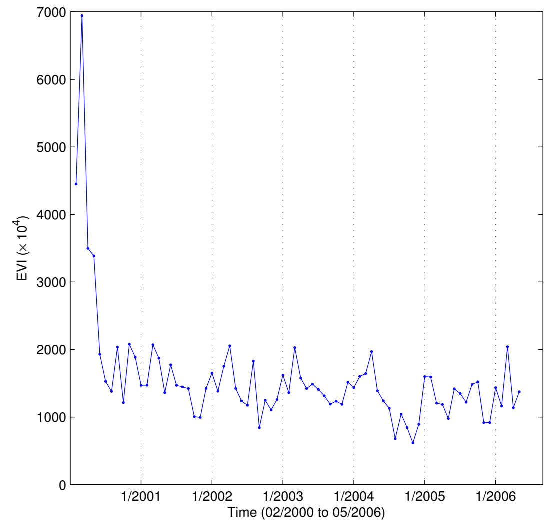
Source: Google Maps.

Figure 6.4: Construction of a shopping center in San Jose.



Source: Google Maps.

Figure 6.5: Construction of a golf course in San Jose.



Source: Google Maps.

Figure 6.6: Construction of the Pacific Commons shopping center in Fremont.

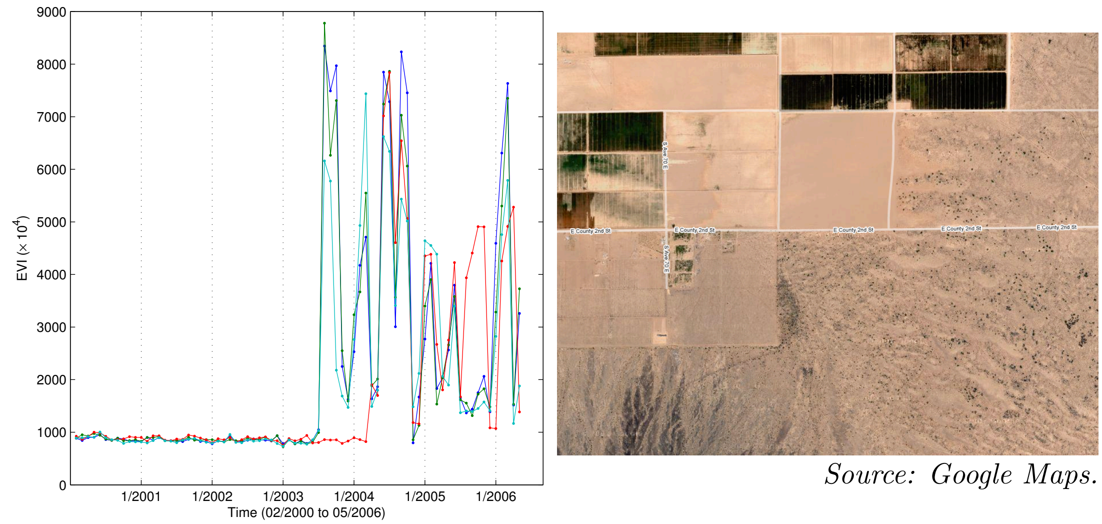
ability of the algorithm to discount natural seasonal changes (such as those seen in Figure 6.1) is particularly appealing.

6.3 Analysis of the entire state of California

We performed an extended study on the entire state of California, which involves about a 30-fold increase in data over the Bay Area. For this data set we found 2,833 locations that had a change score greater than a threshold of 15, a vast majority of which corresponded to land cover change points. We analyzed about 1,000 of the top change locations for the larger California data set and we found that more types of changes were detected than in the Bay Area data set. In particular, we found numerous instances where the land cover had changed from desert to farmland, forests to barren (due to forest fires), farmland to housing subdivisions, and desert to golf courses.

A few interesting land cover changes that were discovered in the California data set are shown in Figures 6.7, 6.8 and 6.9 (note that the figures show several time series each; these correspond to groups of spatially close locations) and described below:

1. *Desert to farmland.* This is a group of points corresponding to a change from a desert location to farmland in southern California. This type of land cover change



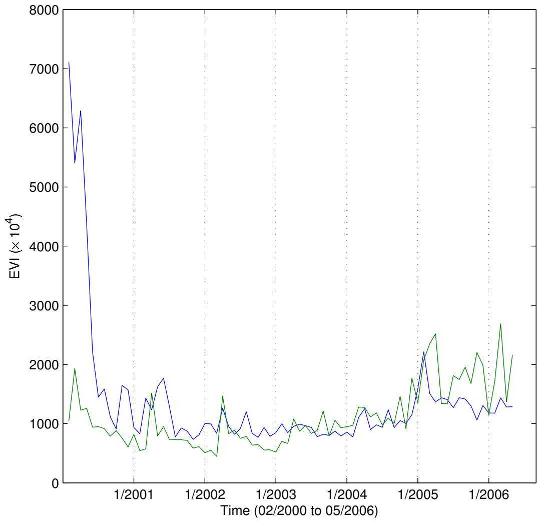
Source: Google Maps.

Figure 6.7: Desert to farmland.

is prevalent in California and across the United States [6, 118].

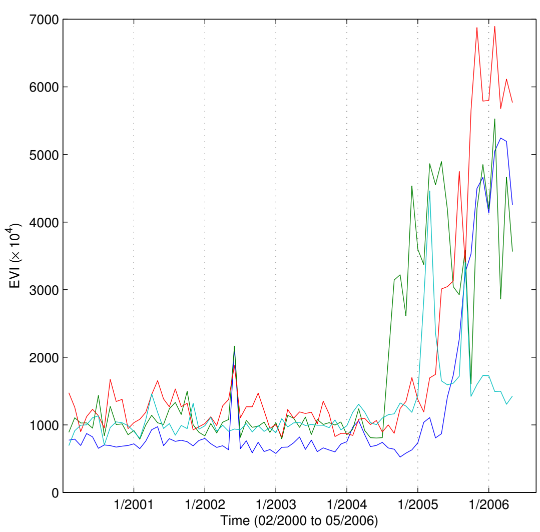
2. *Farmland to subdivision.* This is a location in Sacramento, CA where farmland has been cleared and a housing subdivision is being built. This location is in the north-west outskirts of the city, where the surrounding land currently consists of farms. This is an example of land cover change where urbanization is converting agricultural land into housing.

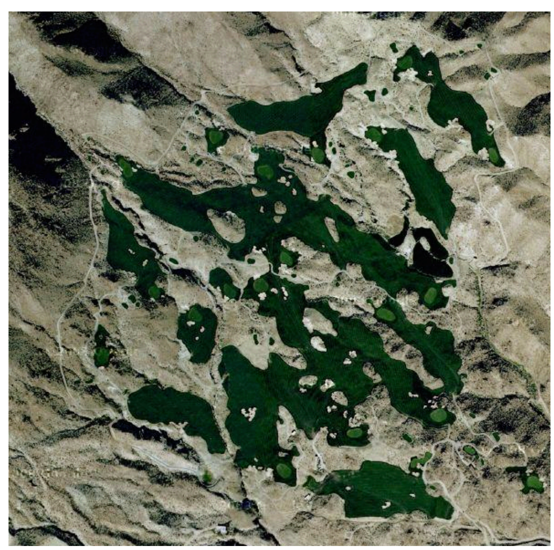
3. *Desert to golf course.* This is an example of a new golf course being built in Palm Desert, CA. This town has over 100 golf courses, putting intense pressure on the water supply [4] in a region where water is already scarce.



Source: Google Maps.

Figure 6.8: Farmland being converted into a housing subdivision in Sacramento.





Source: Google Maps.

Figure 6.9: A newly constructed golf course in Palm Desert.

Chapter 7

Evaluation II: Quantitative Experiments

In this chapter, we will present an experimental evaluation comparing the algorithms discussed in Chapters 4 and 5, their variations, and other well-known algorithms. Specifically, we will present a quantitative evaluation of change detection algorithms with different levels of noise and missing data, validating against *real-world* ground truth data.

This chapter begins by providing details of the evaluation process. In particular, we discuss the underlying time series data sets utilized, the ground truth data, and the evaluation methodology. Next, we explore the performance of the base recursive merging algorithm (RM0) and the yearly delta algorithm (YD0) in relation to a number of variations of the algorithms. Finally, we present a comparative evaluation of our proposed algorithms with other well-known change detection algorithms.

7.1 Background

The structure of the vegetation time series data used in this quantitative evaluation was previously discussed in Section 2.1.1. Here, we provide details of the specific regional subset we focused on, the data preprocessing steps, as well as the ground truth data.

7.1.1 Evaluation Data Sets

We use three evaluation data sets (called `DS1`, `DS2` and `DS3`), all of which consist of forest pixels in California. All the three data sets are derived from raw MODIS input data downloaded from the LP DAAC [3] covering the time period from February 2000 through January 2009. A land cover map obtained from the Ecosystem Modeling Group at NASA Ames Research Center was used to subset forest pixels.[1] The difference between the data sets is in the preprocessing steps: each of the data sets has a different level of noise and missing values.

`DS1` **(Highest quality data)** To create `DS1`, we preprocessed the data to eliminate poor-quality measurements by the following standard data cleaning procedures:

1. The MODIS data sets are tagged with a quality assurance (QA) flag which is used to describe atmospheric and sensor conditions under which the spectral measurements were taken. Therefore, there is a QA flag associated with every observation for every pixel. We used the QA flag to retain only those observations of good quality, removing all observations that were tagged as being of *marginal* or of *low quality*. Another filtering step performed (recommended by earth science domain experts) was the removal of EVI measurements less than or equal to 0 and above 0.9.

2. To reduce the impact of quality filtering, we converted the biweekly data to monthly data by averaging (using a simple mean) the available data for every month.

3. We then discarded all locations that contained any missing data. In other words, the data for a location is retained only if the entire time series is available with no missing values and no low quality data.

`DS2` **(Highest quality data with missing values)** `DS2` is quality-filtered in the same manner as step 1 for `DS1`, i.e. observations that are of low/marginal quality, or

[1] The following land cover classifications were considered forest: Evergreen Needleleaf, Evergreen Broadleaf, Deciduous Needleleaf, Deciduous Broadleaf Forest, Mixed Forests.

Data Set	# of pixels (N)	Frequency	Length of Time Series (T)	Noise	Missing Data
DS1	148,770	Monthly	108	Low level	No
DS2	787,710	16-day	207	Low level	Yes
DS3	787,777	16-day	207	High level	No

Table 7.1: Summary of evaluation data sets.

outside the range $[0, 0.9)$ are discarded. These observations are replaced with a special value to indicate a missing value. Thus, pixels which contain missing and invalid values are not discarded as in the case of DS1.

DS3 **(Unfiltered data)** DS3 consists of the raw data without any processing for quality, i.e., the quality flag is not examined and we do not filter observations outside the recommended valid range. Thus, DS3 has the highest level of noise of all the data sets.

The key characteristics and properties of the three data sets are summarized in Table 7.1. Note that by permitting missing and noisy values, there is more than a *five-fold* increase in the spatial coverage of the data sets.

7.1.2 Ground Truth Data

Change detection studies are frequently plagued by lack of good ground truth data [100], which forces the evaluation process to be more qualitative in nature. This frequently makes it difficult to objectively answer the question: *what is a change?* In this study, we have utilized high quality ground truth data generated by an independent source, and are thus able to perform an objective quantitative evaluation.

The state of California maintains historical databases of fire incidents that affect an area larger than a certain size. For each fire, there is detailed information on how it started, how it spread, how it was controlled, etc.; one of the key aspects of the database is that it contains accurate boundaries of the fires. We obtained fire boundaries for California for the fire seasons for the years 2000 through 2008 from the Ecosystem Modeling Group at NASA Ames Research Center. We use these fire boundaries as the ground truth data for validation purposes.

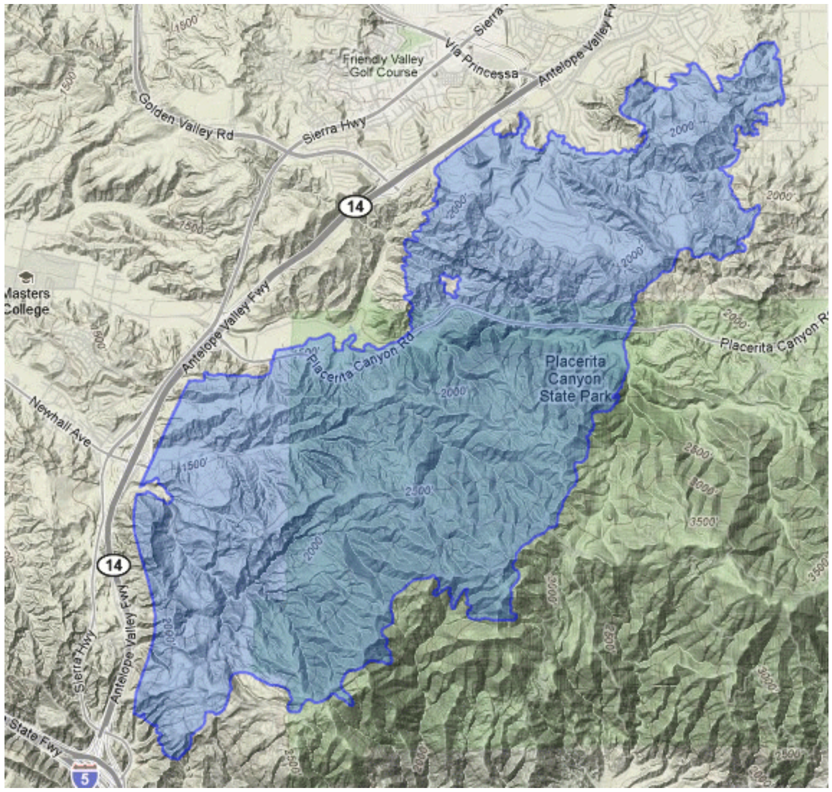


Figure 7.1: Example of a polygon representing the boundary of a fire. (Screenshot of Google Maps)

The ground truth data is in the form of *polygons* which represent the boundaries of forest fires. Each polygon $\mathcal{P}$ is a closed shape that consists of N sides (N is usually in the hundreds), with each vertex represented as a latitude/longitude coordinate pair, and may contain one or more holes $\mathcal{H}_i, i = 1 \ldots n$. A hole is also a polygon that is fully contained within its parent polygon. The boundary of an individual fire is then $\mathcal{P} \setminus \{\mathcal{H}_1 \cup \mathcal{H}_2 \cup \ldots \cup \mathcal{H}_n\}$. A example of a fire boundary is shown in Figure 7.1; the fire occurred in 2004 near Santa Clarita, CA. The blue filled region represents the polygon; the dark blue line is the outside boundary of the polygon, while a hole can be seen in the middle of the map.

		Predicted	
		Fire	No Fire
Ground Truth	Fire	TP	FN
	No Fire	FP	TN

Table 7.2: Confusion matrix.

7.1.3 Evaluation Methodology

Precision and *recall* are two well-known metrics used to evaluate the performance of algorithms in information retrieval, machine learning and data mining [119]. These terms refer to generic metrics that are used in a number of application areas ranging from bioinformatics to network intrusion detection; the algorithms evaluated also span a large variety including classification, web search, anomaly detection, etc. Table 7.2 shows a confusion matrix with the following terms:

True positive (TP) The number of positive instances that are correctly predicted by the model.

False negative (FN) The number of positive instances that are incorrectly predicted as negative.

False positive (FP) The number of negative instances that are incorrectly predicted as positive.

True negative (TN) The number of negative instances that are correctly predicted as negative.

In the context of classification, precision and recall are then defined as follows.

$$\begin{aligned}\text{Precision, } p &= \frac{TP}{TP+FP}\\ \text{Recall, } r &= \frac{TP}{TP+FN}\end{aligned}$$

Given a time series data set D with N pixels, let us assume that any change detection technique returns a list of *change scores* of length N, where each change score (with a one-to-one mapping to a pixel) is a measure of the degree of change for the corresponding pixel. We also have a ground truth data set which consists of the true labels of each of the pixels; let M be the *total* number of actual disturbances as determined by the ground truth. To evaluate the performance of a given change detection algorithm at rank n, we count the number of true disturbances in the top n portion of the sorted change scores of all the pixels, where n is the number of actual disturbances ($1 \leq n \leq M$). Let TP_n be the number of actual disturbances in the top n predicted disturbances, and FP_n be the number of pixels that are in the top n portion but are not actual disturbances.

We evaluate performance by examining the *sorted* list of change scores. Specifically, performance is measured in terms of the number of instances correctly identified and the number of instances missed in the top-n ranked instances. We use a precision metric (called p_n) employed in the context of information retrieval [7] and anomaly/outlier detection [20], which is appropriate for the top-n ranked setting, defined as follows.

$$\text{Precision,}\ \ p_n = \frac{TP_n}{TP_n + FP_n}$$

Recall (defined in the information retrieval context as the ratio of the number of relevant instances retrieved to the number of relevant instances that could have been retrieved) is defined as

$$\text{Recall,}\ \ r = \frac{TP_n}{M}$$

We compute TP_n by counting the number of pixels (in the top n) that lie within the boundaries of a fire polygon (the *intersection* of the points with the polygons); the procedure to determine whether a point lies inside a polygon follows an algorithm based on the *crossing number* test.

Note that we do not perform any post-processing steps in our evaluation procedure. These steps, which can range from filtering misclassified pixels (i.e. non-forest pixels such as farms are in the input data) to raising lower ranked points based on proximity to higher ranked points, can substantially improve the results. However, we do not apply post-processing steps so that performance differences can be directly attributed to algorithmic properties without other factors whose role may be difficult to discern.

7.2 Exploring Variations of the Recursive Merging Algorithm

In this section, we explore variations of a number of different aspects of the recursive merging algorithm. Specifically, the performance implications of various design decisions are discussed. In this section, we will evaluate the performance of the algorithms on `DS1` and `DS3` to understand differences under different noise levels; we do not evaluate on `DS2` in this experiment since we are not examining performance in the presence of missing values. Before we discuss the variations of the algorithm, let us define the following notation (specific to the approach):

$\{\Delta_1, \Delta_2, \ldots\}$	The list of distances between every pair of models that was merged.
Δ_M	The *maximum* distance between any pair of models that was merged.
Δ_m	The *minimum* distance between any pair of models that was merged.
Δ_n	The distance between the *last* pair of models that was merged. Note that this quantity will not always be equal to Δ_M.
Δ_1	The distance between the *first* pair of models that was merged. Note that this quantity will not always be equal to Δ_m.

Table 7.3: Notation for recursive merging approach.

RM_LAST_FIRST This variation assigns the final change score by taking the ratio of the *last* and *first* merge distances (instead of the minimum and maximum). The intuition behind this technique is that since Δ_n involves the least similar segments and Δ_1 involves the most similar segments, their ratio would be high when the segments are not similar.

$$\textit{change score} = \frac{\Delta_n}{\Delta_1}$$

RM_AVG This variation normalizes the change score by the *average* of the merge distances not including the maximum. The idea is to examine whether utilizing *all* of the merge distances in normalizing the change score improves performance. In particular, if the time series has a high degree of variability, it is likely to drive up the distances between all the merged segments.

$$\textit{change score} = \frac{\Delta_M}{\text{mean}(\{\Delta_1, \Delta_2, \ldots\} \setminus \Delta_M)}$$

RM_NO_NORM This scheme is a simplification of the original RM0 algorithm. In particular, the variability modeling component is removed, and the maximum merge distance is directly assigned as the change score.

$$\textit{change score} = \Delta_M$$

Figure 7.2 and 7.2 shows the results of each of the above variations on DS1 and DS3, respectively.

We observe that the RM0, RM_LAST_FIRST, RM_AVG algorithms have comparable performance, but RM_AVG performs best with the higher ranked pixels, while RM_LAST_FIRST performs best overall. We examined a sample of the false positives that *were* detected by RM0 but *not* by RM_AVG, in order to understand the difference in performance in the higher ranked pixels. A majority of these time series were similar to those in Figure 7.3. It can observed that both the time series in the figure have relatively low means, and that there is a temporary change in pattern at some point (year 2002 for the left pane and 2005-2006 for the right pane). RM0 assigned a high score to these time series because though the magnitude of the change is small, the pattern shift is large relative to the rest of the time series. On the other hand, since RM_AVG averages across all model distances, this effect is diminished. Thus, we see that RM0 is sensitive to *small* shifts in a stable time series with a low mean. This characteristic is useful for some problem settings (e.g., detecting drought damage), but leads to an increase in false positives when detecting fires. The performance of the RM_NO_NORM demonstrates the critical role played by variability modeling. In particular, we observe that RM_NO_NORM performs well overall on DS1, although it performs poorly on higher ranked pixels (which are primarily have high variability). However, with the noisy DS3 data set, we see the performance of RM_NO_NORM significantly.

7.3 Exploring Variations of the Yearly Delta Algorithm

We explore variations of a number of different aspects of the yearly algorithm in this section. We will evaluate the performance of the variations on DS1 and DS3 to understand

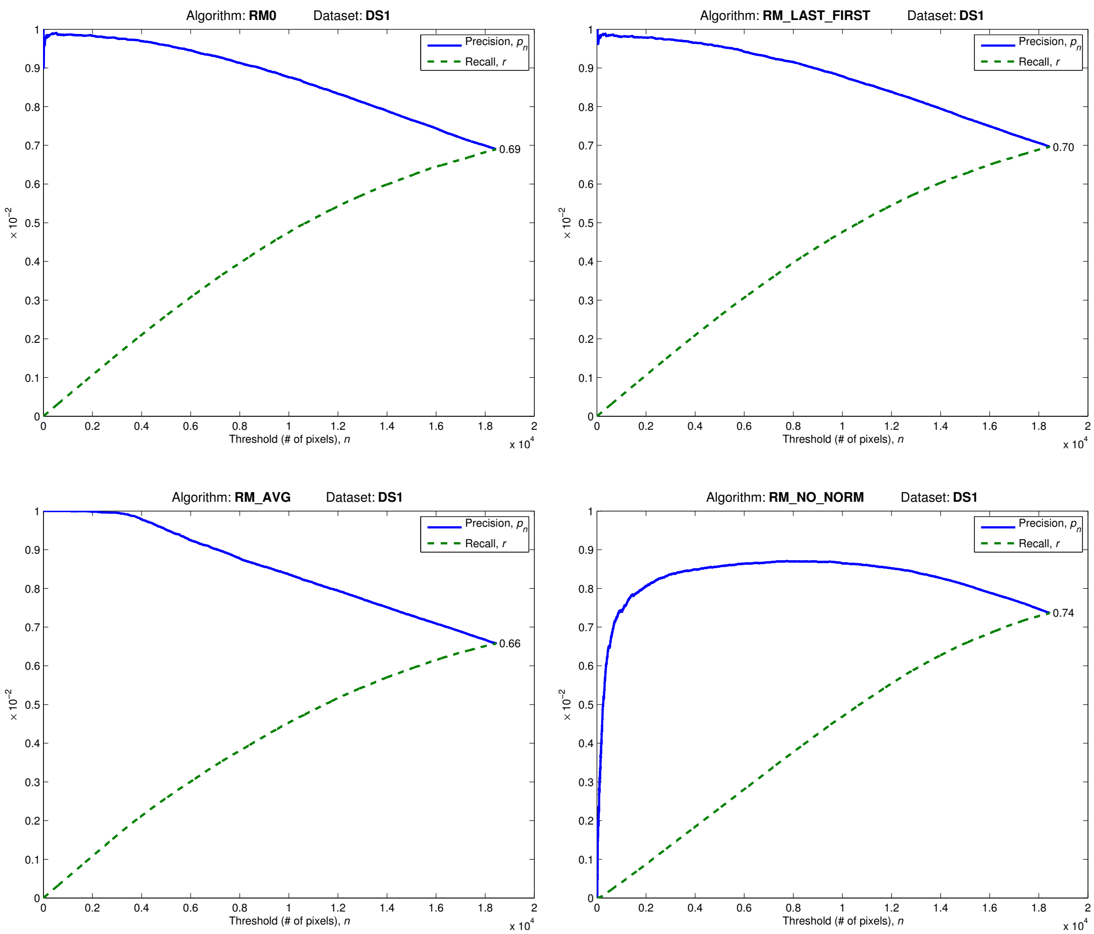


Figure 7.2: Recursive Merging and variations on the `DS1` data set.

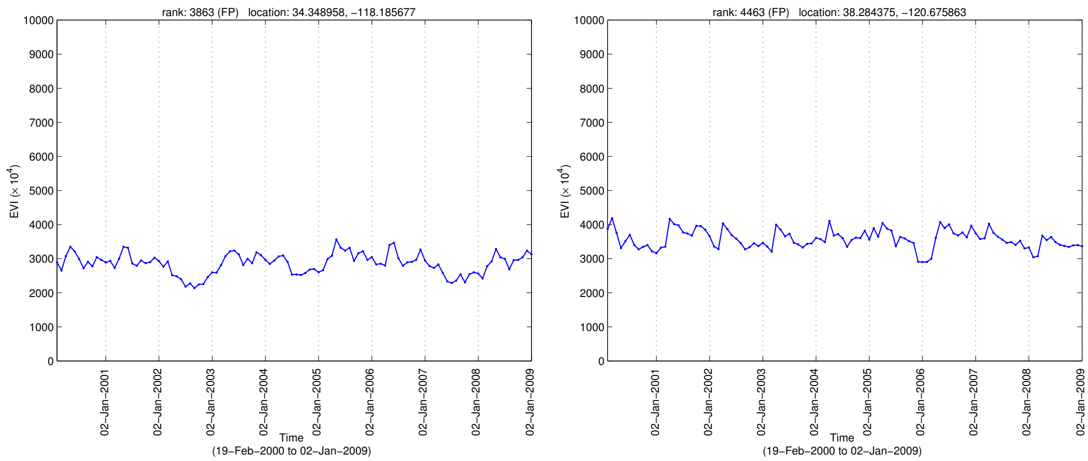


Figure 7.3: Sample of false positives detected by `RM0` in `DS1`.

differences under different noise levels; we do not evaluate on `DS2` in this experiment since we are not examining performance in the presence of missing values.

YD_ALL_PREVIOUS This scheme builds a model that includes all the available time steps. This is different from the standard yearly delta algorithm, which includes only a recent history of the time series in the model.

YD_ALL_PREVIOUS_WEIGHTED This scheme is a variation of the YD_ALL_PREVIOUS algorithm, which gives more importance to recent values when incorporating them in the model.

YD_MANHATTAN In this variation of the yearly delta algorithm, the distance between models is computed using the *Manhattan* distance.

MONTHLY_DELTA This is a simplistic algorithm, which is equivalent to computing the first order difference of the time series [25].

$$\textit{change score} = \max\{t_2 - t_1, t_3 - t_2, \ldots, t_T - t_{T-1}\}$$

SIXMONTH_DELTA This algorithm uses a smaller model that does not take all the available information into account. Thus, it is a scheme that is in between MONTHLY_DELTA and YD0 in terms of model sophistication.

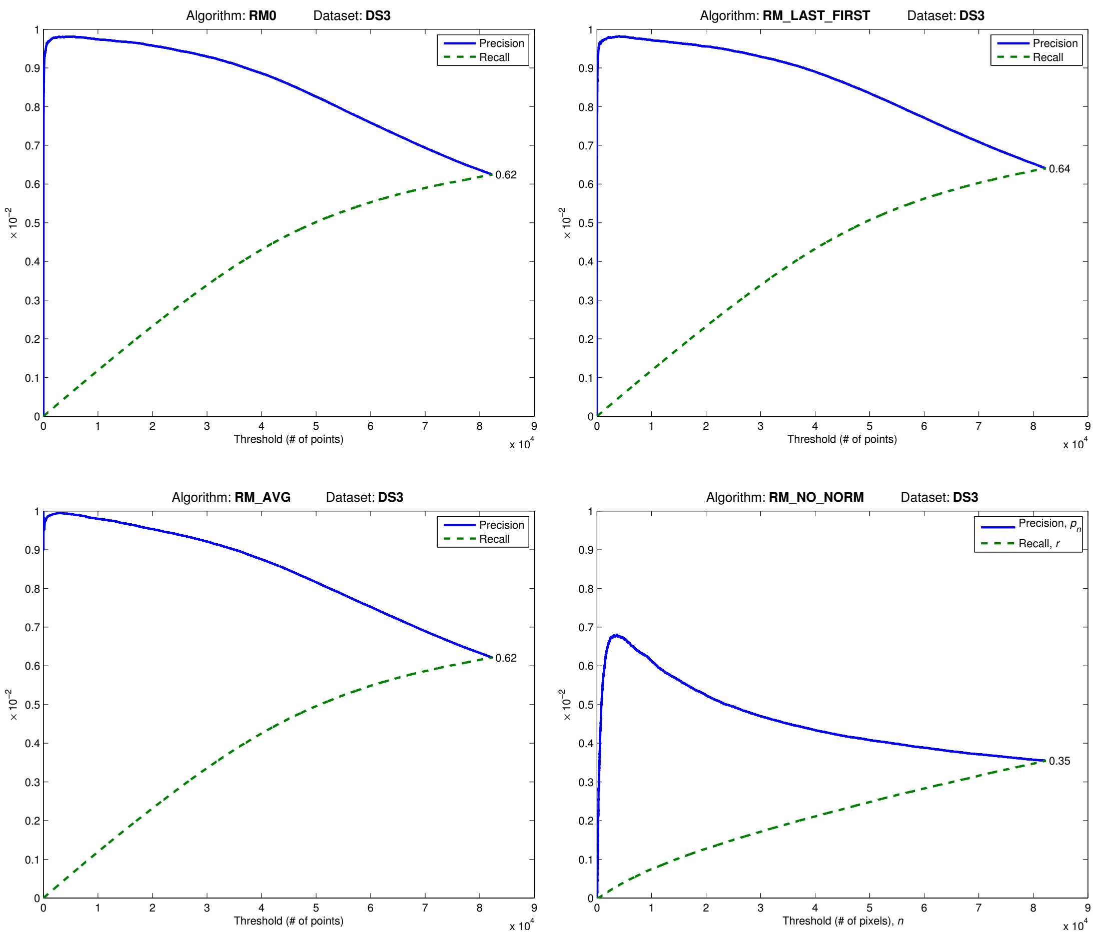


Figure 7.4: Recursive Merging and variations on the DS3 data set.

Figure 7.5 and Figure 7.6 show the results (in the form of precision and recall plots) of all the different variations of the yearly delta algorithm on the DS1 and DS3 data sets, respectively. In the remainder of the section, we will discuss the performance of each algorithm, including illustrative examples to explain specific performance differences.

MONTHLY_DELTA follows the straightforward approach of detecting changes using the difference from one time step to the next. In principle, this scheme should be able to detect many fires since most fires tend to occur in a short time span (implying most of the vegetation is damaged within one time step). However, as can be observed in Figures 7.5 and 7.6, the scheme does not perform well on DS1 and breaks down on the noisy DS3. Figure 7.7 shows examples of *false negatives* missed by MONTHLY_DELTA on DS1 (left) and DS3 (right), but detected by YD_ALL_PREVIOUS. These plots illustrate the drawbacks of the MONTHLY_DELTA algorithm. The plot on the left shows a relatively smooth time series for a pixel which experienced a fire in mid-2006. MONTHLY_DELTA missed this pixel because the drop to the minimum did not occur in a single time step but rather happened over several time steps. The plot on the right shows a moderately noisy time series; the noise is mainly skewing observations upward to higher values. The artificially high value in early 2001 causes the MONTHLY_DELTA algorithm to attach more importance to that portion of the time series rather than the genuine change that occurs in 2004. Pixels that have even more noisy spikes (but no genuine change) than this time series are given a higher rank, and this pixel did not make it into the top-n ranks. This is primarily why the MONTHLY_DELTA algorithm breaks down on DS3. Thus, we see that the monthly delta algorithm is brittle even for low-noise data, but completely breaks down for highly noisy data.

Figure 7.5 shows that YD_MANHATTAN slightly underperforms YD_ALL_PREVIOUS and YD0 overall on DS1. However, the figure also shows that the YD_MANHATTAN algorithm makes many mistakes in the top ranked portions. We examined a sample of these false positives and found that these were primarily of two types: pixels with high variability and pixels with subtle changes. Figure 7.8 shows a sample of both of these types of pixels. The time series on the left has very high variability and some noise, but does not correspond to a typical forest fire pattern. The time series on the right shows a pixel with a partial drop in vegetation in some years, but is clearly not a fire pattern. These types of time series dominate the false positives in the top ranked

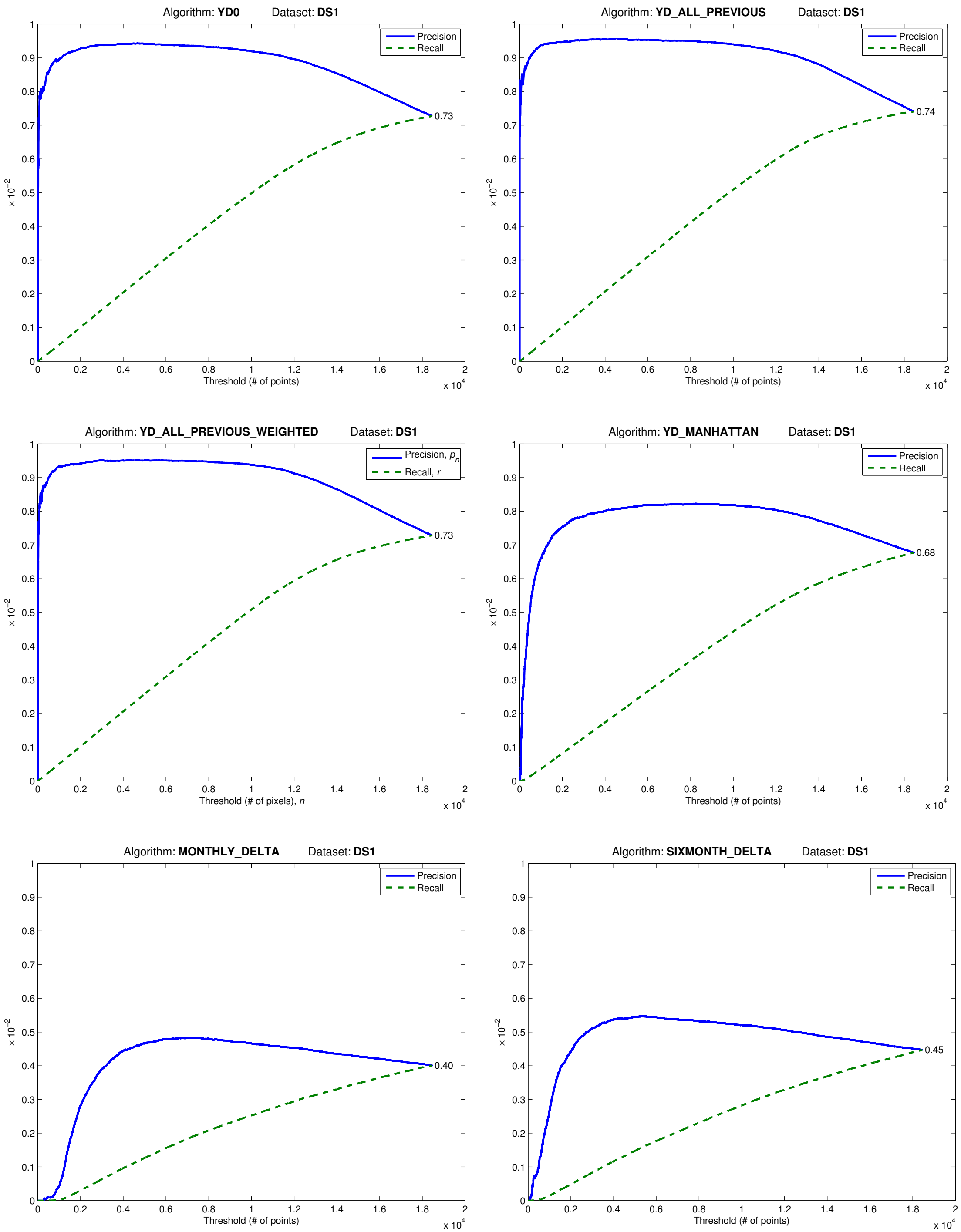


Figure 7.5: Yearly Delta and variations on the DS1 data set.

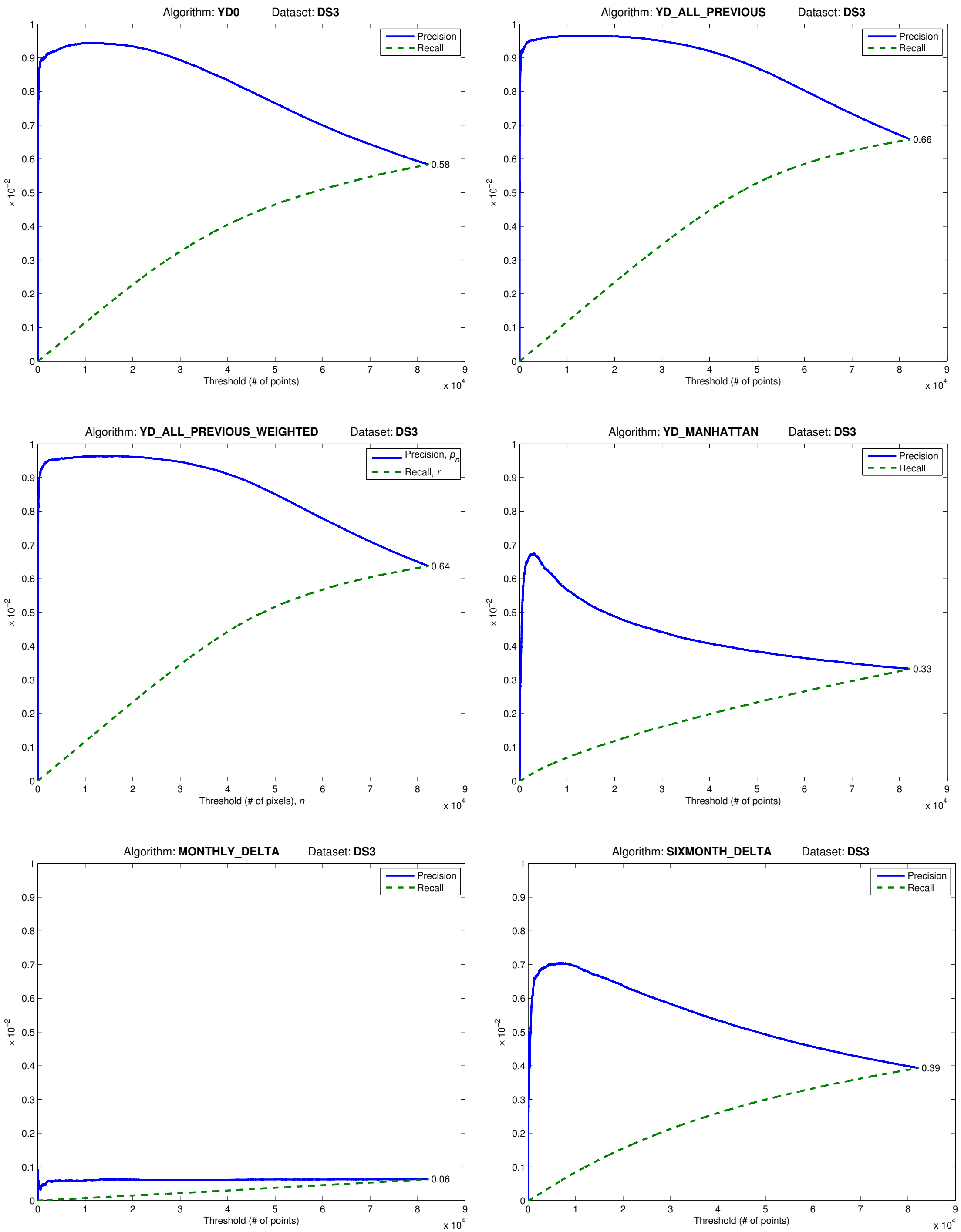


Figure 7.6: Yearly Delta and variations on the DS3 data set.

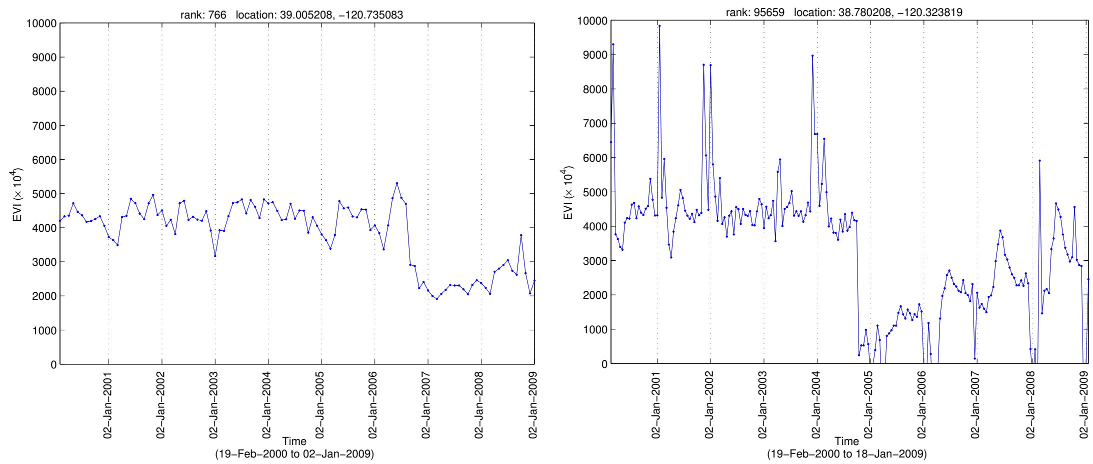


Figure 7.7: Sample of false negatives missed by MONTHLY_DELTA in DS1 (left) and DS3 (right).

pixels returned by the YD_MANHATTAN algorithm. Both these examples illustrate the limitations of using the L_1 distance to compute the difference between models in the yearly delta scheme; i.e. it is sensitive to noise and subtle shifts. The performance of YD_MANHATTAN falls further for DS3, where there is a high level of noise, as can be observed in Figure 7.6.

The SIXMONTH_DELTA algorithm uses a smaller model than the basic YD0 scheme. Interestingly, in contrast to most of the other algorithms discussed in this section, the SIXMONTH_DELTA algorithm performs *better* with the noisy DS3 than with the low noise DS1. Upon examining several of the top ranked pixels from both DS1 and DS3, we discovered that the SIXMONTH_DELTA algorithm is sensitive to asymmetries in the seasonal cycle of the time series but is less sensitive when the time steps are shorter. Figure 7.9 shows an example of a false positive from DS1 (left) and a true positive from DS3 (right). In DS1, a large number of pixels were found to be similar to Figure 7.9(a), with varying degrees of asymmetry in the seasonal cycle. On the other hand, the algorithm was able to detect many pixels such as the time series shown in Figure 7.9(b), which had not only asymmetries but moderate levels of noise. Thus, we observed that the SIXMONTH_DELTA algorithm is more sensitive to asymmetry for shorter time series and robust to moderate levels of noise.

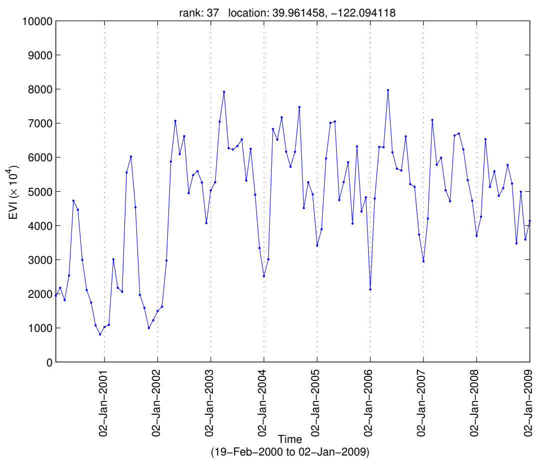
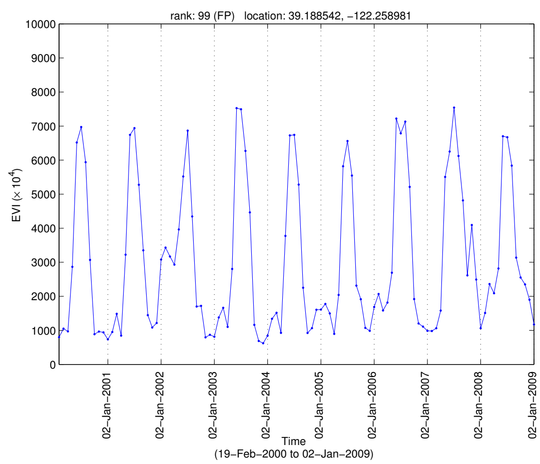
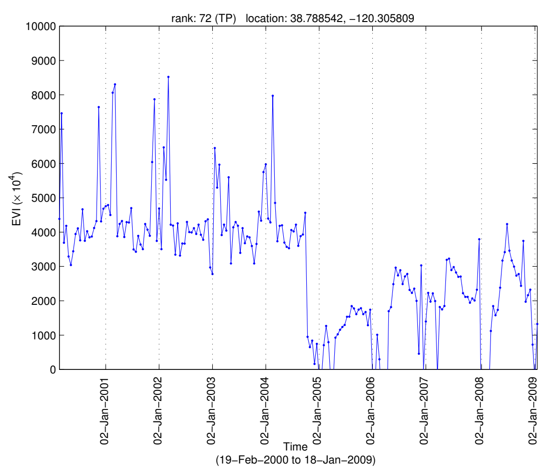


Figure 7.8: Sample of false positives detected by YD_MANHATTAN on DS1.

Figure 7.5 shows that YD_ALL_PREVIOUS and YD0 overall exhibit almost exactly the same behavior on DS1. However, looking at Figure 7.6, where the input data has a high level of noise, it is apparent that the behavior of the two algorithms is very different: YD_ALL_PREVIOUS performs much better overall, and particularly better for the top ranked points. We examined several of the false positives detected by YD0 to understand this behavior further; we also examined false *negatives* that were missed by YD0 but were detected by YD_ALL_PREVIOUS. Figure 7.10 shows a sample false positive and a sample false negative. The plot on the left shows a time series with some noise (mostly noisy downward) and the plot on the right shows a time series with a very high level of noise (mostly noisy upward). In both situations, noise appears to have thrown off the algorithm by artificially raising the score. In the case of the false positive (left), the noisy value in early 2007 followed by a subsequent drop in the time series led to a high score for this pixel, while in the case of the false negative a real fire was given a low score because a noisy value prior to the event led to a low score for the time series. This suggests that YD0 is sensitive to high levels of noise. The YD0 algorithm incorporates a recent history of the time series into the model, while YD_ALL_PREVIOUS incorporates the entire history of the time series into the model. Thus, the robust behavior of YD_ALL_PREVIOUS makes it a better candidate for noisy data sets such as DS3.

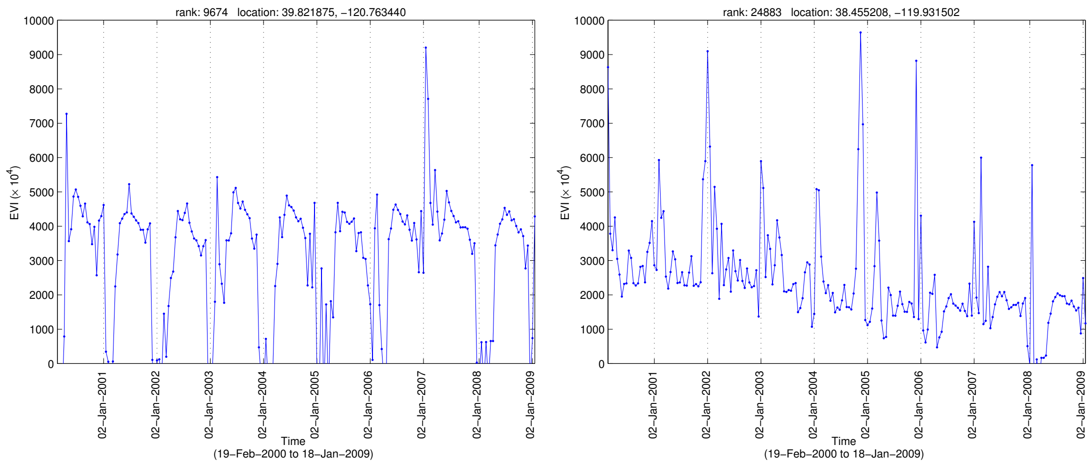


(a) False positive.



(b) True positive.

Figure 7.9: Sample of a false positives detected by SIXMONTH_DELTA on DS1, and a true positive detected on DS3.



(a) False positive. (b) False negative.

Figure 7.10: Sample of a false positive and a false negative detected by YD0 on DS3.

7.4 Comparison of Change Detection Algorithms

In this section, we comparatively evaluate the performance of a number of different change detection algorithms on the DS1 data set. The goal is to examine the relative differences between the various algorithms when the data is of the highest quality, i.e. noise and missing observations are not a factor in performance differences.

RM0 Basic recursive merging algorithm.

YD0 Basic yearly delta algorithm - difference between mean of before and after.

LUNETTA_NO_NORM Variation of basic algorithm from [87] which does not perform normalization.

CUSUM_MEAN For a given time series, the first year's mean is subtracted from each month, and the cumulative sum is computed. The final score is the highest value of the cumulative sum.

CUSUM_MEAN_MISSING Extension of CUSUM_MEAN that handles missing values. When a missing value is encountered, the algorithm sets the value to 0 and proceeds.

Year	Polygon Size	RM0	YD0	LUNETTA_NO_NORM	CUSUM_MEAN
2000	111	54	10	24	12
2001	1142	814	943	796	1009
2002	2407	1383	2238	2197	2164
2003	4946	3609	4258	3512	4338
2004	661	423	571	412	521
2005	192	96	136	100	134
2006	278	146	187	157	152
2007	1935	1413	1669	1332	1360
2008	6778	5312	4078	4556	490
SUM	18450	13250	14090	13086	10180
p_n	1.00	0.72	0.76	0.71	0.55

Table 7.4: Results of algorithms on DS1.

Figure 7.11 shows the results (in the form of precision and recall plots) of all the change detection algorithms on the DS1 data set. The results are also shown in Table 7.4, which shows aggregate counts by year. In the remainder of the section, we will discuss the performance of each algorithm, including illustrative examples, to explain specific performance differences.

Figure 7.11 shows that the CUSUM_MEAN algorithm has a large fraction of false positives in the top ranked pixels. Figure 7.12 shows a sample of top ranked pixels from the CUSUM_MEAN algorithm. The example on the left consists of a time series where (in 2005) there are *multiple* periodic cycles in one year. This leads to the CUSUM score being artificially high for the time series even though no real change has occurred. The example on the right show a false positive where the *length* of the seasonal cycle is different for different years.

A key observation from Table 7.4 (and Tables 7.5 and 7.6 in the subsequent section) is that the recursive merging approach always outperforms the yearly delta approach in the endpoints, i.e., the 2000 and 2008 years. This can be attributed to the fact that YD0 needs a window at the beginning to train the initial model, while RM0 utilizes all the values in building segments.

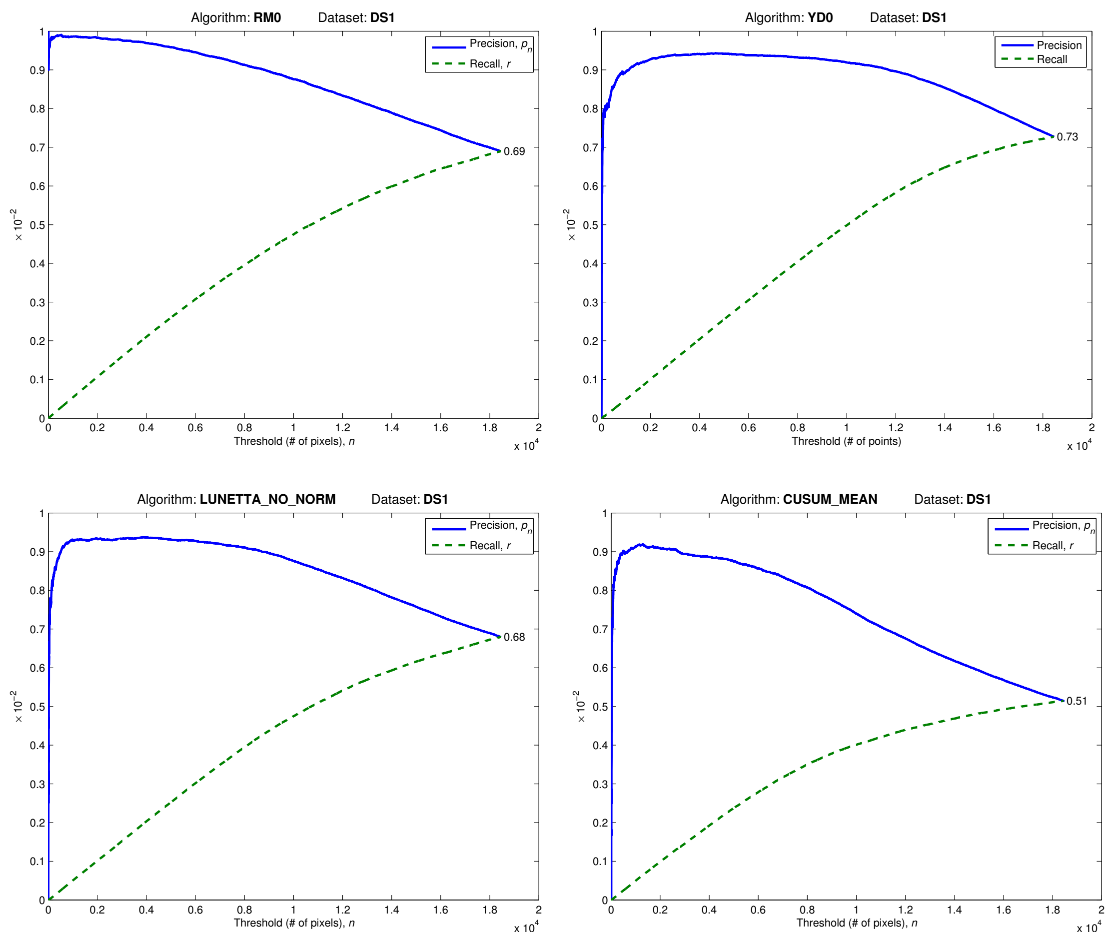


Figure 7.11: Comparison of algorithms on `DS1`.

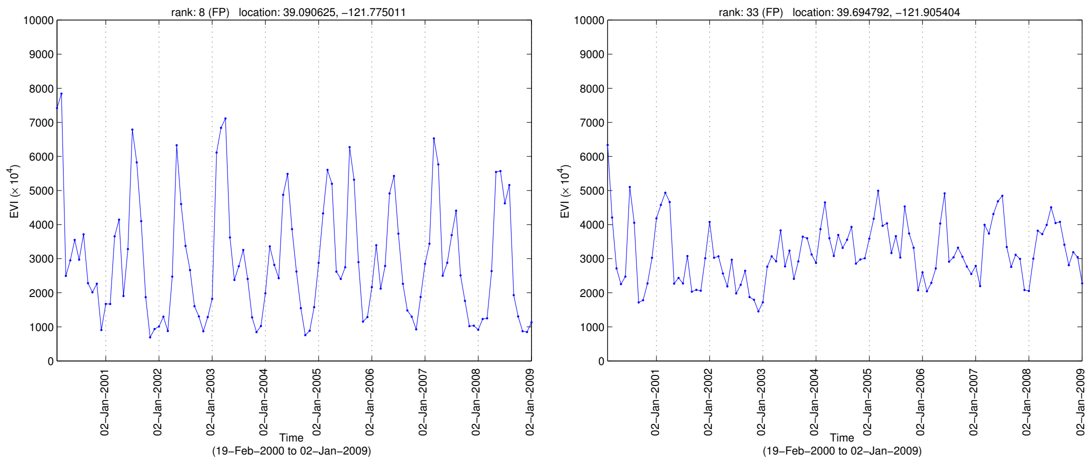


Figure 7.12: Samples of false positives detected by CUSUM_MEAN on DS1.

7.5 Evaluation of Robustness

In this section, we examine the relative performance of various algorithms under different levels of missing and noisy observations. Specifically, we compare the *robustness* in the presence of missing values and noise by examining performance on the DS2 and DS3 data sets; the RM0_MISSING, YD0_MISSING and CUSUM_MEAN_MISSING algorithms are compared on DS2, while RM0, YD0, LUNETTA_NO_NORM and CUSUM_MEAN are compared on DS3. We only evaluated algorithms which had natural extensions to deal with missing data, and hence did not evaluate the LUNETTA algorithm on the DS2 data set.

The results in Table 7.5 and Figure 7.13 show that the RM0_MISSING technique performs nearly as well on DS2 as RM0 does on DS1. On the other hand, we can observe that missing observations lead to a significant drop in performance for the CUSUM_MEAN_MISSING algorithm. Figure 7.14 shows two sample pixels that illustrate some of the reasons for CUSUM_MEAN_MISSING's poor performance. The figure shows a false positive that was detected by CUSUM_MEAN_MISSING but *not* by RM0_MISSING, and a false negative that was not detected by CUSUM_MEAN_MISSING but *was* detected by RM0_MISSING. In the case of the false positive, we see that even a few missing values (shown as gaps in the figure) are enough to throw off the CUSUM score to an artificially high level. The false negative example shows the robustness of RM0_MISSING; there are only a few values in each year of the time series but the algorithm is able to detect

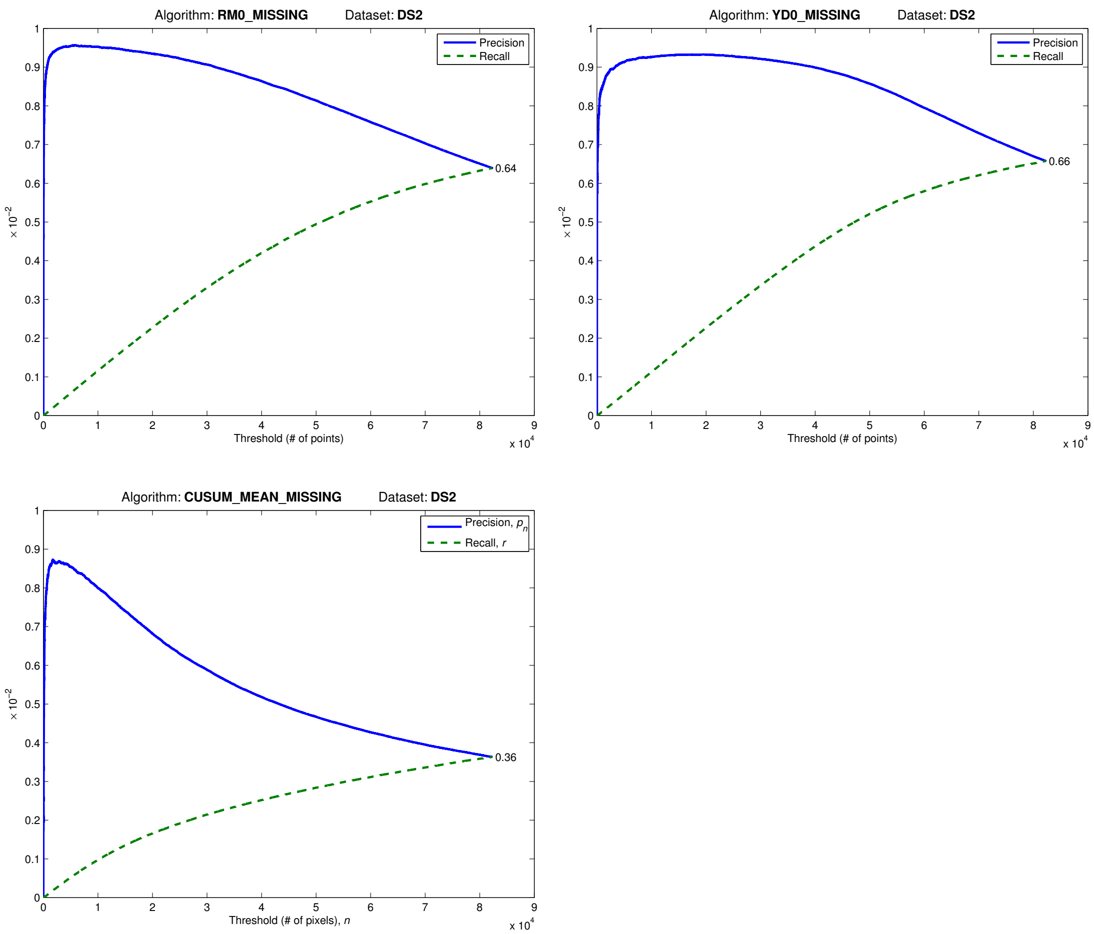


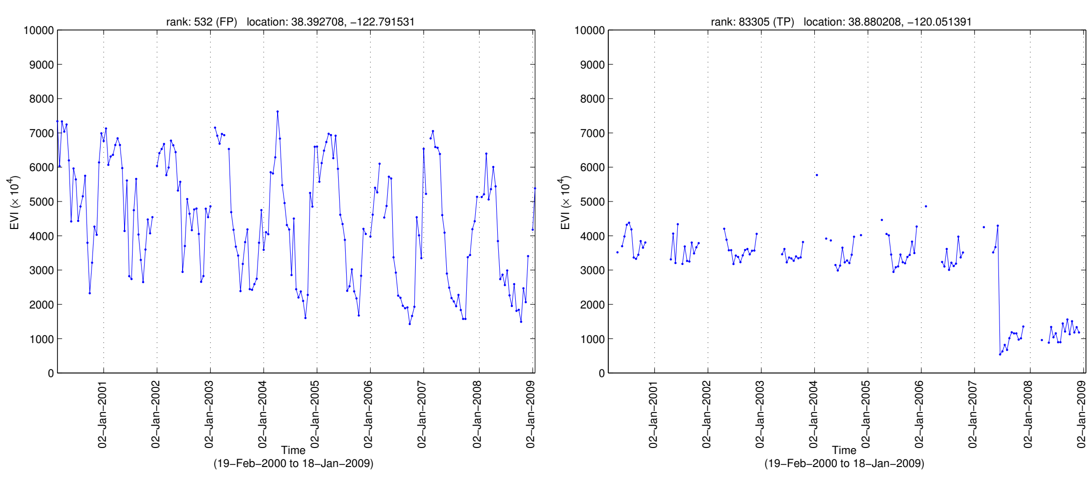
Figure 7.13: Comparison of algorithms with missing data (DS2).

the change point correctly. CUSUM_MEAN_MISSING assigned a very low change score to this time series thereby leaving out this pixel from the top ranked group. Thus, the RM0_MISSING algorithm is very robust in the presence of missing values compared to other techniques.

The DS3 data set is the most challenging in terms of data quality since it has no data or quality filtering whatsoever. The results are shown in Table 7.6 and Figure 7.15. It can be observed that even in these conditions, the RM0 and YD0 algorithms continue to perform well, while the CUSUM scheme shows a large drop in performance compared to DS1. Figure 7.16 shows two sample pixels that illustrate some of the reasons for CUSUM_MEAN's poor performance. The figure shows a false positive that

Year	Polygon Size	RM0_MISSING	YD0_MISSING	CUSUM0_MISSING
2000	1379	630	476	234
2001	6827	4154	5189	4593
2002	12092	7541	9047	7977
2003	12292	8423	9763	8207
2004	4218	3078	3600	3011
2005	744	377	446	280
2006	6165	4076	4689	2072
2007	10666	8923	9683	3316
2008	27901	17622	13961	2758
SUM	82284	54824	56854	32448
p_n	1.00	0.67	0.69	0.39

Table 7.5: Results of algorithms on DS2.



(a) False positive. (b) False negative.

Figure 7.14: Sample of a false positive and a false negative detected by CUSUM_MEAN_MISSING on DS2.

Year	Polygon Size	RM0	YD0	LUNETTA_NO_NORM	CUSUM_MEAN
2000	1379	458	294	431	443
2001	6827	3661	4757	3648	5520
2002	12114	7061	8284	7310	9335
2003	12292	8514	9058	6852	9915
2004	4218	2786	3452	3056	3152
2005	744	293	386	259	336
2006	6165	3900	4235	3296	3948
2007	10671	9285	9177	8459	6736
2008	27901	17581	11013	14890	1742
SUM	82311	53539	50656	48201	41127
p_n	1.00	0.65	0.62	0.59	0.50

Table 7.6: Results of algorithms on DS3.

was detected by CUSUM_MEAN but *not* by RM0, and a false negative that was not detected by CUSUM_MEAN but *was* detected by RM0. In the case of the false positive, it can be observed that when the noise level is very high, CUSUM_MEAN does not perform well, as the CUSUM score is being raised to an artificially high level. This plot also shows the robustness of RM0 even in the presence of many noisy (high spikes) values. The false negative example shows that even a single spike and a small amount of noise are enough to throw off the CUSUM score. CUSUM_MEAN_MISSING assigned a very low change score to this time series and thereby leaving out this pixel from the top ranked group.

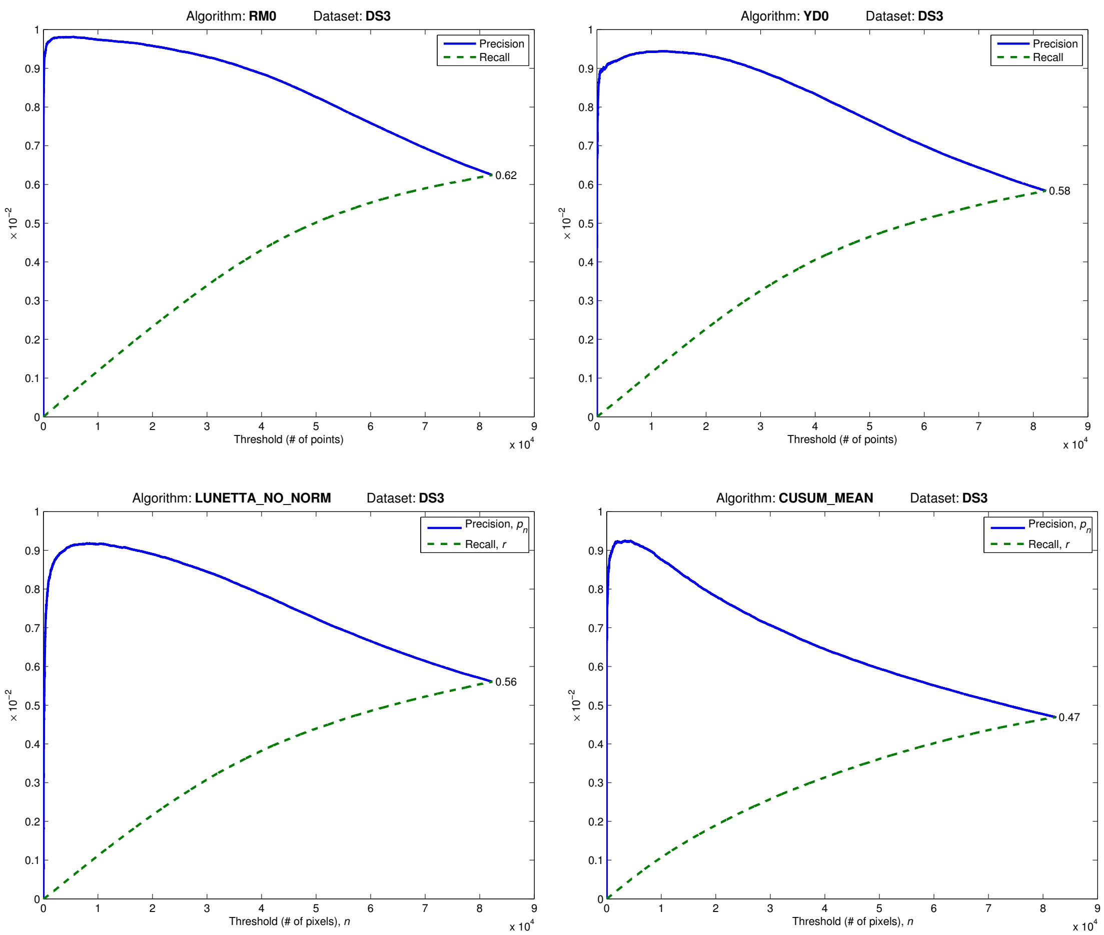
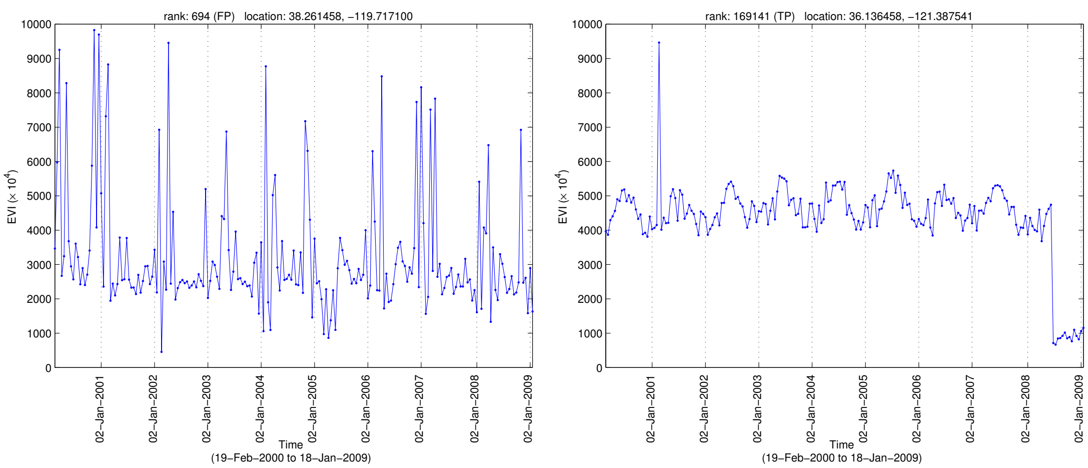


Figure 7.15: Comparison of algorithms with noisy data (`DS3`).



(a) False positive. (b) False negative.

Figure 7.16: Sample of a false positive and a false negative detected by CUSUM_MEAN_MISSING on DS3.

Chapter 8

Spatio-Temporal Event Identification

Land cover changes often span several pixels in a region. Change detection algorithms (most image-based and nearly all time series-based approaches) typically detect disturbance pixels independent of one another, either because of algorithmic design or data constraints. This raises two issues: (i) How to use the spatial context of pixels to improve change detection? (ii) How to group related changes into a single event? The main focus of this chapter is on the second issue. In particular, when change detection is performed using a time series approach, there is a need to group together pixels that correspond to the same change "event," which refers to the *physical* process (e.g. fire, logging, etc.) that led to the land cover being changed. For example, the time series for several pixels corresponding to a well-known fire are shown in Figure 8.1. Each time series was found independently by a change detection algorithm, but ideally one should be able to see a *unified* view of all the time series as is being shown in the figure. We refer to the process of summarizing sets of change pixels on the basis of proximity[1] as the *spatio-temporal event identification* problem. In this chapter, we formulate this problem, discuss related work, as well as some preliminary approaches. We also present a qualitative evaluation of methods for spatio-temporal event identification.

There are numerous advantages to applying a spatio-temporal event identification

[1]The notion of *proximity* is broad, and includes spatial proximity, time series similarity, event similarity, etc.

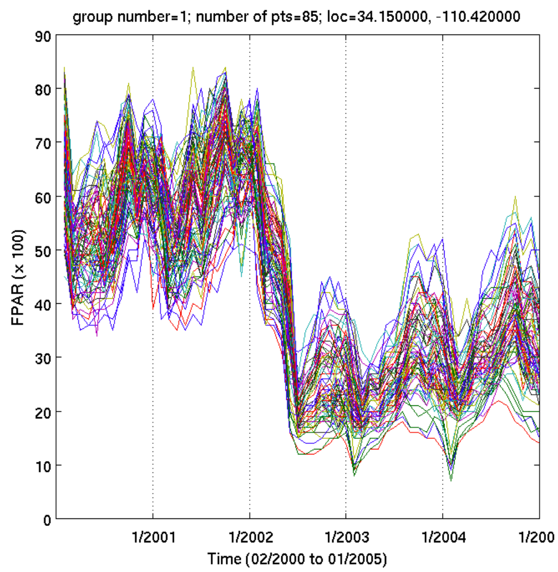


Figure 8.1: This collection of time series shows a dramatic drop in vegetation index (FPAR) around the summer of 2002. These time series correspond to a forested area near Phoenix, and the event corresponds to the well-documented Rodeo Fire [1].

algorithm to a set of change pixels. The most obvious advantage is that the summarized view enables exploration of disturbances (especially those of large scale) in a spatial fashion, for example, by displaying the grouped pixels on an interactive map as shown in Figure 8.2. This is also related to the problem of *characterizing* a detected change (also known as deducing the "fate" of the land cover), an important problem in its own right that is critical for evaluating the impact [41] of land cover change. Event identification can be seen as a step in enabling characterization, in that it gives the analyst a richer view of the changed pixels from which characterization may be easier.

Spatial summarization is also useful for detecting marginally changed pixels that may not otherwise be detected. For example, depending on the terrain, pixels in the middle or on the boundary of a fire may be only marginally disturbed (for example, see Figure 8.3). In other cases, such as logging, the activity may be slowly spreading across a region. Therefore, marginally changed time series may indicate ongoing, small-scale changes that are not large enough to exceed thresholds.

Another use of spatio-temporal identification is when there is incomplete remote sensing data. In this dissertation, we discussed approaches to the problem of missing

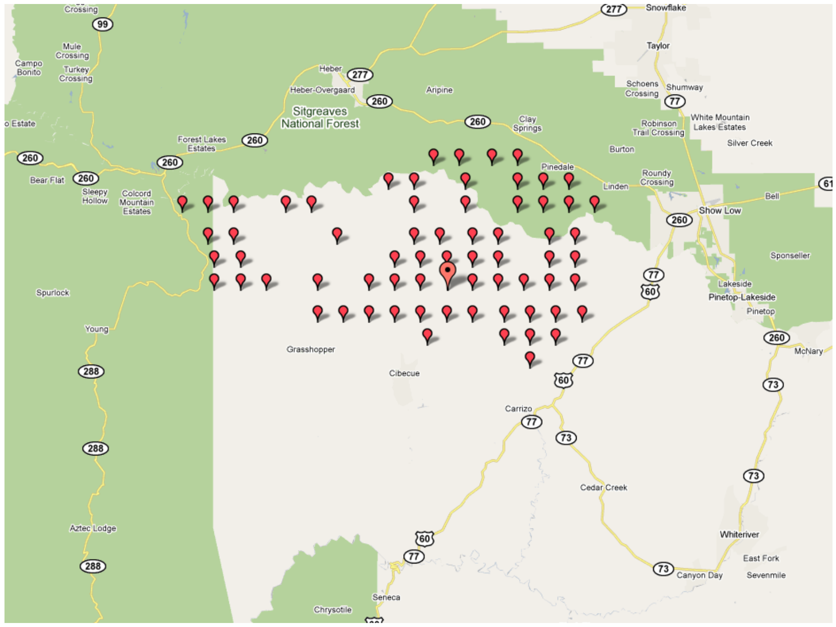


Figure 8.2: This screenshot of Google Maps is an interactive view of the pixels involved in the Rodeo Fire, for which the time series are shown in Figure 8.1.

and noisy data extensively in Chapter 5. In some cases, where the problem of missing and noisy data is especially acute, the techniques from Chapter 5 may not be applicable. However, if there are a few pixels for which changes can be reliably detected, the remaining disturbed pixels in the region can potentially be detected by examining the spatial neighborhood, potentially overcoming the severe missing and noisy data problem. For example, while most change detection algorithms assign a high score to pixels that are easily identified as change, other points that have also changed may received a lower rank (or no rank due to missing data); in these instances an event identification algorithm can group together pixels from the same event thereby improving the results of change detection and providing a more complete view of the underlying event.

8.1 Problem Formulation

Given a time series data set D with N pixels, the output of a change detection algorithm is a list of change scores of length N, where each change score (that has a one-to-one mapping to a pixel) is a measure of the degree of the change for the corresponding

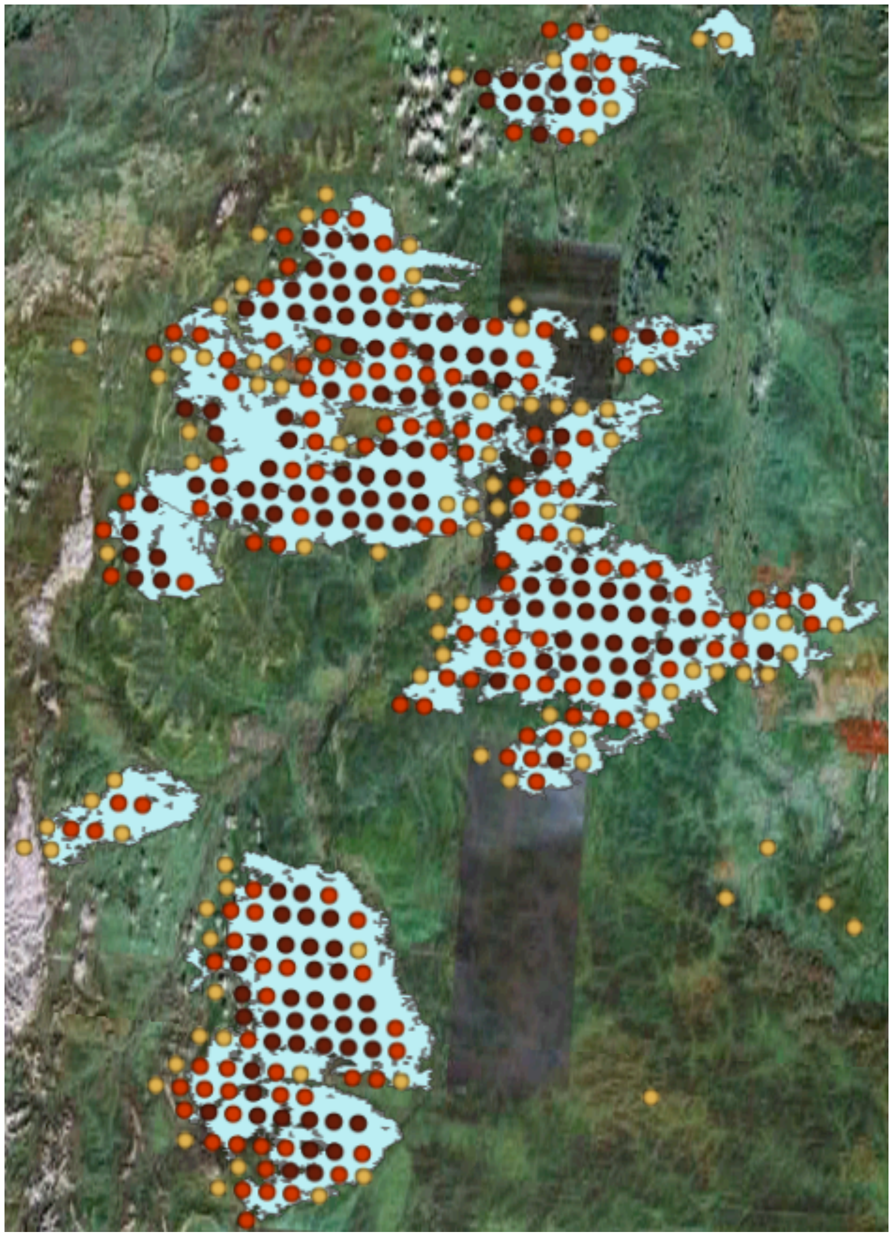


Figure 8.3: This screenshot from Google Earth shows an overlay of ground truth (light blue polygons) and pixels (dots) detected by the YD0 algorithm in province of Yukon Territory in Canada. The shading of each dot indicates the extent to which the pixel was damaged, with red corresponding to the most damage and yellow the least damage. We observe that for all of the fires, there are pixels on the periphery which are marginally damaged.

pixel. The *spatio-temporal event identification* problem is to produce a summarization that groups together pixels (from the top k portion of pixels) corresponding to the same disturbance event.

8.2 Related Work

Spatial clustering is the problem of grouping similar spatial objects into classes. The applications of spatial clustering include a wide variety of domains where geographic data is prevalent, ranging from geophysical data to commercial sales data. Han et al. [52] presented a survey on the spatial clustering problem. They categorize algorithms into four approaches: partitioning, hierarchical, density-based and grid-based. The survey primarily discusses the framework where geographical information is incorporated into the clustering feature space and does not address clustering the output of anomaly or change scoring algorithms.

Chandola and Kumar [21] discussed the problem of *summarization* where the broad goal is to find a compact representation of a data set. In particular, they formulate the problem and discuss its application in the area of network intrusion detection, where categorical network flow data (also called *netflows*) is represented in the form of transactions. Given a ranked list of anomalous network flows, the goal was to summarize the netflows so that a network analyst can analyze a much larger number of flows in a fixed time. Summarization algorithms group the ranked netflows into the most compact representations possible while trying to minimize the level of detail that is lost for data instances. The authors also defined two metrics to determine the quality of a specific summarization output based on the notions of compaction gain and information loss. Although the techniques in [21] were designed for transaction data, summarization is similar to the spatio-temporal event identification problem from a conceptual point of view; i.e. both problems consider a ranked list of objects, and objects with higher ranks server as *anchor* points for grouping other related objects. Thus, in principle these techniques can be extended to time series data.

8.3 Algorithms

8.3.1 Spatial Scan

The spatial scan approach (drawing on methodology from the spatial scan statistic [79]) is a grid-based method that explicitly utilizes spatial information to group together pixels. The generic spatial scan algorithm is as follows:

1. Select a starting position in the spatial grid.
2. Move (either horizontally or vertically) until a grid cell that contains a change pixel that has not already been touched is reached. Call this pixel $\hat{P}$.
3. Examine the spatial neighborhood (defined by a distance threshold) of $\hat{P}$. If there are any unassigned pixels that are within some geographical distance threshold, assign these pixels to $\hat{P}$ and mark them as touched.
4. Repeat step 2 until the entire grid has been examined.

The spatial scan approach offers a methodology to take advantage of spatial information to form meaningful groups. However, it does not give preference to high ranked points that can serve as anchor points. While the spatial neighborhood is considered, the approach ignores other factors such as the shape of the time series and the time of change that can be critical in grouping related events. Finally, this approach can be expensive and is sensitive to the starting point, the direction of exploration (horizontal/vertical) and the distance threshold.

8.3.2 k-means

k-means (and the bisecting k-means variation) is an efficient clustering algorithm that typically converges in a few iterations [119]. We previously used the k-means algorithm to perform a clustering of the EVI data for the San Francisco Bay Area region (see Figure 6.1). When the clusters are viewed on a map and compared to satellite imagery, they corresponded to the actual vegetation in the region very closely, showing that k-means can be effective in clustering EVI data into meaningful groups.

A generic k-means-based spatial event identification algorithm proceeds as follows:

1. The input data set is a set of pixels that have changed as the objects, and the time series as the feature space.

2. The k-means algorithm is run on the input data, with a specific value of k. A distance measure must also be selected for this step.

3. The final clustering from k-means is run through a post-processing process to flag especially noisy clusters, etc. This step is especially useful for bisecting k-means, where the final clustering may contain clusters that are very similar to each other but were split during an early bisection.

4. The resulting clustering after postprocessing is used to define the spatial events.

The optimal number of clusters (k) can be set by visually inspecting the SSE plot, or by using information criterion such as AIC. The k-means approach is fast and efficient, but has a number of limitations. While the scheme does incorporate the shape of the time series, it is difficult to include the time of change and spatial information into the algorithm. Specifically, a central aspect of k-means is the centroids which must be recomputed after every iteration. However, there is no straightforward mechanism to incorporate the time of change and space into the centroid construction process. Finally, the k-means approach also does not support the use of high ranking anchor points.

8.3.3 Spatio-temporal Leader Algorithm

We devised an algorithm that directly utilizes the ranking of change pixels as well as spatial information and the time of change into a framework that incorporates aspects of the Leader clustering algorithm [38]. The intuition behind this approach was that the most highly ranked pixels are likely to represent the "core" of the underlying change process and thus are a good place to start. The algorithm proceeds as follows:

1. The input data set is a list $\mathcal{L}$ of *ranked* pixels that have changed, i.e., the output from a change detection algorithm.

2. A list of centroids $\mathcal{C}$ is initialized to the empty set $\emptyset$.

3. The highest ranked unassigned time series is selected from $\mathcal{L}$. Call this time series $\hat{T}$.

4. The time series $\hat{T}$ is compared to the every centroid in $\mathcal{C}$ using a similarity function.
    - If the similarity between the $\hat{T}$ and a centroid is within a user-supplied threshold, the time series is assigned to the centroid. $\hat{T}$ can either be assigned to the *first* centroid that meets the threshold, or to the *best* matching centroid.
    - If the none of the similarities with the centroids meets the threshold, then $\hat{T}$ is added to $\mathcal{C}$.
5. Steps 3 and 4 are repeated until every time series in the list has been assigned.

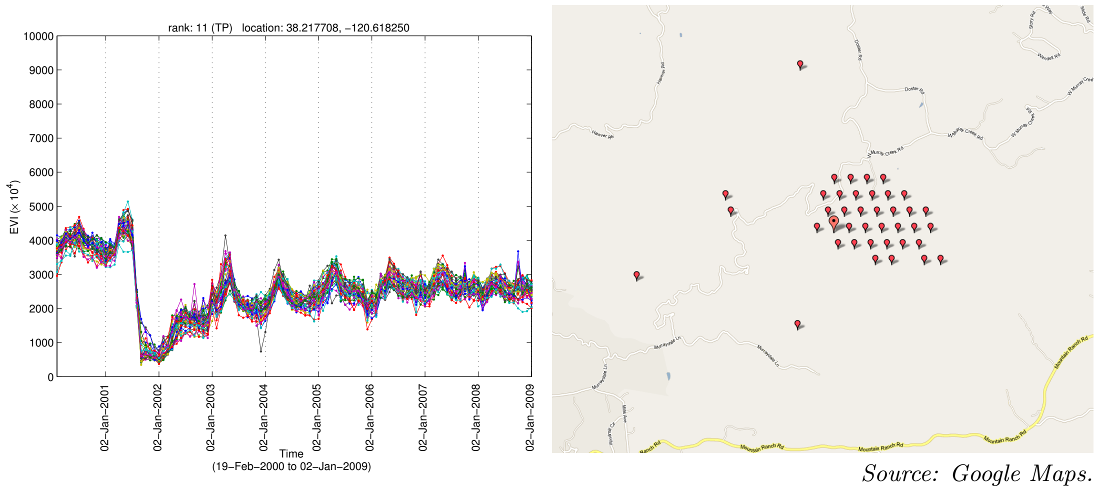
A variation of this algorithm enforces the constraint that pixels grouped into the same disturbance event must share the same time of change. In particular, the similarity measure can be defined such that two pixels are not considered similar unless the time series for both pixels changed within a certain time *window*.

Another variation can take distance into account by incorporating this information into the feature space and modifying the similarity function accordingly. Specifically, space can be incorporated either using a geographic distance threshold, or a function of the geographic distance itself. If a geographic distance threshold is used, then the similarity function is modified to a two-step process, where the first step is to check if the two pixels are within the distance threshold. A function of the geographic distance, on the other hand, can be directly factored into the similarity function.

One of the drawbacks of the spatial leader algorithm is that there are several parameters that must be tuned for the scheme to work effectively. In particular, it can be difficult to find an optimal value for the distance threshold. On the other hand, the spatio-temporal leader algorithm is able to incorporate spatial information, the shape of the time series as well the time of change, unlike the algorithms discussed in Sections 8.3.1 and 8.3.2. We provide illustrative examples using the spatio-temporal leader algorithm in the next section, but do not discuss the spatial scan and k-means approaches further.

8.4 Illustrative Examples

We ran the spatio-temporal leader algorithm on results from a change algorithm discussed in previous chapters of this dissertation. In particular, we utilized the ranked



Source: Google Maps.

Figure 8.4: Sample result from the spatial leader algorithm. Each time series on the left corresponds to an individual pixel. The map on the right shows the location of each pixel mapped using its geocoordinates. The single large pixel is the centroid/spatial leader.

pixels produced by the `YD0` algorithm on the `DS1` data set (see Chapter 4 for a description of `YD0` and Chapter 7 for more details on `DS1`).

The ranked pixels from `YD0` were taken as input to the spatial leader algorithm. Figure 8.4 shows a sample result from this algorithm. This result consists of 40 pixels that were assigned to one group. The figure illustrates a number of aspects of the algorithm. First, it can be observed that the time series in the group are highly homogeneous and clearly correspond to the same land cover event (in this case, a forest fire). As expected, the map shows that many of these locations are spatial grid neighbors. However, it can be observed that a few pixels are located at a considerable distance from the centroid. These pixels would not have been included in the group by the spatial scan approach using a reasonable distance threshold. This ability to form groups that are not necessarily spatial neighbors is one of the key strengths of the spatial leader algorithm.

Chapter 9

Conclusions and Future Directions

In this concluding chapter, we summarize the research results of this dissertation and discuss future directions of research which build upon the ideas and approaches from this dissertation. We also discuss other domains and applications where the techniques developed in this dissertation may be applied.

9.1 Summary

In this dissertation, we translated challenges in analyzing large-scale earth science data to computer science problems, and provided computational solutions to address some of these problems. In particular, three key technical capabilities were developed: (1) scalable time series change detection algorithms that allow global analysis; (2) ability to handle the large number of missing and noisy values present in satellite data sets that dramatically expands spatial coverage; (3) spatial analysis techniques to identify the scale and scope of disturbance events.

We presented a suite of scalable time series change detection algorithms for the land cover change detection problem. We also evaluated their performance on real-world data and obtained key insights into the strengths and weaknesses of the algorithms. The experimental evaluation performed in this dissertation was validated against ground truth for forest fires in the state of California (primarily because this ground truth is of high quality). All of the algorithms presented in this dissertation are *scalable* to massive data sets, which is one of the key requirements for global scale land cover change studies.

We found the recursive merging algorithm to perform consistently well across three data sets with different levels of noise and missing data, primarily due to its nonparametric nature and robustness to noise and variability. The yearly delta scheme is specifically meant for detecting *reduction* in vegetation; in fact it performs the best amongst all algorithms for detecting fires. CUSUM is also primarily geared towards detecting changes in one direction, while LUNETTA is able to detect changes in *both* directions. We note here that a segmentation-based approach such as RM0 is likely to outperform the other algorithms (such as YD0 and CUSUM0) on a broader class of problems where many more types of changes are involved (see Chapter 6 and [13, 14] for examples). One of the unique strengths of CUSUM is its ability to detect both *small* and *gradual* shifts. LUNETTA is the only algorithm to directly incorporate *space* into the change detection process, although it is also sensitive to the coherence of the spatial neighborhood (if the neighborhood contains multiple land cover types, the scheme can lose fidelity).

We discuss a few key challenges and their relationship (strength/limitation) to the four major change detection algorithms below:

Robustness We observed that the recursive merging algorithm was robust to the presence of noisy and missing values in the time series. In particular, we observed that the performance of the RM0 algorithm was much better in relation to other algorithms on noisy data. The YD0 algorithm is able to perform well with missing values, but is sensitive to noise in general. CUSUM and LUNETTA do not offer mechanisms for addressing noisy and missing values, although this limitation may be addressed in future extensions to the algorithms.

Variability Modeling One of the key strengths of the recursive merging algorithm is that it accounts for variability in the time series. There are many sources of variability including those due to climatic variations and regional differences. By accounting for variability, the RM0 algorithm normalizes time series that exhibit high levels of variability but do not have a real change. None of the other algorithms directly model variability.

Time of Change When a time series $\{t_1, t_2, \ldots\}$ is given a high change score by a change detection algorithm, we consider that the land cover has changed for the corresponding pixel. Therefore, there is a time step i that represents the point in time when the land cover conversion activity began. Some change detection algorithms such as CUSUM and YD are able to provide an estimate of the time of change, while RM0 and LUNETTA are limited in their ability to identify the time step of change. However, none of the techniques *accurately* detects the time of change, and all require post-processing steps to find the time of change with high confidence. A further complicating factor is that it is easier to detect the time of change for some change events such as forest fires, but other events such as logging are much more difficult. Extensions to the YD0 and CUSUM algorithms may provide robust estimates of the time of change, especially if space is incorporated into the modeling framework.

Offline Change Detection Except CUSUM, the other three algorithms are all *offline* change detection algorithms in the sense that they operate on entire time series in batch mode. However, there are often settings where an algorithm that can operate in online mode is needed. This aspect is discussed further as a direction for future work in Section 6.

Multiple Change Detection Multiple change detection refers to the problem where an individual pixel may have undergone *several* land cover changes. For longer time series data sets that cover a few decades, such changes will occur often. The LUNETTA algorithm is the only one capable of addressing the multiple change detection problem. In principle, CUSUM and YD0 may also be able to detect multiple changes if extended to handle this setting. On the other hand, it will likely be much more difficult to extend segmentation approaches (such as RM0) to the multiple change detection setting without greatly increasing the computational complexity.

9.2 Future Work

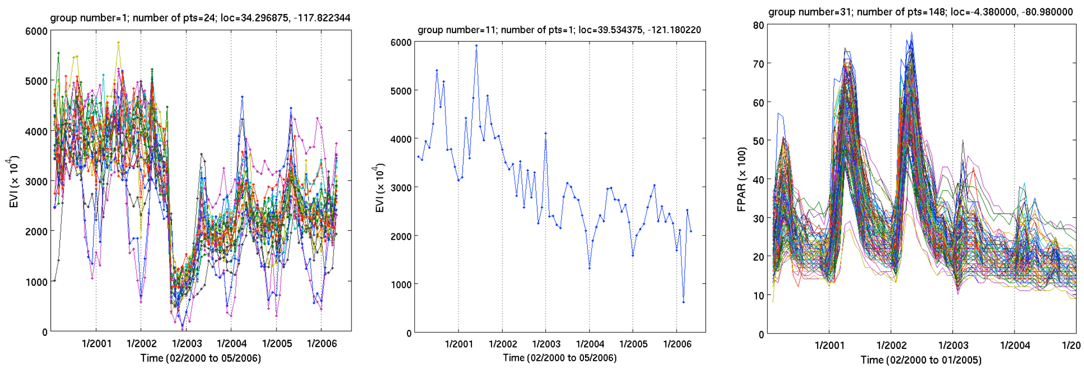
There are several open research problems in the broad area of spatio-temporal change detection for ecosystem and climate data. We have developed techniques in the context of land cover change studies, but the broader earth science domain as a whole raises a number of challenges for computational algorithms. We discuss a few open research problems in this section.

9.2.1 Characterization of Land Cover Changes

Land cover change detection algorithms serve the important role of detecting changes across the globe. Arguably, an equally important step after a land cover change has been detected is characterizing the detected change (i.e., deducing the "fate" of the land cover). This step is critical for evaluating its impact [41], and thus is a challenge that must be addressed in its own right. Changes of interest can happen both in space and time which makes change detection challenging. However, these two sources of information naturally complement each other, and can result in a powerful paradigm for characterizing signatures. For example, Figure 9.1 shows sets of time series corresponding to a forest fire, logging and a drought event, respectively. It can be seen that each of these events has a characteristic signature in the time series. For example, forest fires cause sudden and synchronized declines in vegetation index (e.g., FPAR) in several neighboring locations, whereas logging tends to result in a gradual decline in an isolated fashion. Drought and insect damage also cause gradual decline in vegetation but happen in a synchronized fashion over relatively large regions. There is a need for the design change detection algorithms that make use of such spatio-temporal characteristics and thus allow one to identify specific classes of changes.

9.2.2 Multi-resolution Analysis

Climate and ecosystem data often represent phenomena at multiple scales, both for single variables, and multiple variables. For example, there is a scale disparity between atmospheric, hydrological and land process models, on the one hand, and water, energy, environmental or infrastructural impacts assessments, on the other. Even for a single variable, data sets are frequently available at multiple resolutions. In this case, the hierarchical structure of the multi-resolution data can be exploited for computational efficiency. In particular, when the goal is to find large-scale changes, one can begin by



(a) Fire (b) Logging (c) Drought

Figure 9.1: Time series corresponding to a forest fire, logging and a drought event.

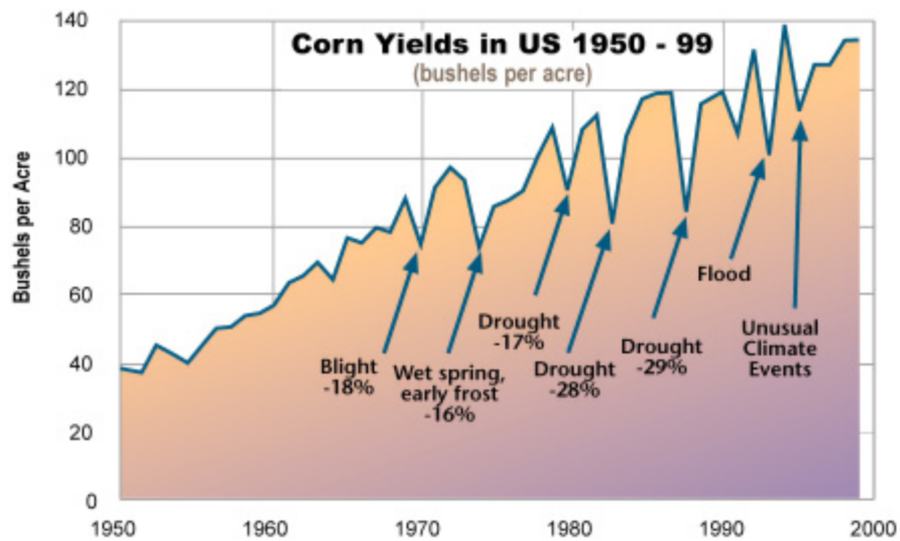
processing the coarsest resolution (which has the fewest pixels), and examine finer resolutions only when shifts are detected at the coarser resolution. This analysis framework raises several research problems: How will subtle changes be detected? Can an adaptive sampling approach be used?

9.2.3 Multivariate Analysis

Hyperspectral data from remote sensors consist of tens to hundreds of variables. To study some types of land cover changes (such as crop rotations), such multivariate data sets will be required because individual vegetation indices by themselves are not discriminative enough to detect such changes. For example, other satellite products, such as from IKONOS, can provide additional information to complement the information in vegetation indices [10].

Agricultural systems are vulnerable to climate extremes, with effects varying from region to region due to differences in soils, production systems, pollution levels, average temperature and other factors (e.g. see Figure 9.2). The impact of climate change on crop yield is an important question that requires multivariate data analysis techniques in order to model relationships between land cover change events and climate anomalies.

On a broader note, recent reports have suggested that increased corn production in the United States has driven up the global price of soybeans, thereby leading to more deforestation in Brazil [110, 15]. To understand such a relationship further, one



Source: US Global Change Research Program

Figure 9.2: Reductions in corn yields often correspond to extreme climate events including droughts and floods.

must analyze multiple time series data sets including agricultural, economic and remote sensing data. Thus, techniques for multivariate change detection need to be developed to answer many important questions. Multivariate analysis raises numerous research issues: How to select the right set of variables? How is spatial misalignment addressed?

9.2.4 Data Quality Improvement

Time series data sets in earth science are often contaminated by noisy and missing values caused by atmospheric interference, geometric distortion, etc. This issue, and several well-known approaches for missing value estimation, noise reduction and smoothing were discussed in Chapter 5. In this dissertation, we have followed the approach of designing *robust* algorithms that are able to perform well even if many observations in the time series are missing or noisy. Specifically, we observed that the performance of the recursive merging algorithm was much better in relation to other algorithms on contaminated data. However, we expect that *improving* data quality (using a combination of missing value estimation, noise reduction and smoothing) would also yield significant performance gains. Recent work by Hird and McDermid [59] has shown that some techniques have the potential to reconstruct a much cleaner time series from a noisy time series of greenness observations. In particular, techniques that are cognizant of domain-specific characteristics (e.g. seasonality, asymmetric noise, etc.) and take advantage of similar observations in *both* space and time can potentially produce a time series that is less contaminated, if not very close to the true time series. However, note that in the context of land cover change detection, special care must be taken when designing data quality improvement algorithms; a sharp drop (e.g. due to a fire) in the time series will *appear* similar to noise when compared to neighboring values, but the algorithm must not treat the value as noise. Therefore, the key test for any data quality improvement algorithm in the context of land cover change detection is whether the overall performance of change detection is better using cleaned data than performance using raw data.

9.2.5 Incremental Update and Real-Time Detection

Ecosystem and climate data from satellites and terrestrial sensors is usually updated at regular intervals (e.g. monthly or daily). This presents two issues for research in change detection algorithms: (i) real-time change detection; (ii) incrementally updating an event database when new observations have arrived.

Real-Time Change Detection There is often a need for real-time results from a change detection algorithm. For example, quick detection is essential for fire containment. In these situations, change detection algorithms will need to operate in real-time as well. This problem setting is also called the streaming or online setting. Some approaches, such as CUSUM, are well-suited for online change detection. However, more research is needed into such algorithms in the context of land cover change.

Incremental Update Even when detection does not need to be real-time, algorithms that are traditionally run in batch-mode would be more efficient if the results can be incrementally updated when new data arrives. In particular, consider a database containing historical land cover change events detected across the globe using monthly vegetation index time series for several years. When the input time series is extended by one month with new data from remote sensors, it would require tremendous computational resources to reprocess the *entire* time series for every location across the globe and update the events database. A more efficient approach would be to design a mechanism for *incrementally* updating the events database without recomputing across all the previous observations. Research questions that arise in this setting: (1) More recent data may be of lower quality (until validated); how does the algorithm adapt? (2) Can algorithms be more robust at end points?

9.2.6 Metrics for Spatial Event Identification Algorithms

We presented a problem formulation for spatial event identification in Chapter 8 of this dissertation. In order to objectively evaluate the performance of a given spatial event identification algorithm, metrics that determine the quality of output must be developed. The metrics can be based on ground truth data, as well as factors such as spatial proximity/connectedness of pixels in an event, similarity of the feature space, etc. These metrics will be critical in quantitative and comparative studies of spatial event identification algorithms.

9.3 Application in Other Domains

There are a number of domains that share characteristics with the earth science domain. In particular, time series data sets are common in several areas, and many of those areas also have data with periodicity, missing/noisy data, and spatial information. The change detection algorithms, techniques for noisy/missing data, and algorithms for spatial event identification developed in this dissertation can be potentially extended to a number of these domains. We discuss a few domains below.

Economics Time series data sets in economics frequently contain periodicity at the annual scale due to the various economic activities that happen on a yearly basis. Therefore, the change detection algorithms can be extended to economic time series data. Missing value techniques, and spatial event identification algorithms may also be applicable.

Health Care ECG data from cardiac monitoring frequently exhibits a high degree of periodicity.

Network Traffic Data sets resulting from the monitoring of network traffic frequently contain cycles at the daily, weekly, and hourly levels. Change detection in this domain may correspond to events such as equipment upgrades, routing changes, or bandwidth usage/availability changes.

Bibliography

[1] Rodeo–Chediski Fire.
http://en.wikipedia.org/wiki/Rodeo-Chediski_Fire.

[2] NASA Earth Observing System.
http://eospso.gsfc.nasa.gov.

[3] Land Processes Distributed Active Archive Center.
http://edcdaac.usgs.gov.

[4] In the California desert, they use water like there's no tomorrow—but tomorrow is coming. *U.S. Water News Online*, June 2003.

[5] T. Aach and A. Kaup. Bayesian algorithms for adaptive change detection in image sequences using markov random fields. *Signal Processing: Image Communication*, 7(2):147 – 160, 1995.

[6] A. Allen. Environmental planning and management of the peri-urban interface: perspectives on an emerging field. *Environment and Urbanization*, 15(1):135–148, 2003.

[7] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, Reading, MA, 1999.

[8] S. Banerjee, B. Carlin, and A. Gelfand. *Hierarchical modeling and analysis for spatial data*. Chapman & Hall, 2004.

[9] S. Banerjee, A. E. Gelfand, A. O. Finley, and H. Sang. Gaussian predictive process models for large spatial data sets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(4):825–848, 2008.

[10] A. Bannari, A. Pacheco, K. Staenz, H. McNairn, and K. Omari. Estimating and mapping crop residues cover on agricultural lands using hyperspectral and ikonos data. *Remote Sensing of Environment*, 104(4):447–459, 2006.

[11] A. Bifet and R. Gavaldà. Kalman filters and adaptive windows for learning in data streams. In *DS 2006: Proceedings of the 9th International Conference on Discovery Science*, pages 29–40. Springer, 2006.

[12] E. Bingham, A. Gionis, N. Haiminen, H. Hiisilä, H. Mannila, and E. Terzi. Segmentation and dimensionality reduction. In *Proceedings of the Sixth SIAM International Conference on Data Mining*, 2006.

[13] S. Boriah, V. Kumar, M. Steinbach, C. Potter, and S. Klooster. Land cover change detection: A case study. In *KDD '08: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 857–865, 2008.

[14] S. Boriah, V. Kumar, M. Steinbach, P.-N. Tan, C. Potter, and S. Klooster. Detecting ecosystem disturbances and land cover change using data mining. In H. Kargupta, J. Han, P. Yu, R. Motwani, and V. Kumar, editors, *Next Generation of Data Mining*. CRC Press, 2009.

[15] B. Borrell. Biofuel showdown: Should domestic ethanol producers pay for deforestation abroad? *Scientific American*, 2009.

[16] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel. *Time Series Analysis: Forecasting and Control*. Wiley, 2008. ISBN 0816211043.

[17] L. Bruzzone and D. Prieto. Automatic analysis of the difference image for unsupervised change detection. *IEEE Transactions on Geoscience and Remote Sensing*, 38(3):1171–1182, 2000.

[18] B. P. Carlin, A. E. Gelfand, and A. F. M. Smith. Hierarchical bayesian analysis of changepoint problems. *Applied Statistics*, 41(2):389–405, 1992.

[19] P. K. Chan and M. V. Mahoney. Modeling multiple time series for anomaly detection. In *Proceedings of the 5th IEEE International Conference on Data Mining*, pages 90–97, 2005. ISBN 1550-4786.

[20] V. Chandola. *Anomaly Detection for Symbolic Sequences and Time Series Data*. PhD thesis, University of Minnesota, 2009.

[21] V. Chandola and V. Kumar. Summarization - compressing data into an informative representation. *Knowledge and Information Systems*, 12(3):355–378, 2007.

[22] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):1–58, 2009. ISSN 0360-0300.

[23] R. Charbonneau and G. Kondolf. Land use change in California, USA: Nonpoint source water quality impacts. *Environmental Management*, 17(4):453–460, 1993.

[24] C. Chatfield. *Time-series forecasting*. Chapman & Hall/CRC, 2001.

[25] C. Chatfield. *The Analysis of Time Series: An Introduction*. Chapman & Hall/CRC, 2004.

[26] D. Cheboli, V. Chandola, and V. Kumar. Anomaly detection for time series: A survey. Technical Report in progress, University of Minnesota, Department of Computer Science and Engineering, 2010.

[27] J. Chen and A. Gupta. On change point detection and estimation. *Communications in Statistics: Simulation & Computation*, 30(3):665–697, 2001.

[28] J. Chen and A. K. Gupta. Testing and locating variance changepoints with application to stock prices. *Journal of the American Statistical Association*, 92(438): 739–747, 1997.

[29] J. Chen, P. Jönsson, M. Tamura, Z. Gu, B. Matsushita, and L. Eklundh. A simple method for reconstructing a high-quality NDVI time-series data set based on the Savitzky-Golay filter. *Remote Sensing of Environment*, 91(3-4):332 – 344, 2004. ISSN 0034-4257.

[30] H. Chernoff and S. Zacks. Estimating the current mean of a normal distribution which is subjected to changes in time. *The Annals of Mathematical Statistics*, 35 (3):999–1018, 1964.

[31] S. Chib. Estimation and comparison of multiple change-point models. *Journal of Econometrics*, 86(2):221–241, 1998.

[32] W. Cohen and M. Fiorella. Comparison of methods for detecting conifer forest change with thematic mapper imagery. In *Remote sensing change detection: environmental monitoring methods and applications. Chelsea,(MI): Sleeping Bear Press. p*, pages 89–102. Ann Arbor Press, Chelsea, MI, 1998.

[33] P. Coppin, I. Jonckheere, K. Nackaerts, B. Muys, and E. Lambin. Digital change detection methods in ecosystem monitoring: a review. *International Journal of Remote Sensing*, 25(9):1565–1596, 2004.

[34] M. Csörgő and L. Horváth. Nonparametric methods for changepoint problems. In P. R. Krishnaiah and C. R. Rao, editors, *Quality Control and Reliability*, volume 7 of *Handbook of Statistics*, pages 403–425. North-Holland, 1988.

[35] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[36] R. Dickinson and P. Kenneday. Impacts on regional climate of Amazon deforestation. *Geophysical Research Letters*, 19(19):1947–1950, 1992.

[37] A. Dries and U. Rückert. Adaptive concept drift detection. In *SDM 2009: Proceedings of the 9th SIAM International Conference on Data Mining*, 2009.

[38] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wily & Sons, Inc., New York, second edition, 2001.

[39] Y. Fang, A. R. Ganguly, N. Singh, V. Vijayaraj, N. Feierabend, and D. T. Potere. Online change detection: Monitoring land cover from remotely sensed data. In *ICDM Workshops*, pages 626–631, 2006.

[40] L. Firoiu and P. R. Cohen. Segmenting time series with a hybrid neural networks - hidden markov model. In *Eighteenth national conference on Artificial intelligence*, pages 247–252, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[41] J. A. Foley, R. DeFries, G. P. Asner, C. Barford, G. Bonan, S. R. Carpenter, F. S. Chapin, M. T. Coe, G. C. Daily, H. K. Gibbs, J. H. Helkowski, T. Holloway, E. A. Howard, C. J. Kucharik, C. Monfreda, J. A. Patz, I. C. Prentice, N. Ramankutty, and P. K. Snyder. Global Consequences of Land Use. *Science*, 309(5734):570–574, 2005.

[42] G. M. Foody. Approaches for the production and evaluation of fuzzy land cover classifications from remotely-sensed data. *International Journal of Remote Sensing*, 17(7):1317–1340, 1996.

[43] M. Friedl and C. Brodley. Decision tree classification of land cover from remotely sensed data. *Remote Sensing of Environment*, 61(3):399–409, 1997.

[44] M. Fuentes. Approximate likelihood for large irregularly spaced spatial data. *Journal of the American Statistical Association*, 102(477):321–331, 2007.

[45] X. Ge and P. Smyth. Segmental semi-Markov models for change-point detection with applications to semiconductor manufacturing. Technical Report UCI-ICS 00-08, University of California, Irvine, 2000.

[46] N. Golyandina and E. Osipov. The "caterpillar"-ssa method for analysis of time series with missing values. *Journal of Statistical Planning and Inference*, 137(8): 2642 – 2653, 2007. ISSN 0378-3758. 5th St. Petersburg Workshop on Simulation.

[47] E. Gombay. Change detection in autoregressive time series. *Journal of Multivariate Analysis*, 99(3):451–464, 2008.

[48] R. E. Gullison, P. C. Frumhoff, J. G. Canadell, C. B. Field, D. C. Nepstad, K. Hayhoe, R. Avissar, L. M. Curran, P. Friedlingstein, C. D. Jones, and C. Nobre. Tropical forests and climate policy. *Science*, 316(5827):985–986, 2007.

[49] V. Guralnik and J. Srivastava. Event detection from time series data. In *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 33–42, New York, NY, USA, 1999. ACM Press.

[50] F. Gustafsson. *Adaptive Filtering and Change Detection.* John Wiley & Sons, 2000.

[51] S. B. Guthery. Partition regression. *Journal of the American Statistical Association*, 69(348):945–947, 1974.

[52] J. Han, M. Kamber, and A. K. H. Tung. Spatial clustering methods in data mining. In H. J. Miller and J. Han, editors, *Geographic Data Mining and Knowledge Discovery*, pages 188–217. Taylor & Francis, 2001.

[53] M. C. Hansen, S. V. Stehman, P. V. Potapov, T. R. Loveland, J. R. G. Townshend, R. S. DeFries, K. W. Pittman, B. Arunarwati, F. Stolle, M. K. Steininger, M. Carroll, and C. DiMiceli. Humid tropical forest clearing from 2000 to 2005 quantified by using multitemporal and multiresolution remotely sensed data. *Proceedings of the National Academy of Sciences*, 105(27):9439–9444, 2008.

[54] D. M. Hawkins. Point estimation of the parameters of piecewise regression models. *Applied Statistics*, 25(1):51–57, 1976.

[55] D. M. Hawkins. Fitting multiple change-point models to data. *Computational Statistics & Data Analysis*, 37(3):323–341, 2001.

[56] D. M. Hawkins and D. F. Merriam. Optimal zonation of digitized sequential data. *Mathematical Geology*, 5(4):389–395, 1973.

[57] A. Henderson-Sellers, R. E. Dickinson, T. B. Durbidge, P. J. Kennedy, K. McGuffie, and A. J. Pitman. Tropical deforestation: Modeling local- to regional-scale climate change. *Journal of Geophysical Research*, 98:7289–7315, 1993.

[58] J. Himberg, K. Korpiaho, H. Mannila, J. Tikanmäki, and H. Toivonen. Time series

segmentation for context recognition in mobile devices. In *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 203–210, Washington, DC, USA, 2001. IEEE Computer Society.

[59] J. N. Hird and G. J. McDermid. Noise reduction of ndvi time series: An empirical comparison of selected techniques. *Remote Sensing of Environment*, 113(1):248–258, 2009.

[60] S.-S. Ho. A martingale framework for concept change detection in time-varying data streams. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 321–327, New York, NY, USA, 2005. ACM.

[61] J. Honaker and G. King. What to do about missing values in time series cross-section data. *American Journal of Political Science*, to appear, 2010.

[62] N. J. Horton and K. P. Kleinman. Much ado about nothing: A comparison of missing data methods and software to fit incomplete data regression models. *The American Statistician*, 61(1):79–90, 2007.

[63] A. Huete, K. Didan, T. Miura, E. P. Rodriguez, X. Gao, and L. G. Ferreira. Overview of the radiometric and biophysical performance of the MODIS vegetation indices. *Remote Sensing of Environment*, 83(1-2):195–213, 2002.

[64] M. Hušková. Nonparametric procedures for detecting a change in simple linear regression models. *Applied Change Point Problems in Statistics*, pages 15–36, 1993.

[65] T. Idé and H. Kashima. Eigenspace-based anomaly detection in computer systems. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 440–449, New York, NY, USA, 2004. ACM Press. ISBN 1-58113-888-1.

[66] C. Inclán and G. C. Tiao. Use of cumulative sums of squares for retrospective detection of changes of variance. *Journal of the American Statistical Association*, 89(427):913–923, 1994.

[67] V. K. Jandhyala, S. B. Fotopoulos, and D. M. Hawkins. Detection and estimation of abrupt changes in the variability of a process. *Computational Statistics & Data Analysis*, 40(1):1–19, 2002.

[68] S. Jin and S. A. Sader. Modis time-series imagery for forest disturbance detection and quantification of patch size effects. *Remote Sensing of Environment*, 99(4): 462–470, 2005.

[69] G. Johannesson. Fixed rank kriging for very large spatial data sets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(18):209–226, 2008.

[70] R. H. Jones. Maximum likelihood fitting of arma models to time series with missing observations. *Technometrics*, 22(3):389–395, 1980.

[71] P. d. Jong. Smoothing and interpolation with the state-space model. *Journal of the American Statistical Association*, 84(408):1085–1088, 1989.

[72] T. Kasetkasem and P. Varshney. An image change detection algorithm based on markov random field models. *IEEE Transactions on Geoscience and Remote Sensing*, 40(8):1815–1823, 2002. ISSN 0196-2892. doi: 10.1109/TGRS.2002.802498.

[73] E. Keogh, S. Chu, D. Hart, and M. Pazzani. An online algorithm for segmenting time series. In *ICDM 2001: Proceedings of the first IEEE International Conference on Data Mining*, pages 289–296, 2001.

[74] E. Keogh, S. Chu, D. Hart, and M. Pazzani. Segmenting time series: A survey and novel approach. In *Data mining in Time Series Databases*. World Scientific Publishing Company, 2003.

[75] J. Knight, R. Lunetta, J. Ediriwickrema, and S. Khorram. Regional scale land cover characterization using modis-ndvi 250 m multi-temporal imagery: A phenology-based approach. *GIScience & Remote Sensing*, 43(1):1–23, 2006.

[76] P. R. Krishnaiah and B. Q. Miao. Review about estimation of change points. In P. R. Krishnaiah and C. R. Rao, editors, *Quality Control and Reliability*, volume 7 of *Handbook of Statistics*, pages 375–402. North-Holland, 1988.

[77] B. Krishnamurthy, S. Sen, Y. Zhang, and Y. Chen. Sketch-based change detection: methods, evaluation, and applications. In *IMC '03: Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, pages 234–247, New York, NY, USA, 2003. ACM.

[78] J. Kucera, P. Barbosa, and P. Strobl. Cumulative sum charts - a novel technique for processing daily time series of modis data for burnt area mapping in portugal. In *MultiTemp 2007: International Workshop on the Analysis of Multi-temporal Remote Sensing Images*, pages 1–6, 2007.

[79] M. Kulldorff. A spatial scan statistic. *Communications in Statistics-Theory and Methods*, 26(6):1481–1496, 1997.

[80] T. L. Lai. Sequential changepoint detection in quality control and dynamical systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57 (4):613–658, 1995.

[81] E. F. Lambin and A. H. Strahlers. Change-vector analysis in multitemporal space: A tool to detect and categorize land-cover change processes using high temporal-resolution satellite data. *Remote Sensing of Environment*, 48(2):231–244, 1994.

[82] R. J. A. Little and D. B. Rubin. *Statistical Analysis with Missing Data.* John Wiley & Sons, 1987.

[83] Y. Liu, S. Nishiyama, and T. Yano. Analysis of four change detection algorithms in bi-temporal space with a case study. *International Journal of Remote Sensing*, 25:2121–2139, 2004.

[84] D. G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31(3):355–395, 1987.

[85] D. Lu, P. Mausel, E. Brondízio, and E. Moran. Change detection techniques. *International Journal of Remote Sensing*, 25(12):2365–2401, 2003.

[86] J. M. Lucas and M. S. Saccucci. Exponentially weighted moving average control schemes: Properties and enhancements. *Technometrics*, 32(1):1–12, 1990.

[87] R. S. Lunetta, J. F. Knight, J. Ediriwickrema, J. G. Lyon, and L. D. Worthy. Land-cover change detection using multi-temporal MODIS NDVI data. *Remote Sensing of Environment*, 105(2):142–154, 2006.

[88] R. Macleod and R. Congalton. A quantitative comparison of change-detection algorithms for monitoring eelgrass from remotely sensed data. *Photogrammetric Engineering and Remote Sensing*, 64(3):207–216, 1998.

[89] J. F. Mas. Monitoring land-cover changes: a comparison of change detection techniques. *International Journal of Remote Sensing*, 20(1):139–152, 1999.

[90] F. Melgani, G. Moser, and S. B. Serpico. Unsupervised change-detection methods for remote-sensing images. *Optical Engineering*, 41(12):3288–3297, 2002.

[91] V. Moskvina and A. Zhigljavsky. An algorithm based on singular spectrum analysis for change-point detection. *Communications in Statistics: Simulation & Computation*, 32(2):319–352, 2003.

[92] A. A. Nielsen, K. Conradsen, and J. J. Simpson. Multivariate alteration detection (mad) and maf postprocessing in multispectral, bitemporal image data: New approaches to change detection studies. *Remote Sensing of Environment*, 64(1):1 – 19, 1998.

[93] T. Ogden and E. Parzen. Change-point approach to data analytic wavelet thresholding. *Statistics and Computing*, 6(2):93–99, 1996.

[94] S. V. Ollinger, A. D. Richardson, M. E. Martin, D. Y. Hollinger, S. E. Frolking, P. B. Reich, L. C. Plourde, G. G. Katul, J. W. Munger, R. Oren, M.-L. Smith, K. T. Paw U, P. V. Bolstad, B. D. Cook, M. C. Day, T. A. Martin, R. K. Monson, and H. P. Schmid. Canopy nitrogen, carbon assimilation, and albedo in temperate and boreal forests: Functional relations and potential climate feedbacks. *Proceedings of the National Academy of Sciences*, 105(49):19336–19341, 2008.

[95] E. S. Page. Continuous inspection schemes. *Biometrika*, 41(1/2):100–115, 1954.

[96] E. S. Page. On problems in which a change in a parameter occurs at an unknown point. *Biometrika*, 44(1/2):248–252, 1957.

[97] N. Parikh and N. Sundaresan. Scalable and near real-time burst detection from ecommerce queries. In *KDD '08: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 972–980, 2008.

[98] D. Picard. Testing and estimating change-points in time series. *Advances in Applied Probability*, 17(4):841–867, 1985.

[99] D. Preston, P. Protopapas, and C. Brodley. Event discovery in time series. In *SDM 2009: Proceedings of the 9th SIAM International Conference on Data Mining*, 2009.

[100] R. Radke, S. Andra, O. Al-Kofahi, and B. Roysam. Image change detection algorithms: a systematic survey. *IEEE Transactions on Image Processing*, 14(3): 294–307, 2005.

[101] A. E. Raftery. Change point and change curve modeling in stochastic processes and spatial statistics. Technical Report 253, University of Washington, Department of Statistics, 1993.

[102] N. Ramankutty, H. K. Gibbs, F. Achard, R. Defries, J. A. Foley, and R. A. Houghton. Challenges to estimating carbon emissions from tropical deforestation. *Global Change Biology*, 13:51–66, January 2007.

[103] S. W. Roberts. Control chart tests based on geometric moving averages. *Technometrics*, 1(3):239–250, 1959.

[104] G. J. Roerink, M. Menenti, and W. Verhoef. Reconstructing cloudfree ndvi composites using fourier analysis of time series. *International Journal of Remote Sensing*, 21(9):1911–1917, 2000.

[105] J. Rogan, J. Franklin, and D. A. Roberts. A comparison of methods for monitoring multitemporal vegetation change using thematic mapper imagery. *Remote Sensing of Environment*, 80(1):143 – 156, 2002. ISSN 0034-4257.

[106] D. P. Roy, P. E. Lewis, and C. O. Justice. Burned area mapping using multi-temporal moderate spatial resolution data–a bi-directional reflectance model-based expectation approach. *Remote Sensing of Environment*, 83(1-2):263–286, 2002.

[107] D. Rubin. *Multiple imputation for nonresponse in surveys.* John Wiley & Sons, 2004.

[108] R. A. Schowengerdt. *Remote sensing: models and methods for image processing.* Academic Press, 2007.

[109] L. Schumaker. *Spline Functions: Basic Theory.* Cambridge University Press, 2007.

[110] T. Searchinger, R. Heimlich, R. A. Houghton, F. Dong, A. Elobeid, J. Fabiosa, S. Tokgoz, D. Hayes, and T.-H. Yu. Use of U.S. Croplands for Biofuels Increases Greenhouse Gases Through Emissions from Land-Use Change. *Science*, 319(5867): 1238–1240, 2008.

[111] A. Sen and M. S. Srivastava. On tests for detecting change in mean. *The Annals of Statistics*, 3(1):98–108, 1975.

[112] M. Severo and J. Gama. Change detection with kalman filter and cusum. *Discovery Science*, pages 243–254, 2006.

[113] S. A. Shaban. Change point problem and two-phase regression: An annotated bibliography. *International Statistical Review*, 48(1):83–93, 1980.

[114] M. Sharifzadeh, F. Azmoodeh, and C. Shahabi. Change detection in time series data using wavelet footprints. *Advances in Spatial and Temporal Databases*, pages 127–144, 2005.

[115] R. Shumway and D. Stoffer. An approach to time series smoothing and forecasting using the em algorithm. *Journal of Time Series Analysis*, 3(4):253–264, 1982.

[116] A. Singh. Digital change detection techniques using remotely-sensed data. *International Journal of Remote Sensing*, 10(6):989–1003, 1989.

[117] N. Sugiura and R. T. Ogden. Testing change-points with linear trend. *Communications in Statistics - Simulation and Computation*, 23(2):287–322, 1994.

[118] W. C. Sullivan, O. M. Anderson, and S. T. Lovell. Agricultural buffers at the rural-urban fringe: an examination of approval by farmers, residents, and academics in the midwestern United States. *Landscape and Urban Planning*, 69(2-3):299–313, 2004.

[119] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, Boston, MA, 2006.

[120] J. Townshend and C. Justice. Analysis of the dynamics of African vegetation using the normalized difference vegetation index. *International Journal of Remote Sensing*, 7(11):1435–1445, 1986.

[121] A. Tsymbal. The problem of concept drift: definitions and related work. Technical Report TCD-CS-2004-15, Computer Science Department, Trinity College, Dublin, 2004.

[122] S. V. Vaseghi. *Advanced Digital Signal Processing and Noise Reduction*. John Wiley & Sons, 2005.

[123] W. Verhoef, M. Menenti, and S. Azzali. A colour composite of NOAA-AVHRR-NDVI based on time series analysis (1981-1992). *International Journal of Remote Sensing*, 17(2):231–235, 1996.

[124] N. Viovy, O. Arino, and A. S. Belward. The best index slope extraction (bise): A method for reducing noise in ndvi time-series. *International Journal of Remote Sensing*, 13(8):1585–1590, 1992.

[125] D. A. Wolfe and E. Schechtman. Nonparametric statistical procedures for the changepoint problem. *Journal of Statistical Planning and Inference*, 9(3):389–396, 1984.

[126] K. Yamanishi and J. Takeuchi. A unifying framework for detecting outliers and change points from non-stationary time series data. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and*

data mining, pages 676–681, New York, NY, USA, 2002. ACM Press. ISBN 1-58113-567-X.

[127] S. Zacks. Survey of classical and Bayesian approaches to the change-point problem: Fixed sample and sequential procedures of testing and estimation. In M. H. Rizvi, J. S. Rustagi, and D. Siegmund, editors, *Recent Advances in Statistics*, pages 245–269. Academic Press, 1983.