DEGREE PROJECT IN MATHEMATICS,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2017

Detecting change points in remote sensing time series

ANNA LUNDEMO

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ENGINEERING SCIENCES

Detecting change points in remote sensing time series

ANNA LUNDEMO

Degree Projects in Mathematical Statistics (30 ECTS credits)
Degree Programme in Applied and Computational Mathematics (120 credits)
KTH Royal Institute of Technology year 2017
Supervisors at Alfred Wegener Institute, Potsdam: Guido Grosse
Supervisor at KTH: Jimmy Olsson
Examiner at KTH: Jimmy Olsson

TRITA-MAT-E 2017:51
ISRN-KTH/MAT/E--17/51--SE

Royal Institute of Technology
School of Engineering Sciences
KTH SCI
SE-100 44 Stockholm, Sweden
URL: www.kth.se/sci

Abstract

We analyse methods for detecting change points in optical remote sensing lake drainage time series. Change points are points in a data set where the statistical properties of the data change. The data that we look at represent drained lakes in the Arctic hemisphere. It is generally noisy, with observations missing due to difficult weather conditions. We evaluate a partitioning algorithm, with five different approaches to model the data, based on least-squares regression and an assumption of normally distributed measurement errors. We also evaluate two computer programs called DBEST and TIMESAT and a MATLAB function called *findchangepts()*. We find that TIMESAT, DBEST and the MATLAB function are not relevant for our purposes. We also find that the partitioning algorithm that models the data as normally distributed around a piecewise constant function, is best suited for finding change points in our data.

Sammanfattning

Vi analyserar metoder för att hitta brytpunkter i optisk fjärranalysdata som beskriver uttorkning av sjöar. Brytpunkter är punkter i en tidsserie vid vilka de statistiska egenskaperna förändras. Datan som vi använder representerar uttorkande sjöar i norra hemisfären. Den är generellt väldigt fluktuerande och många observationer kan saknas på grund av väderförhållandena i dessa områden. Vi undersöker en partitionsalgoritm, med fem olika sätt att modellera datan, baserade på minstakvadratmetoden och på antagande om att mätfelen är normalfördelade. Vi utvärderar också två program som heter DBEST och TIMESAT och en funktion i MATLAB som heter *findchangepts()*. Vi kommer fram till att TIMESAT, DBEST och MATLAB-funktionen inte är lämpliga för att analysera vår data. Vår slutsats är också att partitionsalgoritmen som modellerar datan som normalfördelad runt en styckvis konstant funktion bäst lämpar sig för att hitta brytpunkter i vår data.

Contents

Chapter 1

Introduction

In this study, different methods for analysing time series of remote sensing data will be evaluated. The aim is to compare methods that detect so called change points in time series of optical remote sensing data from permafrost regions. The focus will be on changes in regression coefficients, mean and variance. We will investigate how well the methods find change points in both simulated and real-world data and select those methods that are most relevant for the specific purpose of finding change points in satellite-based optical remote sensing data from permafrost regions.

In the last decades, the Arctic regions have experienced a distinct increase in air temperature [20, p. 27]. The Arcitc regions are sensitive to climate change and global warming [8, p. 326], therefore these regions are interesting areas for research. Especially, low lying river deltas are sensitive to temperature changes [20, p. 27]. In permafrost regions, so called thermokarst lakes are commonly occurring. The growth and drainage of thermokarst lakes is highly influencing the degradation of permafrost in these regions [8, p. 326]. Thawing permafrost generally results in expanding thermokarst lakes and erosion of land surface. Erosion of solid ground can result in drainage of nearby thermokarst lakes. The solid permafrost ground in the bottom of the lake is then exposed to the outer climate conditions, and the permafrost is likely to thaw. In this study we will look at data from draining lakes gathered by satellite-based remote sensing, hereafter called lake drainage data. The data has been found in the Landsat archive of optical remote sensing data. Satellites are registering reflected light such as visible light, near infrared light and short-waved infrared light, from the surface of the earth [20]. We will look at six different indices of characteristics, namely the normalized difference vegetation index; NDVI, the normalized difference moisture index; NDMI, the normalized difference wetness index; NDWI and the so called Tasseled Cap indices TCG, TCB and TCW. TCG is a measure of vegetation, TCB is a measure of brightness and TCW is a measure of wetness. The indices are further described in Chapter 2.

Thorough studies of the spatial patterns of these indices have been done by Nitze and Grosse in [20], covering parts of the Lena Delta in Siberia. Different environmental events

such as drainage of lakes, fires and costal erosion have been studied. However, the knowledge of the temporal patterns is not as vast. Consequently, we have information on what events that have taken place, e.g. drainage of thermokarst lake, and we are interested in analysing at what time the events occurred [20, p. 39]. Therefore, time series analysis of the temporal data, in particularly change point detection, will be carried out in this study. Change point detection is finding the points of a time series at which the structural patterns change. Chen and Gupta states in [4, p. ix] "From the statistical point of view, a change point is a place or time point such that the observations follow one distribution up to that point and follow another distribution after that point." In analysing climate data it is essential to know both where and when an event has occurred. Therefore, change point detection is highly applicable on remote sensing climate data from permafrost regions, but is also relevant in all type of time series analysis. Before doing further analysis of time series it is often essential to first find change points in the time series [2]. When change points have been detected, it is assumed that the data in each segment share some statistical property.

There exist many methods for detecting change points. For example, time series segmentation is an approach where the time series is divided into different segments and to each such segment a model is fitted [10]. Decomposition methods decomposes the time series and describe it by its trend, its seasonal component and the remaining components [10]. These methods can be used together to find patterns in time series. Trends can for example be linear or quadratic and describe how the overall pattern changes over time. Seasonal patterns are periodic, with patterns returning according to a period time, for example annually. When trends and possible seasonal components have been removed from the time series, the remaining component, also called the random component, is analysed. It is desirable that the random component has stationary properties [2]. If the random component is stationary, we have been successful in removing trends and seasonal components from the time series [2], also meaning that we successfully have found change points.

Our data is gathered in permafrost regions in the northern hemisphere. Permafrost is defined as ground that stays below $0°C$ for at least two consecutive years [22, p. 308]. The permafrost regions are often difficult to access. The climate can be harsh and expeditions to these regions expensive. Therefore, gathering data by hand can lead to small data sets, only covering small regions and with short time spans. However, satellite-based remote sensing is an alternative way of collecting data from these regions. There are more than 1300 satellites orbiting the earth collecting data of the climate conditions on the surface of the earth. Using remote sensing, the potential to analyse the climate in remote regions increases substantially. The satellites can easily cover very large regions and continuously gather information about the climate conditions [7]. It opens up for a large database of data covering permafrost regions, reaching over a time span of more than 40 years [20, p. 28]. However, satellite-based optical remote sensing also has its drawbacks. Several factors such as cloud and snow coverage of the ground or particles in the air, can impact the satellites' view of the earth. Also, noise due to other factors such as low sun angles is a

common issue. Therefore, time series of remote sensing satellite data often have so called missing data points, meaning that the time span between the data points is not equal throughout the time series [22]. This may cause complications in the analysis process, especially since there will be information missing. Furthermore, the disturbances cause noisy data with a high risk of unreliable data points, called outliers [20, 15].

First, the data is presented in Chapter 2. In Chapter 3, we present the methods that we analyse. First, we present a partitioning algorithm called optimal partioning, including different models to fit to the data. A MATLAB function *findchangepts()* and the two computer programs DBEST, TIMESAT are also presented. DBEST is claimed to be a fast and accurate method for detecting change points in satellite-based remote sensing data of vegetation indices [10, p. 182]. TIMESAT is a computer program developed by Jönsson and Eklundh, which uses least-squares techniques, such as Savitzky-Golay filters to approximate the time series [14]. In Chapter 4, the different methods are evaluated on a data set describing the event of lake drainage and some methods will be left out from further analysis. Further, in Chapter 5 and 6, the remaining methods are evaluated on simulated data sets. In Chapter 7, we evaluate the remaining methods on several sets of lake drainage data and in Chapter 8 we will discuss the problem of outliers and evaluate the remaining methods on filtered data. Some examples of found change points and fitted trends will be shown in Chapter 9. Finally, in Chapter 10, we have a summarized discussion of our results.

Chapter 2

Data

The original images from which the data sets used in this study are extracted are available in the Landsat archive. The Landsat program is a cooperation between NASA and USGS (United States Geological Survey) and consists of the longest continuously collected satellite data set describing the environmental conditions on the surface of the earth [24]. Different sensors on board Landsat satellites register images of the surface, with a spatial resolution of 30 meters [20, p. 29]. The images are results of reflected sun light, registered for different wavelengths. The images cover large regions but certain points in space have been selected as reference points for specific land cover conditions. The images can be masked to remove erroneous observations, due to for example clouds or snow cover, aiming to create images only containing clear observations with no disturbances. However, the masking is not flawless and the images may contain so called outliers, which are data points that due to erroneous observations deviate from the rest of the data set. Time series have been extracted from these images and data sets from different locations have been used in this study. With each data set comes information about the location and time point of the observation. Some data sets have information on when a change occurred. The information is manually found and there is only one change detected in each data set. Therefore, this information is not always reliable. There is also information about the sensors that have registered the observations. The different sensors used by the Landsat satellites are the Thematic Mapper (TM), the Enhanced Thematic Mapper Plus (ETM+) and the Observing Land Imager (OLI), see [20, section 2.2]. Data is gathered at certain locations and over a time span of 30-40 years. Different indices that represent different properties of the surface, for example vegetation or wetness, have been extracted from the images, based on the amount of sun light reflected from the earth. Each set of data that we will be working with, contains time series of six different multi-spectral indices, TCW, TCG, NDMI, NDVI, NDWI and TCB, that represent different properties of the surface of the earth. NDVI stands for Normalized Difference Vegetation Index and measures the amount of vegetation. The index is calculated as

$$\text{NDVI} = \frac{\rho_{nir} - \rho_r}{\rho_{nir} + \rho_r},$$

where ρ_{nir} is the proportion of near-infrared sun light that is reflected from the earth, and ρ_r is the proportion of visible red light that is reflected. If there is a lot of vegetation, more near-infrared light is reflected and more visible red light is absorbed, compared to a surface where there is little vegetation [20, 23]. Therefore, the value of NDVI is higher where there is more vegetation and lower where there is less vegetation. For all indices, the value is higher if the characteristic represented by the index is more present. The values reach between -1 and 1. NDWI stands for Normalized Difference Water Index and measures water content in bodies containing water.

$$\text{NDWI} = \frac{\rho_g - \rho_{nir}}{\rho_{nir} + \rho_g},$$

where ρ_g is the proportion of green light that is reflected [20, 6]. NDMI, Normalized Difference Moisture Index, measures moisture in vegetation,

$$\text{NDMI} = \frac{\rho_{nir} - \rho_{swir_1}}{\rho_{nir} + \rho_{swir_1}},$$

where $swir_1$ is short-waved infrared light [20]. TCG, TCW and TCB are the so called Tasseled Cap indices greenness, wetness and brightness respectively. They are calculated for Landsat data and are weighted sums of reflectance coefficients ρ for certain wavelengths [20, p. 30].

As mentioned, the signal of optical satellite data can be affected or disturbed by several influencing factors. Only some disturbances can be masked out, which leads to missing or noisy data. In the regions where our data is gathered, clouds and snow are frequent desturbances. Furthermore, air particles and shadows can affect the view over the surface of the earth and particles in the air or on the ground can distort the reflected sun light. Disturbances and malfunctions in the measurement equipment, as well as other factors, also affect the satellite recorded images [20, 23]. Especially the ETM+ sensor has been affected by this [20]. Assuming that the magnitudes of the different errors are approximately the same and that the errors are independent, the errors can be modelled with support from the Central Limit Theorem. The Central Limit Theorem states that the sum of many independent and approximately equally distributed random variables is approximately normally distributed [1, p. 142]. For that reason, a general assumption is that measurement errors are normally distributed with mean 0 and standard deviation σ. This motivates why we can assume that the measurement errors in the remote sensing data that we are studying are approximately equally distributed and thus that the data, by the central limit theorem, is approximately normally distributed.

Moreover, regarding the change points to be searched for in the lake drainage data, it is reasonable but not always true that the indices TCW, TCB, NDMI and NDWI will change simultaneously, whereas the vegetation indices TCG and NDVI might have a delay, since the vegetation may need time to develop after a lake drainage. This conclusion was made from a private conversation with Ingmar Nitze.

Chapter 3

Methods

Different methods for analysing time series of remote sensing data will be presented in this chapter. First, we introduce a dynamical programming algorithm. Dynamical programming is a method to solve large problems using the solutions to smaller sub problems. The algorithm that will be used here is called optimal partitioning and is found for example in [9]. The algorithm finds a global optimum of a specified cost function, describing how well a model has been applied to the data [16]. Different cost functions will also be presented and further described in Chapter 4. Secondly, we introduce a MATLAB function, *findchangepts()*, which is created to find abrupt changes in data and based on the findings in [16] and [17]. Third, DBEST will be presented. According to the creators, it is a "user-friendly program for analysing vegetation time series" [10, p. 182] and was first presented in 2004. Lastly, TIMESAT, "a program for analyzing time-series of satellite sensor data" [14, p. 833] is shortly described.

3.1 Optimal partitioning

Optimal partitioning is described in [16] and [9] and is a dynamical programming algorithm that finds the points that best divide a data set into several segments, by minimizing a defined function describing the deviation from a model to the data, here called a cost function. Dynamical programming solves the computationally heavy problem of localising several dependent points in a large data set, in an efficient way. The computational cost for the optimal partitioning method is $\mathcal{O}(n^2)$, n being the total number of observations in the time series [16]. A chosen model is fitted to the time series, going through all possible combination of change points, and a cost function describing the error of the model fit, is minimised. The optimal partitioning algorithm finds the global minimum of the cost function, given the number of change points to be searched for [16, p. 8].

The optimal partitioning algorithm is described in [16, p. 8] as follows. Let the data points in the time series be $(x_1, y_1), (x_2, y_2) \ldots, (x_n, y_n)$. Let $\tau_1, \tau_2, \ldots, \tau_k$ be the indices of the k candidate change points, with $\tau_i < \tau_j$ if and only if $i < j$ and define $\tau_0 = 0$ and

$\tau_{k+1} = n$. The aim is to minimise

$$\sum_{i=1}^{k+1} C(x_{\tau_{i-1}+1:\tau_i}, y_{\tau_{i-1}+1:\tau_i}) + \gamma,$$

where γ is a penalty constant and $C(x_{\tau_{i-1}+1:\tau_i}, y_{\tau_{i-1}+1:\tau_i})$ is the cost for the segment between change point τ_{i-1} and change point τ_i. The penalty constant is the penalty of adding a change point to the model. The aim is to find $F(n)$ recursively, with

$$F(s) = \min_{t<s}[F(t) + C(x_{t+1:s}, y_{t+1:s}) + \gamma],$$

for $s = 1, 2, \ldots, n$, where $F(s)$ is the optimal partitioning of the points $(x_1, y_1), (x_2, y_2), \ldots, (x_s, y_s)$ and $F(t)$, $t < s$, is the optimal partitioning of the points $(x_1, y_1), (x_2, y_2), \ldots, (x_t, y_t)$. The optimal partitioning of all points $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, $F(n)$, thus depends on the optimal partitioning of the former points. The algorithm to find $F(n)$ is as follows. Let $F(0) = -\gamma$ and cp(0)=NULL. Then, calculate for $\tau^* = 1, 2, \ldots, n$,

$$\begin{aligned} F(\tau^*) &= \min_{0\leq\tau<\tau^*}[F(\tau) + C(x_{\tau+1:\tau^*}, y_{\tau+1:\tau^*}) + \gamma], \\ \tau' &= \operatorname*{argmin}_{0\leq\tau<\tau^*}[F(\tau) + C(x_{\tau+1:\tau^*}, y_{\tau+1:\tau^*}) + \gamma], \\ \mathrm{cp}(\tau^*) &= (\mathrm{cp}(\tau'), \tau'), \end{aligned} \tag{3.1}$$

where $\mathrm{cp}(\tau^*)$ are the positions of the change points for the data set $(x_1, y_1), (x_2, y_2), \ldots, (x_{\tau^*}, y_{\tau^*})$ and cp(n) are thus the positions of all the found change points in the entire data set. For each point τ^* in the data set, the optimal partition of the points $(x_1, y_1), (x_2, y_2), \ldots, (x_{\tau^*}, y_{\tau^*})$ is calculated, and we notice that it is dependent on the optimal partition of the former points. The algorithm is further described in [16, 17].

The penalty constant γ adjusts the trade off between the cost function and the number of change points. The penalty constant is set by the user to determine the number of change points to be searched for. The relation between the penalty constant and the number of change points is highly dependent on the data. A higher value of the penalty constant results in fewer change points. Fewer change points result in lower variance of the model describing the time series and thus higher bias. Therefore, the penalty constant adjusts the bias-variance trade off. The penalty that is added to the total value of the cost function when one change point is added to the model, equals the penalty constant. Setting γ to zero means that adding a change point adds no penalty to the function $F(n)$, and the data will be divided into as small segments as possible. A change point is only added to the model if it results in that the total cost of the model decreases with at least the value of γ.

3.1.1 The RSS cost functions

A set of cost functions can be created by fitting a polynomial to the time series, using least-squares methods. Call the segments $\kappa_j = \{\tau_{j-1}, \tau_{j-1}+1, \ldots, \tau_j\}$, for $j = 1, 2, \ldots, k+1$. The

data set $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ is approximated by a piecewise continuous function $p(x, \bar{\beta}_0, \bar{\beta}_1, \bar{\beta}_2)$, given by a second order polynomial on each interval,

$$p(x, \bar{\beta}_0, \bar{\beta}_1, \bar{\beta}_2) = \beta_{0,j} + \beta_{1,j} x_i + \beta_{2,j} x_i^2, \quad \forall i \in \kappa_j,$$

for $j = 1, 2, \ldots, k+1$, where

$$\begin{aligned}
\bar{\beta}_0 &= (\beta_{0,1}, \beta_{0,2}, \ldots, \beta_{0,k+1}), \\
\bar{\beta}_1 &= (\beta_{1,1}, \beta_{1,2}, \ldots, \beta_{1,k+1}), \\
\bar{\beta}_2 &= (\beta_{2,1}, \beta_{2,2}, \ldots, \beta_{2,k+1}).
\end{aligned}$$

The parameters $\bar{\beta}_0$, $\bar{\beta}_1$ and $\bar{\beta}_2$ are estimated by minimising the residual sum of squares (RSS),

$$\mathrm{RSS}(\bar{\beta}_0, \bar{\beta}_1, \bar{\beta}_2) = \sum_{i=1}^{n} (y_i - p(x_i, \bar{\beta}_0, \bar{\beta}_1, \bar{\beta}_2))^2,$$

where the residual $\epsilon_i = y_i - p(x_i, \bar{\beta}_0, \bar{\beta}_1, \bar{\beta}_2)$ is assumed to be normally distributed with constant variance σ^2, i.e. $\epsilon_i \sim N(0, \sigma^2), i = 1, 2, \ldots, n$, as described in [3, p. 63]. Let us write this in the notation of equation (3.1). Fixing τ and τ^*, the optimal parameter values were found by minimising the RSS,

$$\begin{aligned}
\mathrm{RSS} &= \sum_{i=\tau+1}^{\tau^*} \epsilon_i^2 = \sum_{i=\tau+1}^{\tau^*} (y_i - \beta_{0,\tau,\tau^*} - \beta_{1,\tau,\tau^*} x_i - \beta_{2,\tau,\tau^*} x_i^2)^2, \\
\frac{\partial \mathrm{RSS}}{\partial \beta_{0,\tau,\tau^*}} &= \sum_{i=\tau+1}^{\tau^*} -2(y_i - \beta_{0,\tau,\tau^*} - \beta_{1,\tau,\tau^*} x_i - \beta_{1,\tau,\tau^*} x_i^2) = 0, \\
\frac{\partial \mathrm{RSS}}{\partial \beta_{1,\tau,\tau^*}} &= \sum_{i=\tau+1}^{\tau^*} -2x_i(y_i - \beta_{0,\tau,\tau^*} - \beta_{1,\tau,\tau^*} x_i - \beta_{2,\tau,\tau^*} x_i^2) = 0, \\
\frac{\partial \mathrm{RSS}}{\partial \beta_{2,\tau,\tau^*}} &= \sum_{i=\tau+1}^{\tau^*} -2x_i^2(y_i - \beta_{0,\tau,\tau^*} - \beta_{1,\tau,\tau^*} x_i - \beta_{2,\tau,\tau^*} x_i^2) = 0,
\end{aligned}$$

The parameter values that minimise the RSS, $\hat{\beta}_{0,\tau,\tau^*}$, $\hat{\beta}_{1,\tau,\tau^*}$ and $\hat{\beta}_{2,\tau,\tau^*}$ are the solutions to the equation system

$$\begin{pmatrix}
\tau^* - \tau & \sum_{i=\tau+1}^{\tau^*} x_i & \sum_{i=\tau+1}^{\tau^*} k x_i^2 \\
\sum_{i=\tau+1}^{\tau^*} x_i & \sum_{i=\tau+1}^{\tau^*} x_i^2 & \sum_{i=\tau+1}^{\tau^*} x_i^3 \\
\sum_{i=\tau+1}^{\tau^*} x_i^2 & \sum_{i=\tau+1}^{\tau^*} x_i^3 & \sum_{i=\tau+1}^{\tau^*} x_i^4
\end{pmatrix}
\begin{pmatrix}
\hat{\beta}_{0,\tau,\tau^*} \\ \hat{\beta}_{1,\tau,\tau^*} \\ \hat{\beta}_{2,\tau,\tau^*}
\end{pmatrix}
=
\begin{pmatrix}
\sum_{i=\tau+1}^{\tau^*} y_i \\ \sum_{i=\tau+1}^{\tau^*} y_i x_i \\ \sum_{i=\tau+1}^{\tau^*} y_i x_i^2
\end{pmatrix}.$$

The system of equations was solved for those τ, τ^* for which $\tau + 1 < \tau^* - 1$, i.e. so that there are at least three points to fit a linear function to. For less than three points, the cost function will equal zero. The fitted values $\hat{y}_i$ were calculated as

$$\hat{y}_i = \hat{\beta}_{0,\tau,\tau^*} + \hat{\beta}_{1,\tau,\tau^*} x_i + \hat{\beta}_{2,\tau,\tau^*} x_i^2, \quad \forall\ \tau < i \leq \tau^*. \tag{3.2}$$

The cost function becomes

$$C(x_{\tau+1:\tau^*}, y_{\tau+1:\tau^*}) = \begin{cases} \sum_{i=\tau+1}^{\tau^*}(y_i - \hat{y}_i)^2, & \tau + 1 < \tau^* - 1, \\ 0, & \text{otherwise.} \end{cases}$$

We call this cost function the *Quadratic RSS cost function*.

We will also model the data with a first order polynomial on each interval, assuming $\bar{\beta}_2 = 0$. Thus, for all $\tau^* = 1, 2, \ldots, n$ and for all $0 \leq \tau < \tau^*$ the RSS is minimised and the parameters $\hat{\beta}_{0,\tau,\tau^*}$ and $\hat{\beta}_{1,\tau,\tau^*}$ are calculated. Fixing τ and τ^*, the parameter values were found by minimising the RSS as described above. The parameters minimising the RSS, $\hat{\beta}_{0,\tau,\tau^*}$ and $\hat{\beta}_{1,\tau,\tau^*}$, are the solutions to the equation system

$$\begin{aligned} (\tau^* - \tau) \cdot \hat{\beta}_{0,\tau,\tau^*} + \hat{\beta}_{1,\tau,\tau^*} \sum_{i=\tau+1}^{\tau^*} x_i &= \sum_{i=\tau+1}^{\tau^*} y_i, \\ \hat{\beta}_{0,\tau,\tau^*} \sum_{i=\tau+1}^{\tau^*} x_i + \hat{\beta}_{1,\tau,\tau^*} \sum_{i=\tau+1}^{\tau^*} x_i^2 &= \sum_{i=\tau+1}^{\tau^*} y_i x_i, \end{aligned}$$

which is written on matrix form as

$$\begin{pmatrix} \tau^* - \tau & \sum_{i=\tau+1}^{\tau^*} x_i \\ \sum_{i=\tau+1}^{\tau^*} x_i & \sum_{i=\tau+1}^{\tau^*} x_i^2 \end{pmatrix} \begin{pmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \end{pmatrix} = \begin{pmatrix} \sum_{i=\tau+1}^{\tau^*} y_i \\ \sum_{i=\tau+1}^{\tau^*} y_i x_i \end{pmatrix}.$$

The fitted values $\hat{y}_i$ are calculated as

$$\hat{y}_i = \hat{\beta}_{0,\tau,\tau^*} + \hat{\beta}_{1,\tau,\tau^*} x_i, \quad \forall\ \tau < i \leq \tau^*, \tag{3.3}$$

and the cost function becomes

$$C(x_{\tau+1:\tau^*}, y_{\tau+1:\tau^*}) = \begin{cases} \sum_{i=\tau+1}^{\tau^*}(y_i - \hat{y}_i)^2, & \tau + 1 < \tau^* - 1, \\ 0, & \text{otherwise,} \end{cases}$$

We call this cost function the *linear RSS cost function*.

3.1.2 The likelihood cost functions

Also, the likelihood function can be used as cost function in the dynamical programming algorithm, see [16]. As in Section 3.1.1, we call the segments $\kappa_j = \{\tau_{j-1}, \tau_{j-1} + 1, \ldots, \tau_j\}$, for $j = 1, 2, \ldots, k + 1$. We assume that the data in each segment can be modelled by

$$y_i = \beta_{0,j} + \beta_{1,j} x_i + \epsilon_i,$$

for all x_i such that $i \in \kappa_j$, where ϵ_i is the error term, i.e. the deviance from the data point and the modelled regression line described by $\beta_{0,j}$ and $\beta_{1,j}$. We assume further that the

ϵ_i's are uncorrelated and identically distributed as $N(0, \sigma_j^2)$ for $i \in \kappa_j$, with support from the Central Limit Theorem [4, section 4.2]. Thus, y_i is assumed to be the outcome of a stochastic variable Y_i, with

$$Y_i \sim N(\beta_{0,j} + \beta_{1,j} x_i, \sigma_j^2),$$

for all x_i such that $i \in \kappa_j$. The probability density function of Y_i is

$$f_{Y_i}(y_i | x_i, \beta_{0,j}, \beta_{1,j}, \sigma_j^2) = \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{(y_i - \beta_{0,j} - \beta_{1,j} x_i)^2}{2\sigma_j^2}},$$

for all $i \in \kappa_j$. Assuming that the Y_i's are independent, the likelihood function on the data set $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, is the product of the probability density functions f_{Y_i},

$$\begin{aligned} L(\bar{\beta}_0, \bar{\beta}_1, \bar{\sigma}^2) &= f_{Y_1, Y_2, \ldots, Y_n}(y_1, y_2, \ldots, y_n | x_1, x_2, \ldots, x_n, \bar{\beta}_0, \bar{\beta}_1, \bar{\sigma}^2), \\ &= \prod_{j=1}^{k+1} \prod_{i \in \kappa_j} \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{(y_i - \beta_{0,j} - \beta_{1,j} x_i)^2}{2\sigma_j^2}}, \end{aligned}$$

also described in [1], where

$$\begin{aligned} \bar{\beta}_0 &= (\beta_{0,1}, \beta_{0,2}, \ldots, \beta_{0,k+1}), \\ \bar{\beta}_1 &= (\beta_{1,1}, \beta_{1,2}, \ldots, \beta_{1,k+1}), \\ \bar{\sigma} &= (\sigma_1, \sigma_2, \ldots, \sigma_{k+1}). \end{aligned}$$

Assuming that the number of observations in segment κ_j is m, the unbiased maximum likelihood (ML) estimates of the parameters are,

$$\begin{aligned} \hat{\beta}_{0,j} &= \bar{y}_i - \hat{\beta}_{1,j} \bar{x}_i, \\ \hat{\beta}_{1,j} &= \frac{\sum_{i \in \kappa_j} (x_i - \bar{x}_i)(y_i - \bar{y}_i)}{\sum_{i \in \kappa_j} (x_i - \bar{x}_i)^2}, \\ \hat{\sigma}_j^2 &= \frac{1}{m-2} \sum_{i \in \kappa_j} (y_i - \hat{\beta}_{0,j} - \hat{\beta}_{1,j} x_i)^2, \end{aligned}$$

where

$$\begin{aligned} \bar{x}_i &= \frac{1}{m} \sum_{i \in \kappa_j} x_i, \\ \bar{y}_i &= \frac{1}{m} \sum_{i \in \kappa_j} y_i, \end{aligned}$$

as described in [1]. Let us again translate this into the notation of equation (3.1). We fix τ and τ^* and let $m = \tau^* - \tau$. We use twice the negative likelihood function as cost

function, so that the costs for the different segments can be added to give the total cost for the model including the change points. Further, we have made the assumption that there must be at least five data points in each segment. Thus, if $\tau + 1 < \tau^* - 3$ the cost function for a specific segment starting on index $\tau + 1$ will be

$$\begin{aligned} -2\ln[L(\hat{\beta}_{0,\tau,\tau^*}, \hat{\beta}_{1,\tau,\tau^*}, \hat{\sigma}^2_{\tau,\tau^*})] &= -2\ln\left[\frac{1}{(2\pi\hat{\sigma}^2_{\tau,\tau^*})^{(\tau^*-\tau)/2}} e^{-\sum_{i=\tau+1}^{\tau^*} \frac{(y_i - \hat{\beta}_{0,\tau,\tau^*} - \hat{\beta}_{1,\tau,\tau^* x_i})^2}{2\hat{\sigma}^2_{\tau,\tau^*}}}\right], \\ &= (\tau^* - \tau)\ln[2\pi\hat{\sigma}^2_{\tau,\tau^*}] + 2\sum_{i=\tau+1}^{\tau^*} \frac{(y_i - \hat{\beta}_{0,\tau,\tau^*} - \hat{\beta}_{1,\tau,\tau^*} x_i)^2}{2\hat{\sigma}^2_{\tau,\tau^*}}. \end{aligned}$$

If $\tau + 1 \geq \tau^* - 3$ however, the cost function was set to infinity, which makes it impossible to have a segment with only two points. The cost function is thus

$$C(x_{\tau+1:\tau^*}, y_{\tau+1:\tau^*}) = \begin{cases} (\tau^* - \tau)\ln[2\pi\hat{\sigma}^2_{\tau,\tau^*}] + 2\sum_{i=\tau+1}^{\tau^*} \frac{(y_i - \hat{\beta}_{0,\tau,\tau^*} - \hat{\beta}_{1,\tau,\tau^* x_i})^2}{2\hat{\sigma}^2_{\tau,\tau^*}}, & \tau + 1 < \tau^* - 3, \\ \text{infinity}, & \text{otherwise}, \end{cases}$$

and we call it the *likelihood regression cost function.*

We also investigate the likelihood function setting $\bar{\beta}_1 = 0$. The likelihood function for the data set $(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)$, is in that case,

$$\begin{aligned} L(\bar{\beta}_0, \bar{\sigma}^2) &= f_{Y_1, Y_2, \ldots, Y_n}(y_1, y_2, \ldots, y_n | x_1, x_2, \ldots x_n, \bar{\beta}_0, \bar{\sigma}^2), \\ &= \prod_{j=1}^{k+1} \prod_{i \in \kappa_j} \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{(y_i - \beta_{0,j})^2}{2\sigma_j^2}}. \end{aligned}$$

The unbiased maximum likelihood estimates of the parameters, as described in [1], are

$$\begin{aligned} \hat{\beta}_{0,j} &= \bar{y}_j = \frac{1}{m}\sum_{i \in \kappa_j} y_i, \\ \hat{\sigma}_j^2 &= \frac{1}{m-1}\sum_{i \in \kappa_j} (y_i - \bar{y})^2. \end{aligned}$$

We fix τ^* and τ and let $m = \tau^* - \tau$. As previously, we use twice the negative likelihood function as cost function, with the assumption that there must be at least five data points in each segment. Thus, the cost function for a specific segment starting on index $\tau + 1$ is

$$C(x_{\tau+1:\tau^*}, y_{\tau+1:\tau^*}) = \begin{cases} (\tau^* - \tau)\ln[2\pi\hat{\sigma}^2_{\tau,\tau^*}] + 2\sum_{i=\tau+1}^{\tau^*} \frac{(y_i - \hat{\beta}_{0,\tau,\tau^*})^2}{2\hat{\sigma}^2_{\tau,\tau^*}}, & \tau + 1 < \tau^* - 3, \\ \text{infinity}, & \text{otherwise}. \end{cases}$$

We call this cost function the *likelihood mean cost function.*

3.2 MATLAB function *findchangepts()*

The MATLAB function *findchangepts()*, is based on the dynamical programming algorithm described in Section 3.1 and in [16, 17]. The function finds abrupt changes in time series, based on a statistics specified by the user. The statistics can either be *mean*, detecting changes in mean, *rms*, detecting changes in root-mean-square level, *std*, detecting changes in standard deviation or *linear*, detecting changes in linear trend. The user can further specify the maximum number of changes to be searched for or the minimum improvement of the residual that adding a change point results in. The user can also specify a minimum distance between the change points. The function is further described in the MATLAB documentation [18]. We have investigated the performance of the method keeping the linear trend constant in each segment. The function *findchangepts()* assumes equidistant time span between the observations and therefore cannot take information about missing data points into account.

3.3 DBEST

The DBEST algorithm (Detecting Breakpoints and Estimating Segments in Trend), is described in [10]. It is a segmentation method, which segments time series based on an iterative change point detection algorithm. Assume that we have n data points. First, so called *level shift points* are detected in the following way. The difference

$$d_i = y_{i+1} - y_i, \ i = 1, 2, \ldots, n-1,$$

is computed and compared to a user-specified value θ_1, called *first level-shift-threshold*. For points y_i for which $d_i > \theta_1$, it is investigated whether the difference has resulted in a significant change in mean. The mean over a time period of length ϕ, called the *duration-threshold*, before the point y_i is compared to the mean over a time period of length ϕ after the point y_i. If the difference in mean before and after y_i exceeds a value θ_2, called the *second level-shift-threshold*, the point y_i is regarded as a candidate level shift point. The candidate level shift points are sorted in descending order of absolute value of mean difference. If the time distance between the candidate level shift point y_i and the previous candidate level shift point is at least ϕ, y_i is called a level shift point. The data set is separated at the level shift points and divided into segments. If there are any seasonal components, the different segments are deseasonalised separately, using STL decomposition [10].

Secondly, the *peak/valley function*, f, is defined as,

$$\begin{aligned} f_i &= \frac{|\text{sign}(y_{i+1} - y_i) - \text{sign}(y_i - y_{i-1})|}{2} \quad i = 2, 3, \ldots, n-1, \\ f_0 &= 1, \\ f_n &= 0. \end{aligned}$$

The points for which the peak/value function equals one are called *peak* and *valley* points depending on whether $\text{sign}(y_{i+1} - y_i) - \text{sign}(y_i - y_{i-1})$ equals -2 or 0. Between all successive pairs of peak and valley points, let us call them p_j and p_{j-1}, a straight line is fitted. The distance between the line and each data point that lie between p_j and p_{j-1} is computed. This is done for all points in the time series and for each pair of peak/ valley points, the intermediate point with the greatest distance to the line is compared to a user defined value ϵ. DBEST can also calculate a default value on ϵ as,

$$\epsilon = 3 \cdot R^*,$$

where "R^* is the root mean square error for the fit obtained using the peak and valley points plus the time series' start point and the level-shift points" [10, p. 186].

A second function called the *turning point function*, g, is introduced with g_i equal to one for the points for which the distance is greater than ϵ or if f_i equals one. Also, g_i equals one for the *level shift points*. Otherwise g_i equals zero. DBEST calculates g_i for $i = 1, 2, \ldots, n$. The points for which $g_i = 1$ are called *turning points*, and we denote them as t_j. Hence, assuming that we have m turning points, $g_{t_j} = 1$ for $t_1, t_2, \ldots, t_m$ and $g_i = 0$ if i is not equal to t_j for any j. The same procedure is done with the turning points, i.e. a straight line is fitted to each pair of turning points and the distances from the intermediate data points to the line are calculated and compared to ϵ. When g does not change anymore, all turning points t_j are found and the iteration is finished [10].

Lastly, the *trend local change function*, h, is calculated and h_i equals zero if g_i equals zero and $t_{j+1} - t_j$ otherwise. We assume that we have m turning points with $1 < t_1 < \ldots < t_m < t_{m+1} = n$ [10]. Then,

$$h_i = \begin{cases} t_{j+1} - t_j, & \text{if } i = t_j, \text{for some } j, \\ 0, & \text{otherwise.} \end{cases}$$

The turning points are sorted in descending order of h. Least-squares fits are calculated, first for the entire time series without change points, secondly adding to the model the first turning point, i.e. the turning point for which h has its maximum value. Third, the second turning point is added to the model, etc. The BIC (Bayesian Information Criterion) is calculated for each fit. The user specifies θ_1, θ_2 and ϕ and ϵ can either be specified by the user or computed by DBEST. The user also gives information on whether the data is cyclical or non-cyclical, i.e. if the data has a seasonal component and specify in what way the change points are chosen. Either, the number of change points (here called turning points), the so called generalisation threshold value or the so called change magnitude threshold value must be specified. The user can also specify the significance level α of the found change points.

3.4 TIMESAT

TIMESAT is a computer program for analysing time series of satellite data, described in [5, 13, 14]. TIMESAT primarily investigates seasonal components, such as start, end, duration and amplitude of seasons [14], using three different least-squares methods. Savitzky-Golay filter use "local polynomial functions" [14, p. 834] to approximate the time series. The second least-squares method used in TIMESAT, fits the data to a sum of polynomial and harmonic functions. The third method, is to approximate the time series with Gaussian functions [14]. Analysing sequential data, i.e. time series, with TIMESAT requires equally spaced data, i.e. the same time interval between each pair of successive points. Also it requires several data values per year.

Part I

Selection of relevant methods

In this part, we evaluate the different methods presented on eight different time series. The methods that are investigated are the optimal partitioning algorithm with four different cost functions, the MATLAB function *findchangepts()*, DBEST and TIMESAT. We discuss the result and select some of the methods for further analysis. First, we evaluate the methods on a typical data set describing a lake drainage event in the Arctic hemisphere. The data set contains six different time series, one for each index discussed in Chapter 2. In this chapter we will also further go through how the methods are implemented. The implementation of the methods holds throughout the study. Secondly, we evaluate the methods on two simulated data sets. The data sets are simulated to capture some of the typical problems in analysis of remote sensing time series, such as missing data points.

Chapter 4

Example lake drainage data

In this chapter, the seven methods presented are evaluated on the data in Figure 4.1. We present figures with the change points that have been found for each method. We show one change point for the indices TCG, NDMI, NDVI and NDWI and two change points for the indices TCW and TCB. The number of change points for the different indices have been chosen by looking at the data in Figure 4.1, since we do not have any reference change points to compare with. We find two obvious change points for TCW and TCB and one obvious change point for the other indices. The MATLAB function *findchangepts()*, as well as DBEST and TIMESAT take equally spaced data as input, i.e. data for which the time span between each pair of successive data points is equal throughout the time series. Therefore, an equally spaced data set was created, taking one value each year from each time series in Figure 4.1. The reduced data set is shown in Figure 4.2. The original data set in Figure 4.1, contained gap years in the beginning of the measurement period, i.e. years where no observations were made. These years were removed when creating the reduced data set.

We discuss how well the methods generally seem to apply to satellite-based remote sensing time series of lake drainage data that are mentioned in Chapter 2, finding that TIMESAT, DBEST and the MATLAB function *findchangepts()* are not well suited for our purposes of finding change points in satellite-based remote sensing lake drainage data.

4.1 Data

The example data set is shown below.

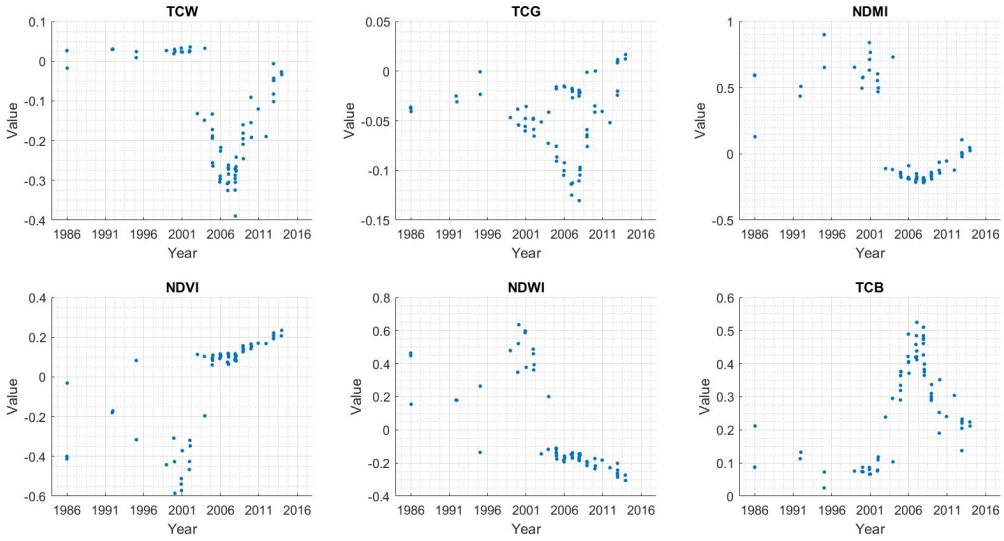

Figure 4.1: Time series from the Landsat archive describing the event of lake drainage.

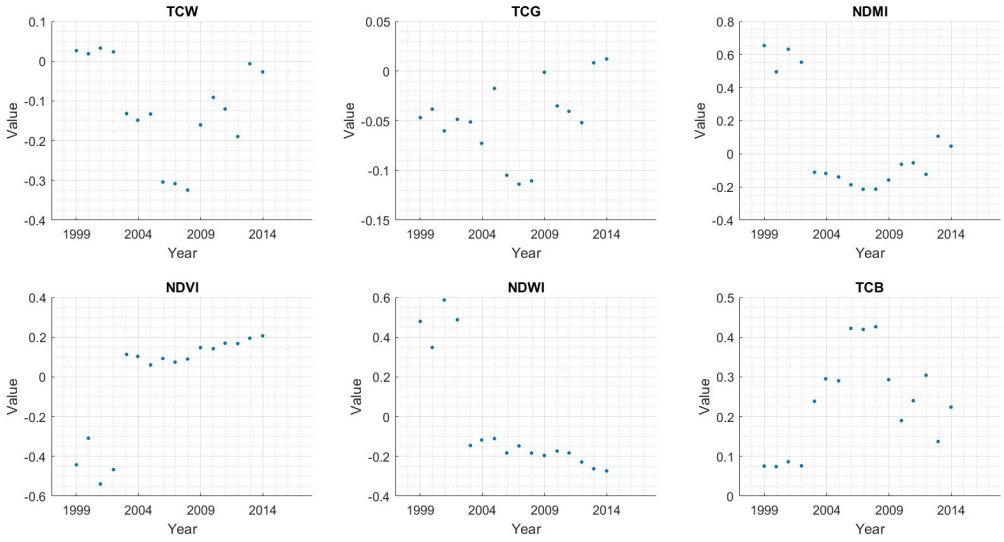

Figure 4.2: Equally spaced lake drainage data. One observation per year taken from the time series in Figure 4.1.

4.2 Results

4.2.1 Optimal partitioning

Change points were searched for, using the optimal partitioning algorithm described in equation (3.1). We used the four cost functions presented in Chapter 3.1; the linear RSS cost function, the quadratic RSS cost function, the likelihood regression cost function and the likelihood mean cost function. The optimal partitioning algorithm takes a value of the penalty constant γ as input and the output is the positions of the change points that have been found. The penalty constant γ was manually varied to find two change points each for the indices TCW and TCB and one change point for the indices TCG, NDMI, NDVI and NDWI.

The algorithm can take into account both the time and the value, i.e. the two dimensional vector (x_i, y_i) from the data set $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$. Therefore, the method accurately handles unequally spaced data, or time series with missing observations.

The linear RSS

First, change points were searched for using the linear RSS cost function and the result is presented in the figures below.

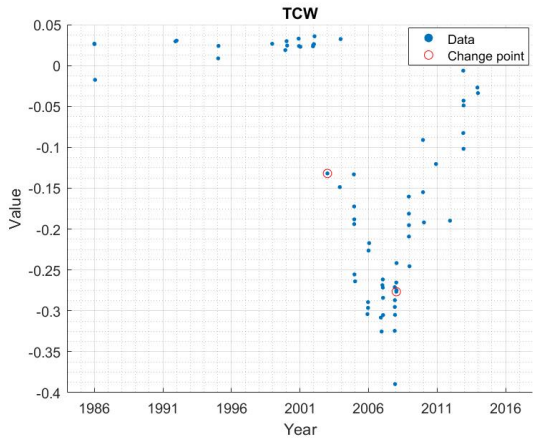

Figure 4.3: Data of TCW segmented with linear RSS.

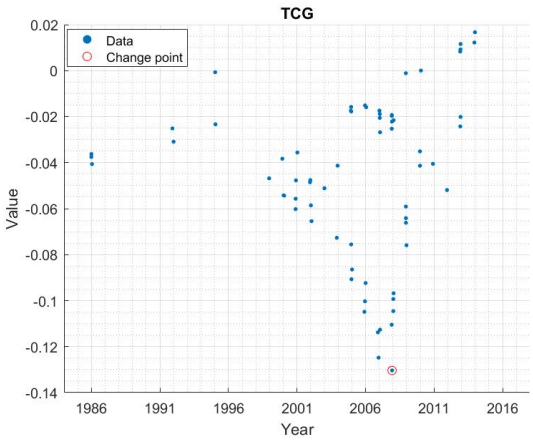

Figure 4.4: Data of TCG segmented with linear RSS.

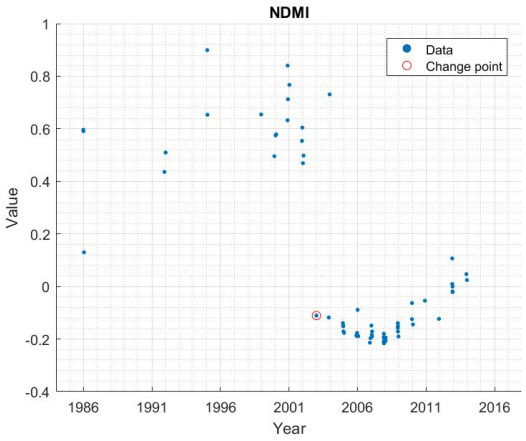

Figure 4.5: Data of NDMI segmented with linear RSS.

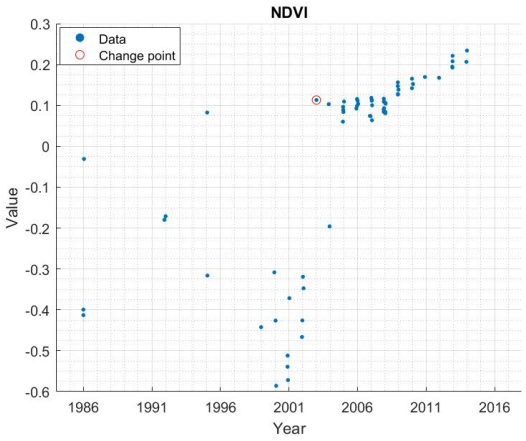

Figure 4.6: Data of NDVI segmented with linear RSS.

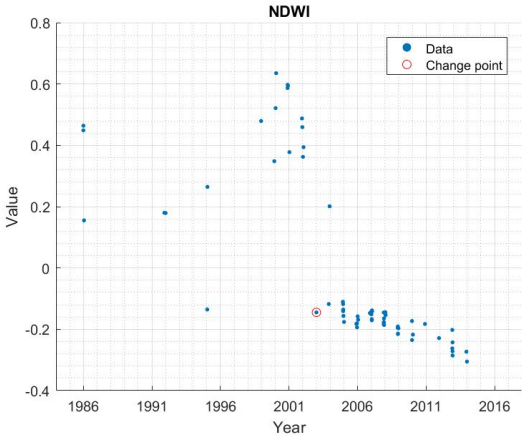

Figure 4.7: Data of NDWI segmented with linear RSS.

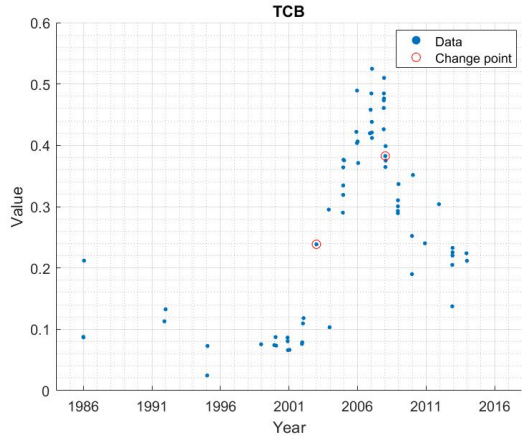

Figure 4.8: Data of TCB segmented with linear RSS.

The quadratic RSS

Secondly, the quadratic RSS cost function was used and the results are shown below.

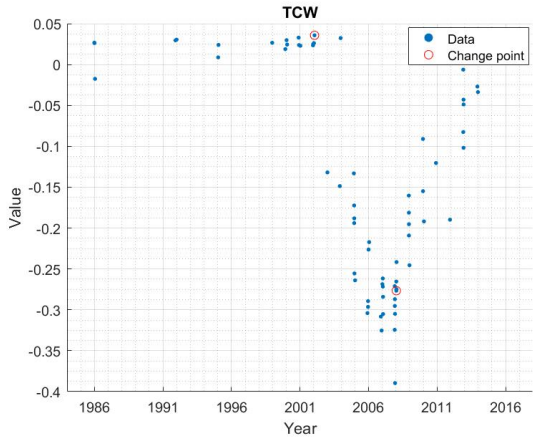

Figure 4.9: Data of TCW segmented with quadratic RSS.

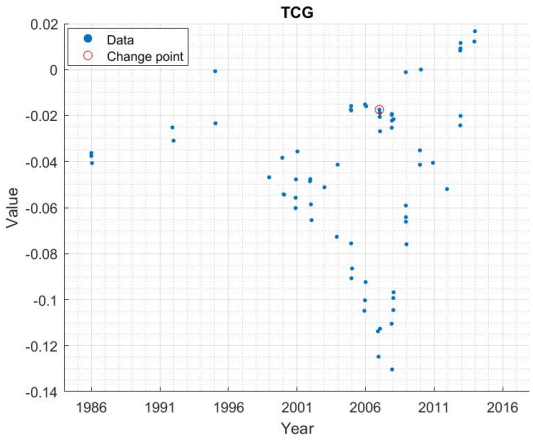

Figure 4.10: Data of TCG segmented with quadratic RSS.

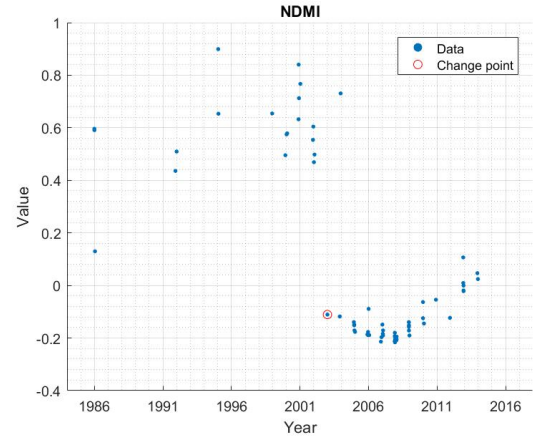

Figure 4.11: Data of NDMI segmented with quadratic RSS.

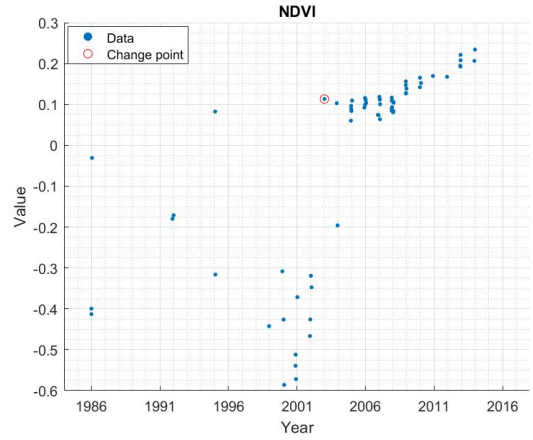

Figure 4.12: Data of NDVI segmented with quadratic RSS.

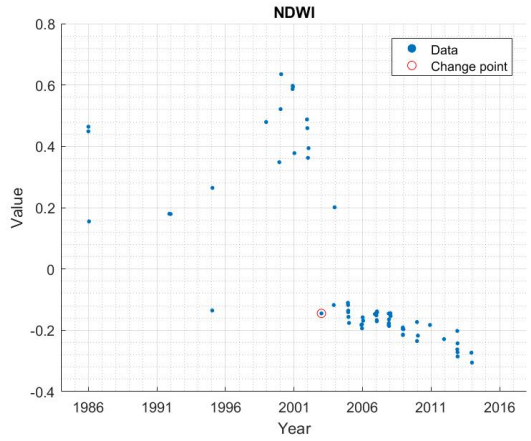

Figure 4.13: Data of NDWI segmented with quadratic RSS.

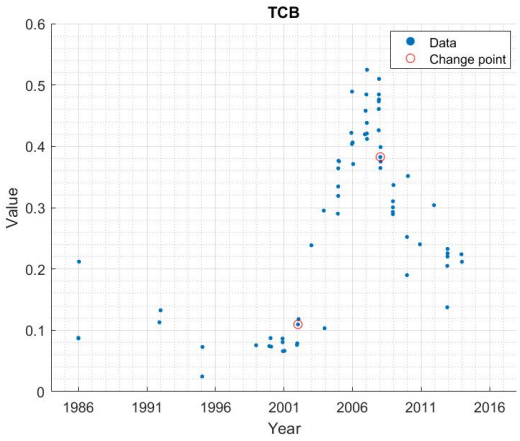

Figure 4.14: Data of TCB segmented quadratic RSS.

The likelihood regression function

Now, the likelihood regression cost function was used in the dynamic programming algorithm. The result of assuming normally distributed data and changing variance is shown in the figures below.

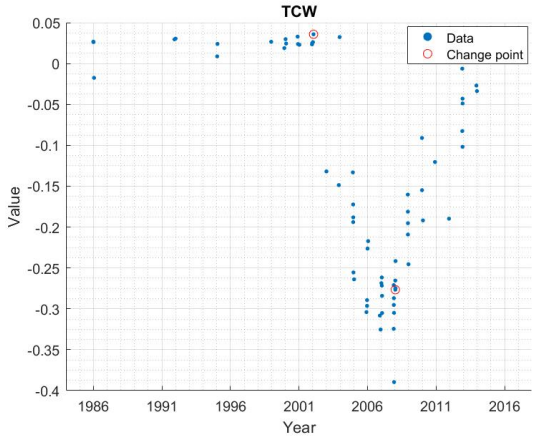

Figure 4.15: Data of TCW segmented with the likelihood regression function.

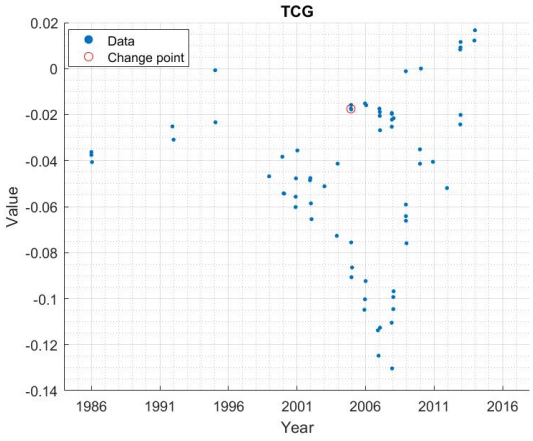

Figure 4.16: Data of TCG segmented with the likelihood regression function.

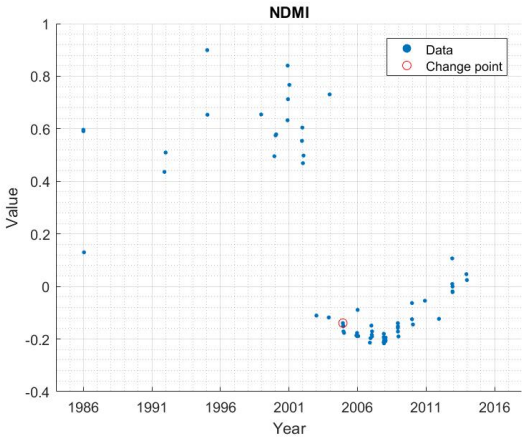

Figure 4.17: Data of NDMI segmented with the likelihood regression function.

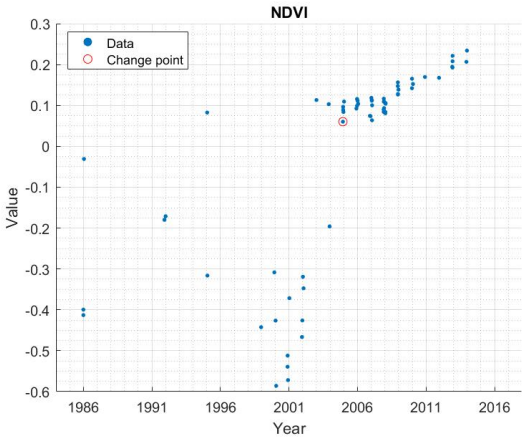

Figure 4.18: Data of NDVI segmented with the likelihood regression function.

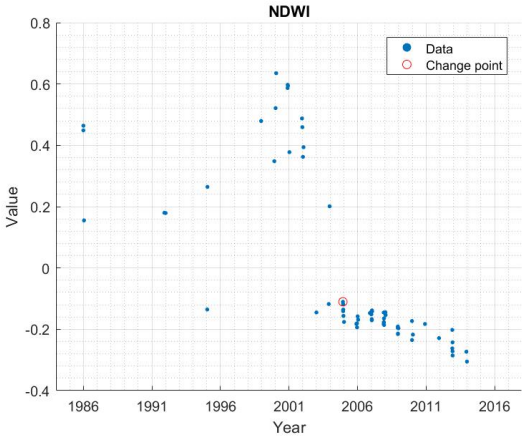

Figure 4.19: Data of NDWI segmented with the likelihood regression function.

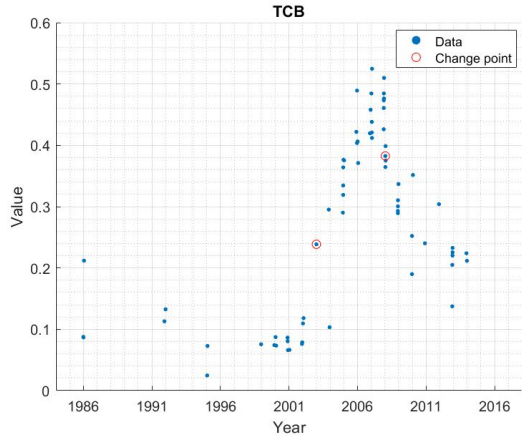

Figure 4.20: Data of TCB segmented with the likelihood regression function.

The likelihood mean function

Last, the likelihood mean function was used as cost function, see the result in the figures below.

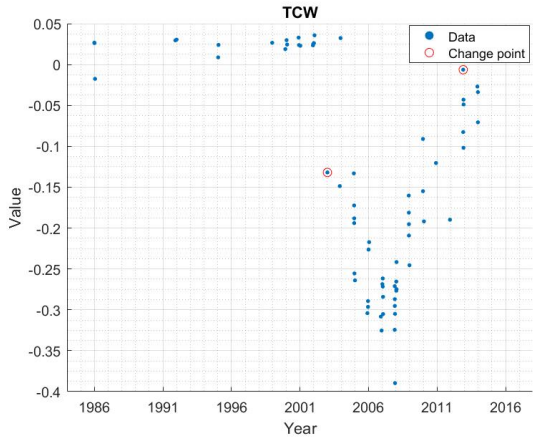

Figure 4.21: Data of TCW segmented with the log likelihood mean function.

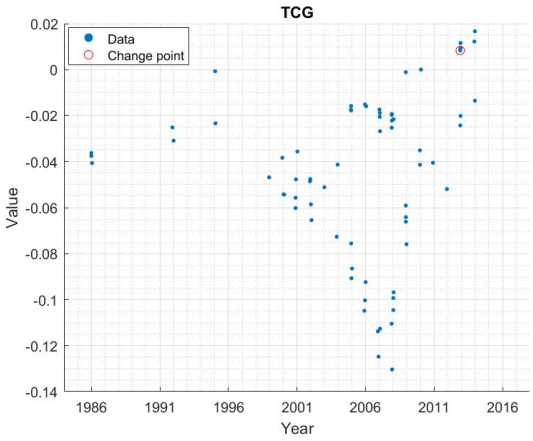

Figure 4.22: Data of TCG segmented with the likelihood mean function.

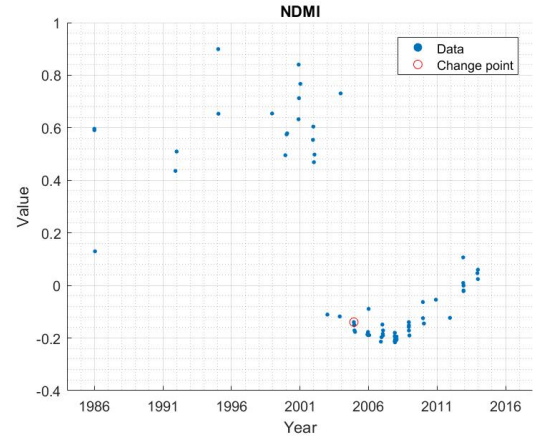

Figure 4.23: Data of NDMI segmented with the likelihood mean function.

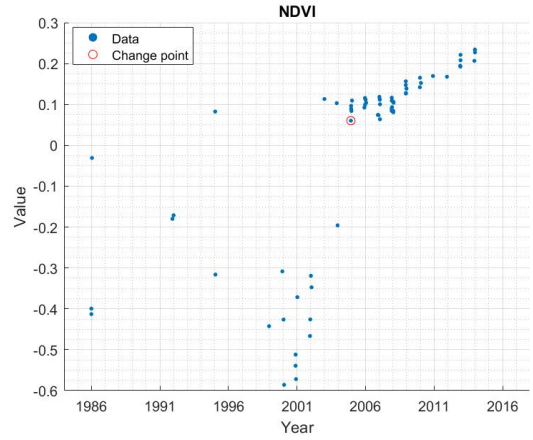

Figure 4.24: Data of NDVI segmented with the likelihood mean function.

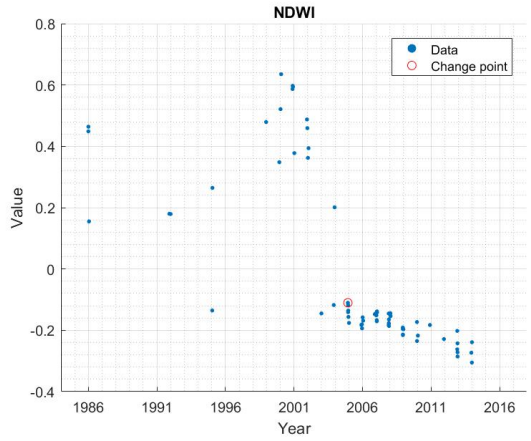

Figure 4.25: Data of NDWI segmented with the likelihood mean function.

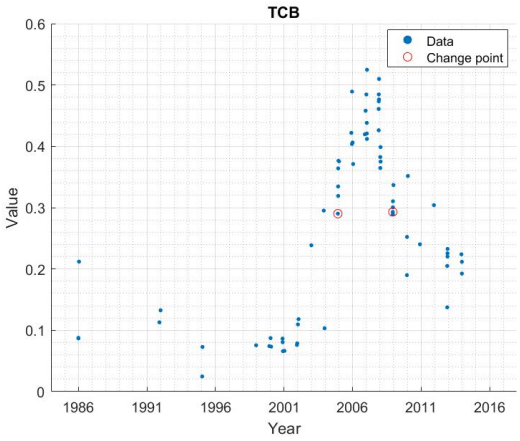

Figure 4.26: Data of TCB segmented with the likelihood mean function.

4.2.2 MATLAB function *findchangepts()*

We specified a maximum number of change points to be searched for with *findchangepts()*. This was two for TCW and TCB and one for the other indices. The result is shown in the figures below.

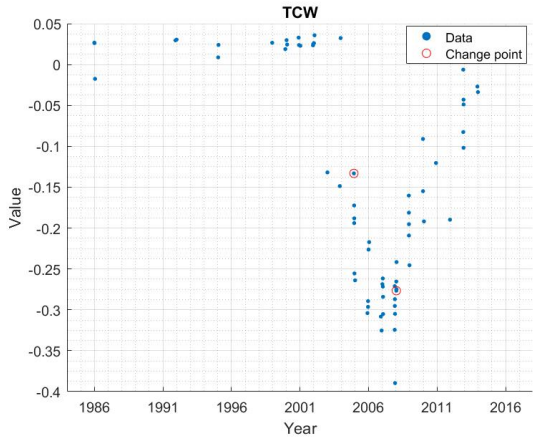

Figure 4.27: Data of TCW segmented with *findchangepts()*.

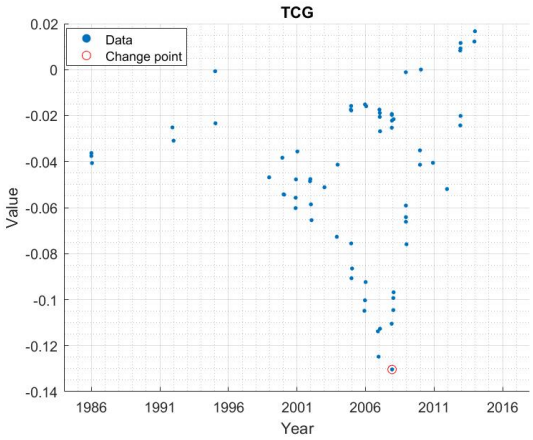

Figure 4.28: Data of TCG segmented with *findchangepts()*.

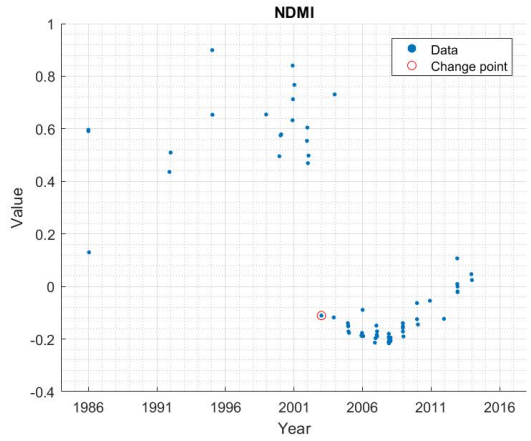

Figure 4.29: Data of NDMI segmented with *findchangepts()*.

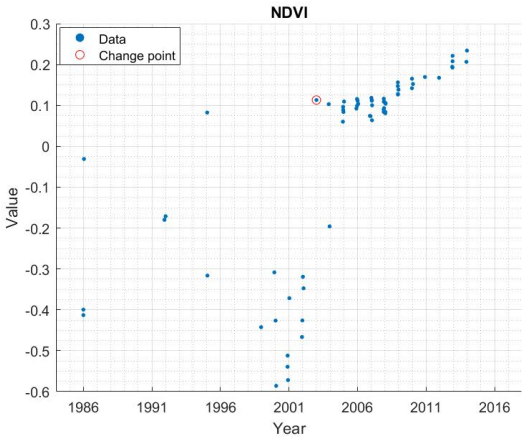

Figure 4.30: Data of NDVI segmented with *findchangepts()*.

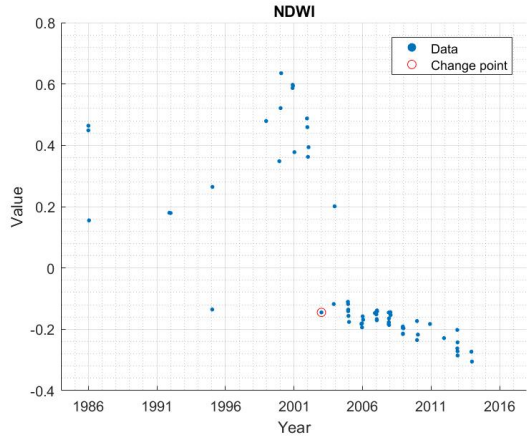

Figure 4.31: Data of NDWI segmented with *findchangepts()*.

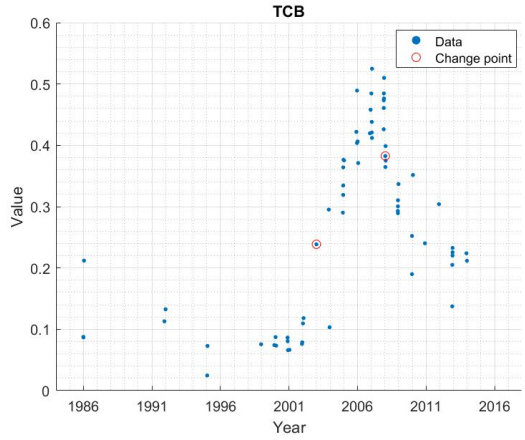

Figure 4.32: Data of TCB segmented with *findchangepts()*.

4.2.3 DBEST

DBEST, developed and described in [10], detects the start point and the end point of a change, and we have chosen only to plot one of the points that have been found for each change. Where the duration time was one point, for example if the start of the change was point 30 and the end of the change was point 31, we plotted the second of the two points. All changes had a duration time of one point, except for the change in NDWI and one of the changes in TCB, for which the duration was two points. For these changes we plotted the middle point. For all six indices, DBEST was run with $\theta_1 = 0.1$, $\theta_2 = 0.2$, $\phi = 20$ and default value on ϵ, recall Section 3.3. We also included two figures showing the output of DBEST for the original TCB data and the evenly spaced TCB data, Figures 4.40 and 4.39

respectively. These figures show the data points, plotted with straight lines in between. Also, the magnitude of the *trend local change function*, h, is shown in the histograms.

DBEST seems to be able to handle some missing data points but not too many. We tried using DBEST on the unevenly spaced data showing TCB (see Figure 4.1), adding *NaN* to the time series where there were missing data points, but the program returned an error message. No change points could be found.

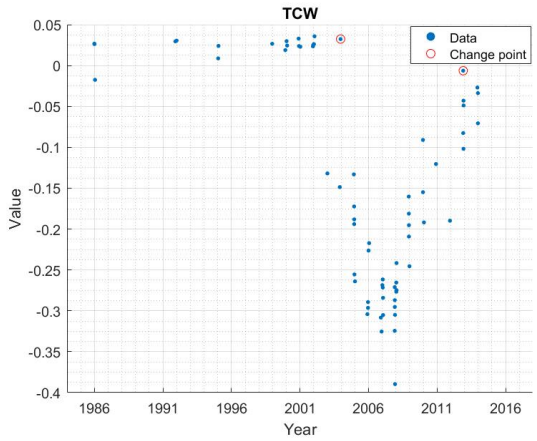

Figure 4.33: Data of TCW segmented with DBEST.

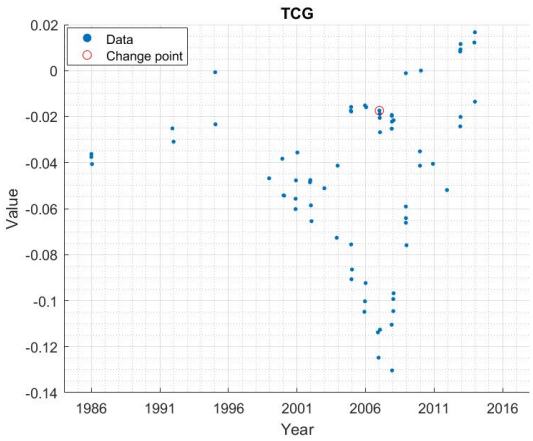

Figure 4.34: Data of TCG segmented with DBEST.

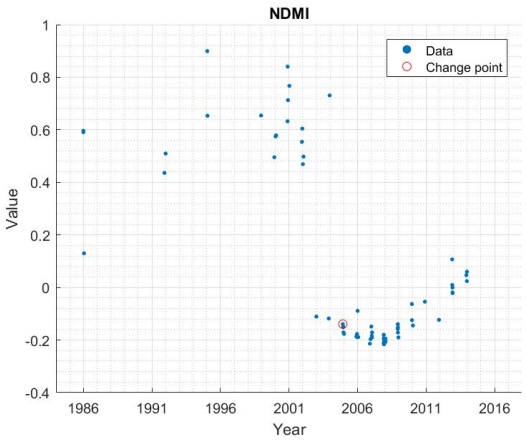

Figure 4.35: Data of NDMI segmented with DBEST.

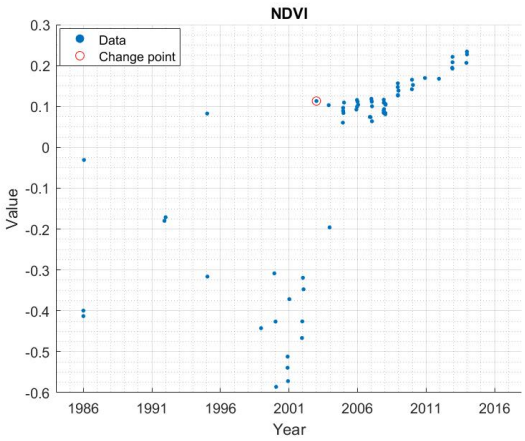

Figure 4.36: Data of NDVI segmented with DBEST.

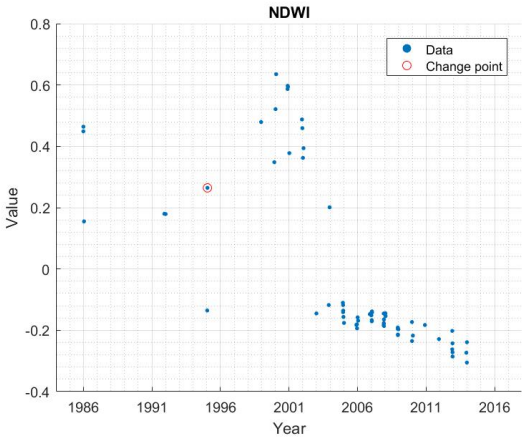

Figure 4.37: Data of NDWI segmented with DBEST.

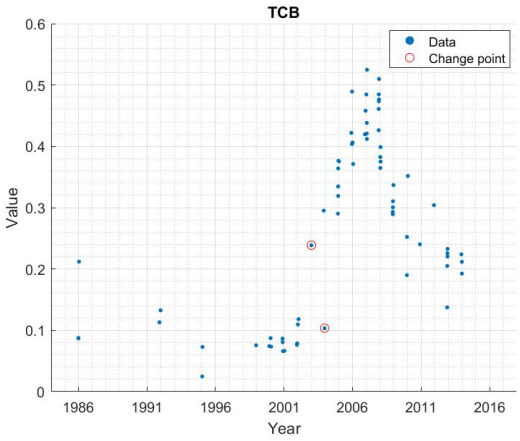

Figure 4.38: Data of TCB segmented with DBEST.

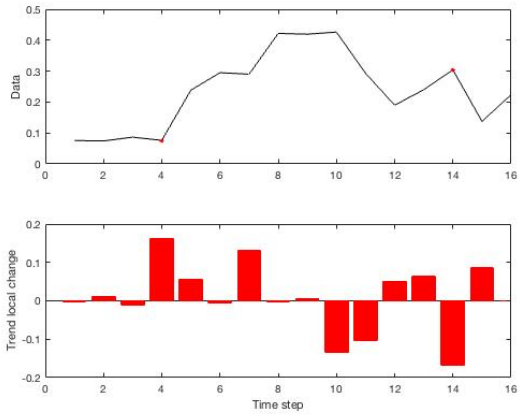

Figure 4.39: Data of equally spaced TCB, analysed with DBEST.

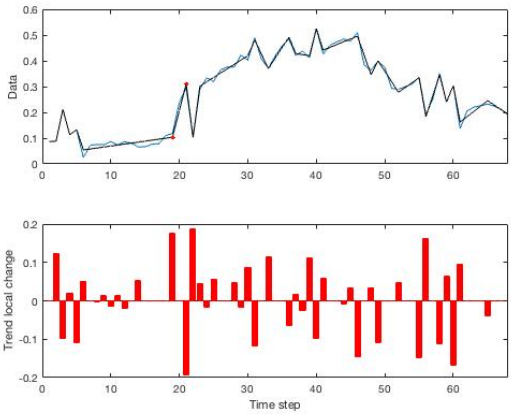

Figure 4.40: Data of unequally spaced TCB, analysed with DBEST

4.2.4 TIMESAT

TIMESAT was first run with the original data set in Figure 4.1 as input. However, the program returned an error message. There were too many data points missing. TIMESAT can deal with a few missing data points but in principle requires equally spaced data [5]. Analysing time series, TIMESAT also requires the user to specify the time interval in years during which the data has been gathered, as well as the number of data points each year, as described in [5, p. 10]. TIMESAT was therefore instead evaluated on the equally spaced data shown in Figure 4.2. However, TIMESAT could not handle the information that our data sometimes included only one data point each year. Therefore, we had to modify the information given to the program. We erroneously told the program that the data

set spanned over two years and that we had eight data points each year. TIMESAT was evaluated on the equally spaced TCB data shown in Figure 4.2. Three different models were fitted to the data. The results are shown in the figure below.

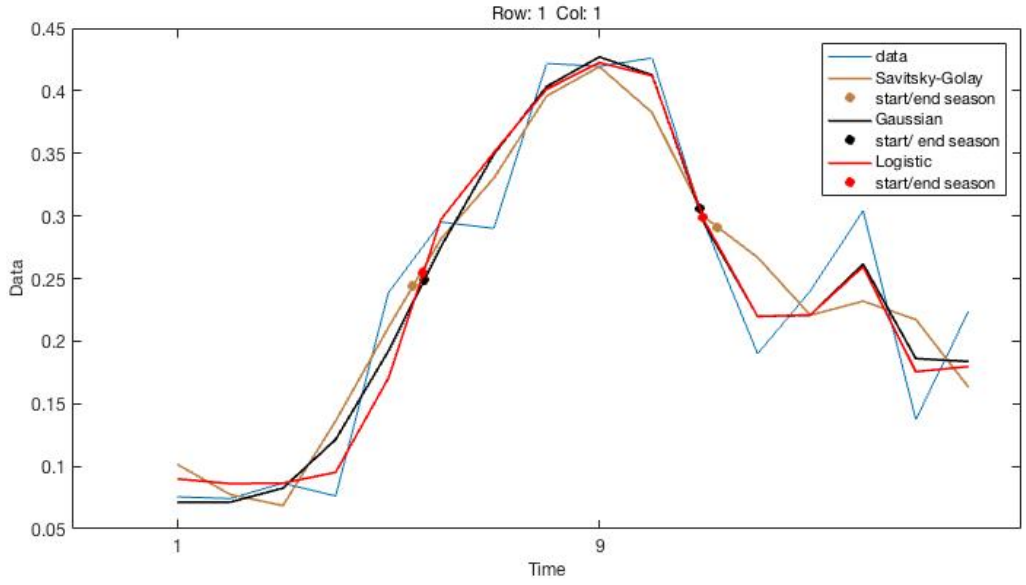

Figure 4.41: Data of equally spaced TCB analysed with TIMESAT and smoothed with a Savitsky-Golay filter, gaussian functions and logistic regression respectively. Start and end of potential seasons are shown.

4.3 Discussion

We do not have any reference change points to compare the results with. However the time series for NDMI, NDVI, and NDWI have at least one apparent change point, TCW and TCB has at least two, as discussed earlier. For TCG it is a little harder to distinguish any change point, although we can see that we first have a descending trend and later an increasing trend. We evaluate the methods based on the change points found by the eye.

Both TIMESAT and DBEST have trouble analysing unequally spaced data. The primary aim of TIMESAT is to analyse seasonal patterns and not structural changes. Although the data that we are working with might have seasonal components, as described in [20], TIMESAT is not suitable for our purposes. We are asked to specify the number of observations each year and the data sets that we are looking at in this study both contain gap years and generally not the same amount of observations each year. We had to modify the information about our data to make it possible to analyse with TIMESAT. DBEST on the other hand, can handle information about a few missing observations but not the total amount of missing observations in the data shown in Figure 4.1. We analysed the

time series in Figure 4.1 with DBEST, assuming equidistant time step between the data points. For the indices NDMI and NDVI, DBEST seems to find the accurate change points. The time series of NDWI has quite fluctuating/ noisy data approximately until year 2002. This seems to affect DBEST, which has found one of the points where the data is most fluctuating. Searching for two change points in the TCW time series, we notice that the first point seems accurate while the second change point does not. It is difficult to draw any conclusions with the TCG time series since it is very fluctuating. For the TCB time series, the two change points seem to have been found very close to each other. One of the change points seems accurate but DBEST misses to detect a second change that, from observing Figure 4.1, should be between year 2006 and 2011. Looking at the output of DBEST, Figures 4.39 and 4.40, we see that the *trend local change function*, h, has its maximum values at the found change points and the value h_i for point y_i seems to be highly dependent on the points y_{i-1} and y_{i+1}, compare to the theory describing DBEST in Section 3.3. The conclusion is that DBEST seem to be very sensitive to fluctuations and noisy data.

Table 4.1 shows the change points found with each method. The MATLAB function *findchangepts()* with statistic specified as *linear*, and the optimal partitioning algorithm with the different cost functions find similar changes. However it is interesting to note that in the time series of NDMI, NDVI and NDWI, the likelihood cost functions find point 23, whereas the RSS cost functions as well as *findchangepts()* find point 20. The main difference between the likelihood cost functions, the RSS cost functions and the MATLAB function *findchangepts()* lie in the time series of TCG, where only linear RSS and *findchangepts()* find the same point.

Method	TCW	TCG	NDMI	NDVI	NDWI	TCB
Linear RSS	20, 47	46	20	20	20	20, 47
Quadratic RSS	19, 47	36	20	20	20	18, 47
Likelihood regression	20, 47	23	23	23	23	19, 47
Likelihood mean	20, 61	61	23	23	23	23, 51
MATLAB function *findchangepts()*	23, 47	46	20	20	20	20, 47
DBEST	22, 61	36	23	20	7	21, 22

Table 4.1: Change points found with different methods in the data shown in Figure 4.1.

The conclusion from the first evaluation of the methods is that TIMESAT is not suitable for our purposes and is thus left out from further evaluation. Moreover, DBEST seems to

be affected by noisy data and/ or outliers and will be further evaluated together with the other methods, on two simulated data sets.

Chapter 5

First constructed data set

We do an evaluation of the remaining methods on a simulated data set. The methods are the optimal partitioning algorithm with cost functions linear RSS, quadratic RSS, the likelihood regression function and the likelihood mean function, the MATLAB function *findchangepts* and DBEST. The aim is to create a very simple data set, with an obvious change point, and investigate whether the methods find the accurate point. The idea was to create two different data sets with the same mean, the first of which with no slope but a relatively high variance and the second of which having a very small variance but an apparent slope. For the optimal partitioning algorithm, γ was chosen to find one change point and DBEST and *findchangepts()* were also told to find one change point.

5.1 Simulated data

The constructed data consists of 93 points with $x_1 = 1, x_2 = 2, \ldots, x_{93} = 93$. The first 60 points, $y_1, y_2, \ldots, y_{60}$, were produced by randomising 60 data points from a N(0, 0.15) distribution. After that, a series with 33 data points was created,

$$y_i = -0.2 + 0.0125(x_i - 61) + z_i,$$

for $i = 61, 62, \ldots, 93$, where z_i were randomised from N(0, 0.001). Thus, the first 60 points as well as the last 33 points, come from a normal distribution with mean zero, although the variances differ.

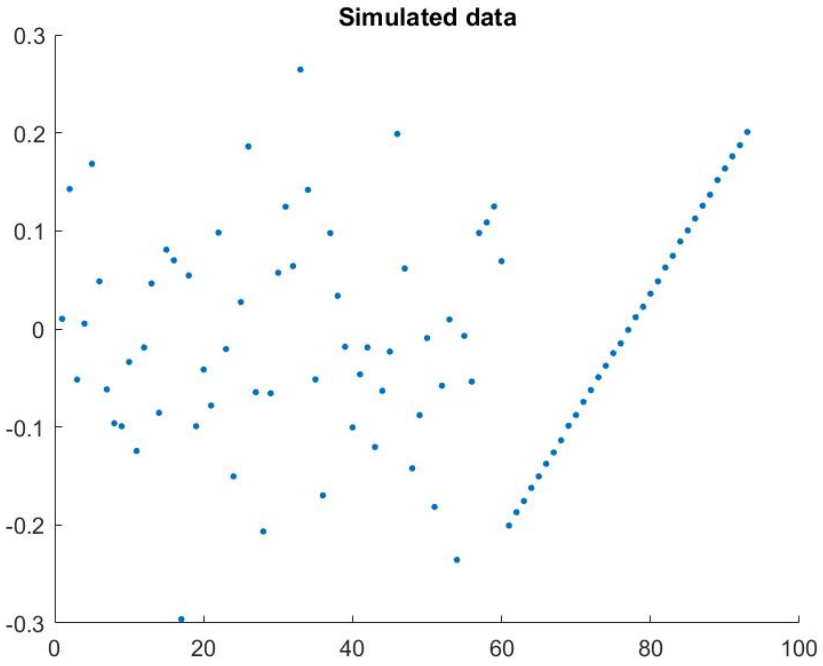

Figure 5.1: Constructed data generated from N(0, 0.15) and $y = -0.2 + 0.0125(x - 61) + z$.

5.2 Results

DBEST was run with $\theta_1 = 0.1$, $\theta_2 = 0.2$, $\phi = 10$, default value on ϵ and confidence level $\alpha = 5\%$. The result was the same for other values of ϕ, for example 24 and 33. DBEST found point 33 to be the start of the change and point 36 to be the end of the change. Point 33 is shown in Figure 5.7. The dynamical programming algorithm was run with penalty constants γ to find one change point. The penalty constants are presented with each figure respectively. The MATLAB function *findchangepts()* was also told to find one change point. The optimal partitioning algorithm with the likelihood mean function as cost function found point 83. The other cost functions as well as the MATLAB function *findchangepts()* found point 61.

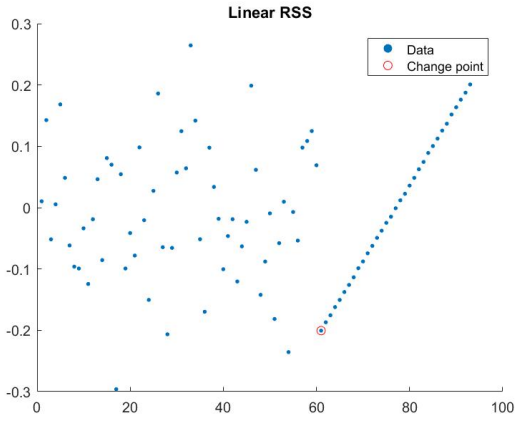

Figure 5.2: Optimal partitioning with cost function linear RSS used on the data in Figure 5.1 and γ=0.2.

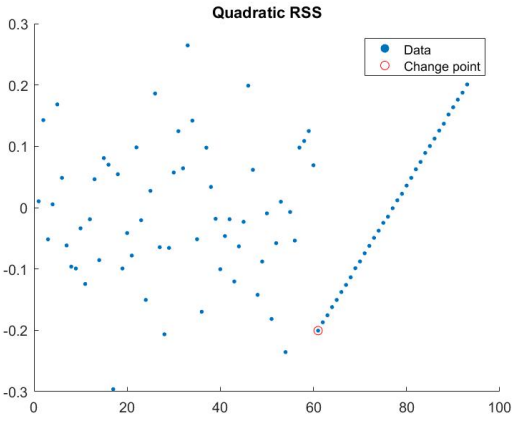

Figure 5.3: Optimal partitioning with cost function quadratic RSS used on the data in Figure 5.1 and γ=0.2.

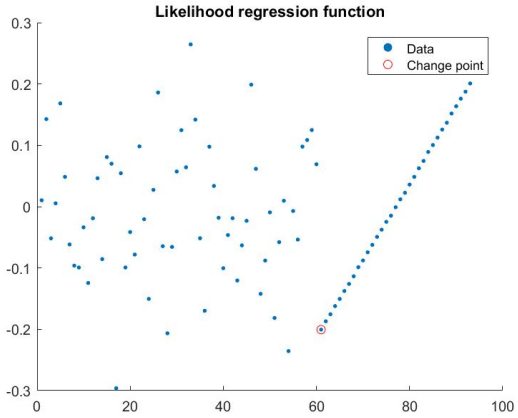

Figure 5.4: Optimal partitioning with the likelihood regression cost function used on the data in Figure 5.1 and $\gamma = 40$.

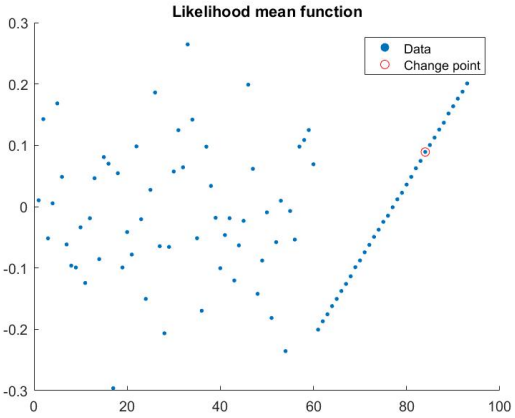

Figure 5.5: Optimal partitioning with the likelihood mean cost function used on the data in Figure 5.1 and $\gamma = 10$.

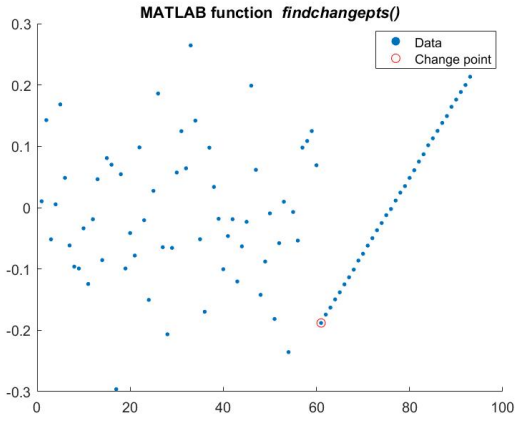

Figure 5.6: The MATLAB function *findchangepts()* used on the data in Figure 5.1.

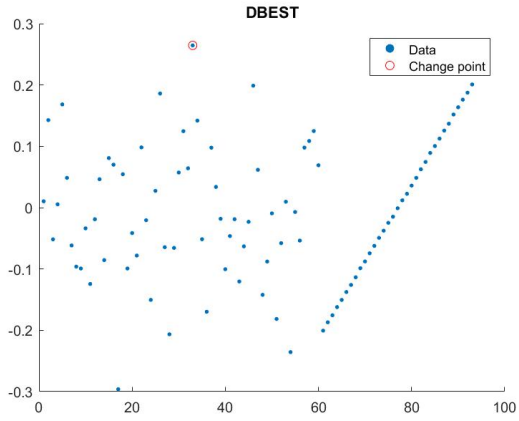

Figure 5.7: DBEST used on the data in Figure 5.1.

5.3 Discussion

The optimal partitioning algorithm found the correct change point for the cost functions linear RSS, quadratic RSS and the likelihood regression function. On the other hand, the algorithm did not find the correct change point for the likelihood mean function. DBEST has also not found the correct change point. As mentioned before, the MATLAB function *findchangepts()* is based on the same dynamical programming approach as described in Section 3.1. Since the simulated data (Figure 5.1) has equidistant time step between each point, it is reasonable that the result is similar to the optimal partitioning algorithm with cost function linear RSS and the likelihood regression function. Further, let us discuss why DBEST did not find the true change point. When finding the *peak/valley* points p_i and the *turning* points t_i, the method only takes into account preceding and succeeding values, which might make it possible for outliers or fluctuations to have great influence on the result. The reason why the correct change point was not found in this example, is that the jump from the preceding point to the correct change point is not big enough to be detected as a *peak/valley point* or a *turning point*, at least not with the parameter values that we specified. Furthermore, the mean of the first 60 points is approximately the same as the mean of the last 33 points, making it difficult to spot the point as a *level shift point*. Recall, in Chapter 4, we have also concluded that fluctuations for which the *trend local change function* is relatively great, can be regarded as change points. This is what has happened in this evaluation, see Figure 5.8. Hence, we conclude that DBEST is not suited for our purposes.

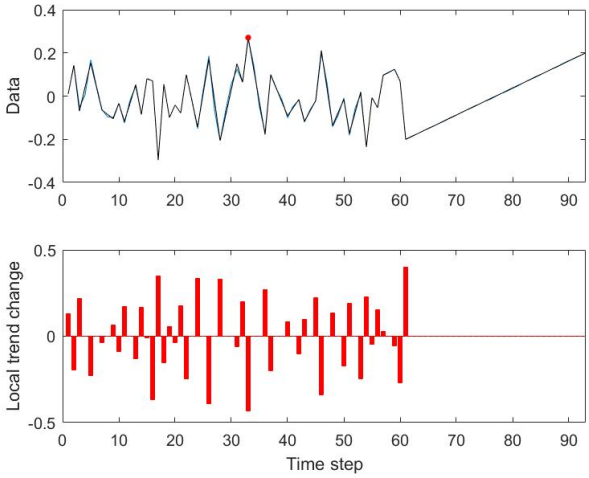

Figure 5.8: Output from DBEST when fitted to the simulated data in Figure 5.1.

Chapter 6

Second constructed data set

We simulated another data set in order to compare how DBEST, the MATLAB function *findchangepts()* and the optimal partitioning algorithm with the different cost functions perform. The idea was to create a data set with apparent missing values and investigate how well the methods handle it. As shown in Figure 6.1, the first 51 points, up to $x_{51} = 80$, have an apparent linear pattern with 29 points missing. The last 40 points have a linear pattern with no slope. The obvious change point is point 52. All methods were set to find one change point.

6.1 Simulated data

First, we created a data set $\bar{x}$ with

$$x_i = \begin{cases} i & i = 1, 2, \ldots, 30, \\ i + 29 & i = 31, 32, \ldots, 91. \end{cases}$$

Now, 51 data points $z_1, z_2, \ldots, z_{51}$, were randomised from $Z_1 \sim N(0, 1.5)$. The first 51 points in the data set were then calculated as

$$y_i = -30 + x_i + z_i,$$

for $i = 1, 2, \ldots, 51$. We call this data set $\bar{y}_1$. Secondly, 40 data points, $z_{52}, z_{53}, \ldots, z_{91}$ were randomised from $Z_2 \sim N(0, 1.5)$ and the 40 last data points were calculated as

$$y_i = 50 + z_i,$$

for $i = 52, 53, \ldots, 91$, where 50 would be the last value in the series $\bar{y}_1$, provided that the variance was zero. Both data set $\bar{y}_1$ and $\bar{y}_2$ have variance 1.5.

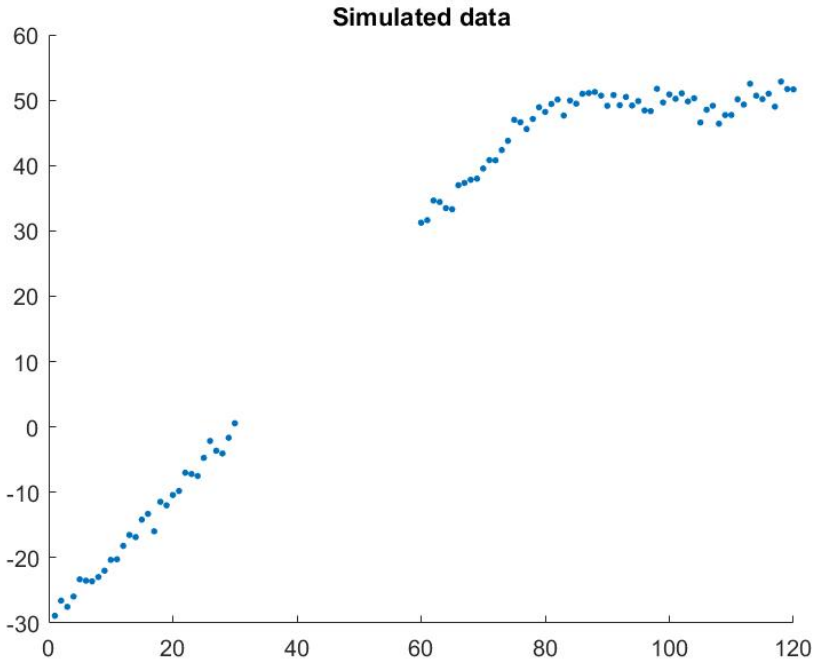

Figure 6.1: The simulated data set described above.

6.2 Results

All methods were run to find one change point. DBEST was run with $\theta_1 = 0.1$, $\theta_2 = 0.2$, $\phi = 24$, default value on ϵ and confidence level $\alpha = 5\%$.

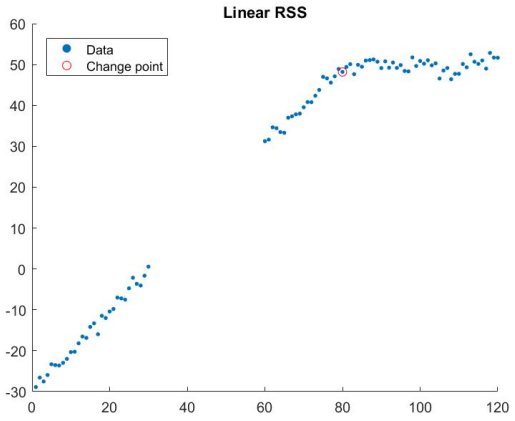

Figure 6.2: Optimal partitioning with linear RSS as cost function and $\gamma = 40$, used on the data in Figure 6.1.

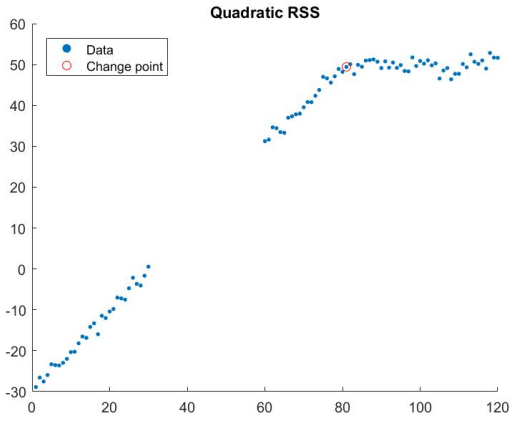

Figure 6.3: Optimal partitioning with the quadratic RSS as cost function and $\gamma = 40$, used on the data in Figure 6.1.

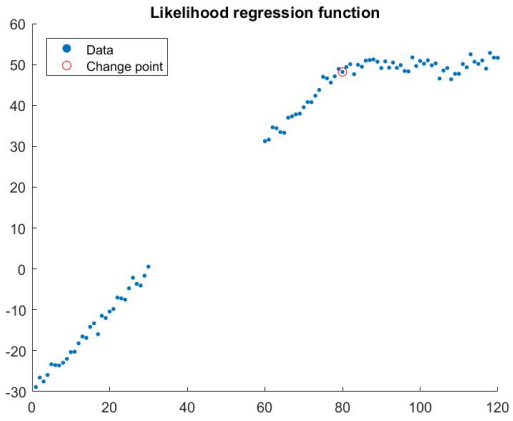

Figure 6.4: Optimal partitioning with the likelihood regression cost function and γ=100, used on the data in Figure 6.1.

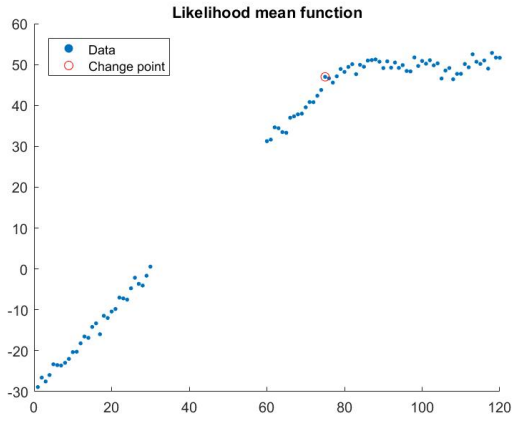

Figure 6.5: Optimal partitioning with the likelihood mean cost function and $\gamma = 100$, used on the data in Figure 6.1.

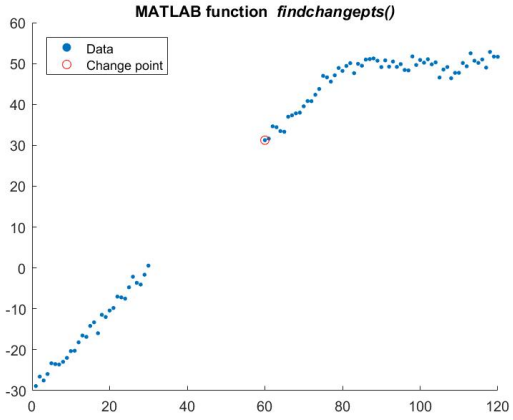

Figure 6.6: The MATLAB function *findchangepts()* used on the data in Figure 6.1.

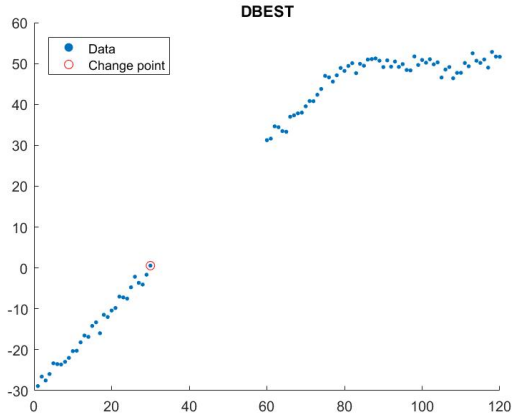

Figure 6.7: DBEST used on the data in Figure 6.1.

6.3 Discussion

The optimal partitioning algorithm with linear RSS or the likelihood regression function as cost function found point 51 as change point whereas the likelihood mean function found point 46 and the quadratic RSS function found point 52. However, the MATLAB function *findchangepts()* erroneously found point 31 and DBEST found point 30 as the start of the change and point 33 as the end of the change. Point 30 is shown in Figure 6.7. The linear and the quadratic RSS, the likelihood regression function and the likelihood mean function were all very close to finding the correct change point, whereas DBEST and the

MATLAB function *findchangepts()* were not. This example shows that the two methods that do not take the time into account have trouble handling missing data points. If the time is not taken into account, there will be an obvious jump in the data at point 31. What is interesting to note is that the likelihood mean function, although not taking time into account, is able to approximately find the correct change point. The reason is that the likelihood mean cost function only searches for differences in mean and not in regression parameters. The model thus contains fewer parameters and is simpler than a linear model. The simpler the model, the fewer complex patterns can be detected and the jump at point 31 will not be detected.

We discuss why DBEST did not find the correct change point a bit more thorough. Since there is a big jump at point 30, this point will probably be detected as a *level shift point* and since the data is relatively smooth and there are no other jumps, the magnitude of the *trend local change function*, h will be large at the jump, whereas at the actual change point (point 52), there is no jump and no peak, see Figure 6.8. Taking the results from the two simulated data sets (Figure 5.1 and Figure 6.1) into account, as well as the performance on the example data in Chapter 4, we conclude that DBEST is not suited for our purposes and therefore we leave it out from further analysis. Moreover, the MATLAB function *findchangepts()*, will be left out, since it did not perform well on the simulated data set in Figure 6.1, even though it performed well on the data in Figure 5.1 and on the example data in Chapter 4. The reason is that since we know that the lake drainage data that we want to analyse can contain big gaps with missing data points, we regard it as important for the models to be able to take missing data points into account.

Lastly, we decide to leave out the quadratic RSS from further evaluation, even though it has performed well so far. The reason is that it has not performed notably better than the linear RSS cost function. Due to the bias-variance trade off described in Section 3.1, we prefer simpler models with fewer parameters, provided that the bias is the same, i.e. that the deviation from the model and the observed data is the same for the models. The quadratic model has more parameters than the linear model and is therefore not to prefer. Modelling the data with the likelihood mean function also leads to fewer parameters compared to the likelihood regression function. Therefore, even though the likelihood mean function did not perform well on the simulated data in Figure 5.1, we keep it for further evaluation.

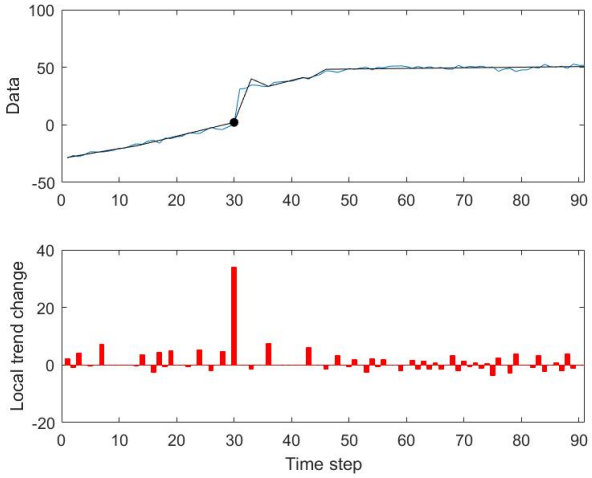

Figure 6.8: Output from DBEST when fitted to the simulated data in Figure 6.1.

Part II

Deeper study of selected methods

In Part I, we have come to the conclusion to leave out DBEST, TIMESAT, the MATLAB function *findchangepts()* and the optimal partitioning algorithm with the quadratic RSS cost function from further analysis. In Chapter 7, we evaluate the linear RSS, the likelihood regression function, the likelihood mean function and a modification of the linear RSS, that we call the absolute value cost function. We will evaluate the four methods on several lake drainage data sets and compare the results to reference change points. See Section 3.1.1 for description and implementation of the linear RSS cost function and Section 3.1.2 for description of the likelihood functions and lastly Section 7.1 for implementation of the absolute value cost function. In Chapter 8, the problem with outliers and noisy data is discussed and we filter the lake drainage data with a median filter and evaluate the four methods on the filtered data. The found change points are also compared to the reference change points. Moreover, in Chapter 9, we show examples of found change points and regression lines that have been fitted to the data. We evaluate the methods on 46 lake drainage data sets, mainly observed between 1985 and 2015. Each data set contains observations for the six different multi-spectral indices TCW, TCG, NDMI, NDVI, NDWI and TCB. For each each index, there are around 50-150 data points.

Chapter 7

Evaluation on original lake drainage data

In this chapter we evaluate the optimal partitioning algorithm wit four cost functions on 46 optical remote sensing lake drainage data sets. Each data set represents one set of observations during a specific time. At every observation, six different parameters, or indices (TCW, TCG, NDMI, NDVI, NDWI and TCB), were measured. Therefore, each data set consist of six time series, (see Figure 4.1 for an example). First, we search for one change point for each index in the 46 lake drainage data sets. The goal is to use these data sets to evaluate how well the four methods find change points. For each method, we count the number of data sets for which the change points differ from the reference change points with at most 3 years and for at least 4 of the six indices.

We also present the deviation from the found change points to reference change points. The reference change points have been manually detected by looking at the data sets. This has been done by Ingmar Nitze. Only one change point has been registered for each data set and the change point has been assumed to be the same for all indices in a data set. Since there are assumptions behind the detection of the optically measured reference change points, these might not be correct.

Last, we investigate how well the cost functions find two change points. The dynamical programming algorithm was run for different penalty constants γ, to find the desired amount of change points. We also comment on whether to search for one, two or additional change points.

7.1 Methods

We evaluate the linear RSS cost function, see Section 3.1.1, the likelihood regression cost function, Section 3.1.2, the likelihood mean cost function, Section 3.1.2 and the absolute value cost function. The absolute value cost function is created to decrease the influence

of noise and outliers, by modifying the linear RSS cost function. We fix τ and τ^*, as in Section 3.1.1. The regression parameters $\hat{\beta}_{0,\tau,\tau^*}$ and $\hat{\beta}_{1,\tau,\tau^*}$ are calculated identically as in Section 3.1.1 minimising the RSS. Also, the estimated values $\hat{y}_i$ for $i = 1, 2, \ldots, n$ are calculated as in Section 3.1.1. The difference lies in the cost function and not in the estimated parameters. With

$$\hat{y}_i = \hat{\beta}_{0,\tau,\tau^*} + \hat{\beta}_{1,\tau,\tau^*} x_i, \quad \forall\ \tau < i \leq \tau^*,$$

and for fixed τ and τ^*, the model calculates the function,

$$\sum_{i=\tau+1}^{\tau^*} |y_i - \hat{y}_i|,$$

as proposed in [12]. The cost function is thus

$$C(x_{\tau+1:\tau^*}, y_{\tau+1:\tau^*}) = \begin{cases} \sum_{i=\tau+1}^{\tau^*} |y_i - \hat{y}_i|, & \tau + 1 < \tau^* - 1, \\ 0, & \text{otherwise}, \end{cases} \tag{7.1}$$

and we call it the *absolute value cost function*. This method is consequently based on the same assumption and parameter estimations as the linear RSS although the absolute value cost function corresponds to minimising the L_1 norm while the linear RSS cost function corresponds to minimising the L_2 norm. Using the absolute value cost function, points far away from the estimated line do not have as great influence on the total cost as using the linear RSS cost function.

7.2 Choice of number of change points

We are interested in deciding whether a specific model should contain zero, one, two, three or even more change points. Therefore, we need to find a method for choosing the number of change points in a model. We investigate how to use the penalty constant γ for this purpose and comment on Akaike's Information Criterion.

Improvement of fit

We use the penalty constant in the dynamical programming algorithm (equation (3.1)) to determine the number of change points to be found. The penalty constant, γ, is a measurement of the improvement of the fit when one change point is added to the model. The largest γ that gives one change point corresponds to the improvement in the cost function from modelling the data with one change point instead of zero. Likewise, the largest γ that gives two change points, corresponds to the improvement in the cost function from modelling the data with two change points instead of one. It can be shown like this. The total value of the function $F(n)$ in the dynamical programming algorithm, equation (3.1), is for zero change points,

$$F_0(n) = C_0,$$

where C_0 is the total cost of the fit, for example the RSS or twice the negative log likelihood function and n is the number of observations in the time series. For one change point we have,

$$F_1(n) = C_1 + \gamma,$$

where C_1 is the total cost using one change point. Similarly, for two change points,

$$F_2(n) = C_2 + 2\gamma,$$

where C_2 is the total cost using two change points. In order for the optimal partitioning algorithm to choose one change point instead of zero we must have $F_1 < F_0$, which results in,

$$C_1 + \gamma < C_0,$$

and in order for the algorithm to find two change points we must have $F_2 < F_1$, or

$$\begin{aligned} & C_2 + 2\gamma < C_1 + \gamma, \\ \implies & C_2 + \gamma < C_1, \end{aligned}$$

and $F_2 < F_0$, or

$$C_2 + 2\gamma < C_0.$$

We can thus use the value of the penalty constant γ to decide what number of change points to choose. However, choosing a limit of γ is arbitrary and user dependent. The reasoning is that if the improvement of the fit is very small with one additional change point, that change point might not be valid. On the contrary, if the improvement of the fit resulting from adding one change point is relatively large, that change point should be valid. However, "very small" and "relatively large" are vague expressions. Therefore, the user should investigate what limits that she/ he finds plausible.

A comment on Akaike's Information Criterion

Akaike's Information Criterion, AIC, is a well known method for determining the number of parameters in a model. Therefore, one could imagine that AIC is a good method for deciding the number of change points to be searched for. AIC is defined as

$$\text{AIC} = 2k - 2\ln(\hat{L}),$$

where k is the total number of estimated parameters and $\hat{L}$ is the likelihood function with the estimated parameters. The model for which AIC is minimised, should be chosen as the valid model.

We analysed the result of AIC for a subset of the lake drainage data sets presented in this report. However, AIC principally chose models with as many change points as possible, up to some large limit. This is not what we would expect if we had succeeded in finding a sufficiently good model to describe the data. Therefore, we simulated several data sets to investigate whether AIC was suitable for the purpose of determining the number of change points in noisy lake drainage data. Since the lake drainage data sets contain around 100 observations, we simulated a data set with 50 observations randomised from $Y_1 \sim N(0, 0.5)$ and 50 observations randomised from $Y_2 \sim N(4, 0.5)$ in order to have an obvious change point at the 51^{st} observation. Running the dynamical programming algorithm with the likelihood function as cost function, and comparing the result of AIC for 0, 1, 2, 3, etc. change points, we found that AIC still decreased with the number of change points, until some limit, where it again increased. We would expect it to decrease from zero to one change point and then increase from one to two change points. We also noted that if we chose the first change points to be the 51^{st} and randomly chose the second and third change point etc, AIC was smallest for the model with one change point. The results above also hold for the linear RSS model.

Secondly, we simulated another data set containing 50 observations randomly generated from N(0, 0.0001) and 50 observations randomly generated from N(4, 0.0001), i.e. with a significantly smaller variance than in the previous simulation. Then, the result was as expected, i.e. the best model according to the AIC was the one with one change point. Therefore, we conclude that the reason why we cannot use AIC for determining the accurate number of change points is that the variances or the noise components in the lake drainage data sets are too large. To reduce the noise, AIC prefers large models. We conclude that we cannot use AIC to determine the number of change points in our data.

7.3 Results and discussion

One change point

The performance of the methods on the original data sets is presented in Table 7.1. Recall that each data set consists of six time series describing the indices TCW, TCG, NDMI, NDVI, NDWI and TCB. We compare the found change points to the reference change points and calculate the mean deviation from the reference change points. We also calculate the number of found change points for which the deviation to the reference change points is 0 respectively $\pm$ 1-3 and present the number of change points that have not been found for each method. (It is assumed that every time series has one change point). Table 7.1 suggests that the likelihood mean function is the best cost function for finding change points in our data, followed by the absolute value cost function.

Cost function	Deviation 0	Deviation ± 1-3	Mean deviation	Entries with no change points
Linear RSS	38 %	31 %	2.6	10 %
Likelihood regression	39 %	29 %	2.5	5.6 %
Likelihood mean	54 %	27 %	1.2	8.3 %
Absolute value	41 %	30 %	2.3	10 %

Table 7.1: The number of observations with certain deviation from the reference change points.

We also estimate how accurate the methods are based on that they should find approximately the same change points for at least the indices TCW, NDMI, NDWI and TCB, since there might be a delay in the indices TCG and NDVI. We count the number of data sets for which at least four of the change points, found in one data set, differ by at most 3 years. The linear RSS function, fulfils these requirements for 34 % of the data sets and the likelihood regression function fulfils the requirements for 43 % of the data sets. The likelihood mean function and the absolute value function fulfil these requirements for 65 % and 46 % respectively. We therefore conclude that the likelihood mean function performs best on the original data, followed by the absolute value function, the linear regression function and last the linear RSS function.

Two change points

We also comment on the result when two change points have been found. We conclude that for very few data sets, the change points differ with at most 3 for at least four of the six indices, hence we cannot draw any conclusions about the methods by searching for more than one change point.

Chapter 8

Evaluation on filtered lake drainage data

In this chapter we discuss the problem with noisy data. We suggest a filter to apply on the data and evaluate the performance of the four remaining methods on the filtered data. We also compare the performance on the filtered data sets with the performance on the original data sets. The idea is to use the methods on the filtered data sets, to find the change points in the original data sets. We filter the 46 data sets used in Chapter 7. As described in Chapter 7, each data set represent one set of measurements during a specific time. At every observation, six different parameters or indices were measured. Therefore, each data set consist of six time series, (see Figure 4.1 for an example). First, we search for one change point. For each method, we count the number of data sets for which the change points differ with at most 3 for at least four of the six indices. We also present the deviation from the reference change points that have been found manually.

We also investigate how well the cost functions find two change points. The dynamical programming algorithm was run for different penalty constants γ, to find the desired amount of change points. We also comment on whether to search for one, two or additional change points.

8.1 Dealing with noise and outliers

As mentioned in Chapters 1 and 2, the lake drainage data that we analyse are likely to contain outliers and other noise. Outliers and noise can have a great influence on the parameter estimation. If it is obvious that an observation depends on for example measurement errors, that observation can be regarded as an outlier and removed from the time series [2, p.23]. However, we do not have information on whether a certain observation depends on measurement errors or on environmental conditions. Therefore, we need to find other methods to deal with outliers.

Median filter

One approach to decrease the influence of outliers and noise is to use a filter to smooth the data. We apply a median filter on the data, among others presented in [21], and run the different methods on the filtered data. We start with the original data set $(x_1, y_1), (x_2, y_2) \dots (x_n, y_n)$ and create a new data set $(x_1, \tilde{y}_1), (x_2, \tilde{y}_2) \dots (x_n, \tilde{y}_n)$. The value of a point $\tilde{y}_i$ in the filtered data set will be the median of the values in a pre specified window around point y_i. For example, if the window size is 3 points, the $\tilde{y}_i$ in the filtered data set will be the median of the points y_{i-1}, y_i and y_{i+1} in the original data set. This approach cannot be used at the end points $\tilde{y}_1$ and $\tilde{y}_n$. Here, we need another method. At the boundaries we will add to the original data series multiple boundary values. For example, if the window size is 5 points, the new data set with original values will be $y_{-1}, y_0, y_1, y_2, y_3, \dots, y_{n-1}, y_n, y_{n+1}, y_{n+2}$, with $y_{-1} = y_0 = y_1$ and $y_{n+2} = y_{n+1} = y_n$. Now, the median can be taken at the points y_2 and y_{n-1}. However, at the boundaries $\tilde{y}_1$ and $\tilde{y}_n$, the median will always be the boundary values respectively and these will not be affected by the filter. Therefore, we remove the boundary values from the filtered data set. For the linear RSS, the likelihood regression function and the absolute value function presented in equation (7.1), the window was set to 5 points. For the likelihood mean cost function however, a 5-point-window mostly resulted in no found change points. Therefore, the likelihood mean function was evaluated on data filtered with window size 3. Observe that since we remove the first and the last value from the filtered data set, point $\tilde{y}_i$ in the filtered data set will correspond to point y_{i+1} in the original data set.

A comment on the likelihood functions

Outliers and fluctuations can be made less influential when modelling with the likelihood functions by specifying a limiting distance between the change points. Then, the estimate of the variance will be more accurate. The limiting number of points between two change points is chosen by setting the likelihood function to infinity for segments smaller than the limiting number. The smaller the number between two change points, the more inaccurate the estimation of the variance, since there is a smaller number of observations in the estimation.

8.2 Results and discussion

One change point

First, as in Section 7.3, we evaluate the performance of the methods on the filtered data sets, when one change point has been searched for. Observe that the likelihood mean cost function has been applied on data sets filtered with a median filter with window size 3 and the other cost functions have been used on data sets filtered with window size 5. This was because the likelihood mean function did not find any change points for the majority of the data sets, when applied to the data filtered with window size 5. We

count the number of entries that have zero deviation from the year of the reference change point. Also, the number of found change points that deviate more than zero and less than four years from the reference change points are calculated. The mean deviation is also calculated. Remember that the years of the reference change points have been manually observed and are therefore not always reliable. By looking in Table 8.1, we see how well the methods perform. We notice that the likelihood mean function has the best performance. The absolute value function performs second best, followed by the likelihood regression function and the linear RSS function.

Cost function	Deviation 0	Deviation ± 1-3	Mean deviation	Entries with no change points
Linear RSS	37 %	34 %	3.0	4.0 %
Likelihood regression	38 %	29 %	2.8	11 %
Likelihood mean	55 %	27 %	1.2	4.0 %
Absolute value	40 %	31 %	2.8	4.0 %

Table 8.1: The number of observations with certain deviation from the reference change points.

We also estimate how accurate the methods are based on that they should find approximately the same change points for at least the indices TCW, NDMI, NDVI and TCB. We count the number of data sets for which at least 4 of the change points, found in one data set, differ by at most 3 years. The linear RSS function, fulfils these requirements for 33 % of the data sets and the likelihood regression function fulfils the requirements for 41 % of the data sets. The likelihood mean function and the absolute value function fulfil these requirements for 67 % and 48 % respectively. We therefore conclude that the likelihood mean function performs best on the filtered data, followed by the absolute value function, the linear regression function and last the linear RSS function.

With these results and the results from Chapter 7, we conclude that the likelihood mean cost function performs best on both the original data sets and on the filtered data sets. The absolute value cost function performs second best, followed by the likelihood regression cost function and the linear RSS cost function. We also conclude that the overall performances of the methods on the original data and the filtered data, do not differ significantly.

Two change points

Let us comment on the performance of the methods when searching for two change points. As in Section 7.3, we conclude that for very few of the data sets, the change points differ with less than 3 years for at least four of the data sets.

Chapter 9

Examples of change points and regression lines

In this chapter we present figures showing some examples of change points that have been found for different data sets and different cost functions. The change points have been found with the median filter applied and the figures show the filtered data. We also show some trends that have been fitted to the time series after the change points have been detected. The order of the polynomials describing the fitted trends have not been chosen with respect to the cost function that has been used to find the change points.

9.1 Examples

The linear RSS

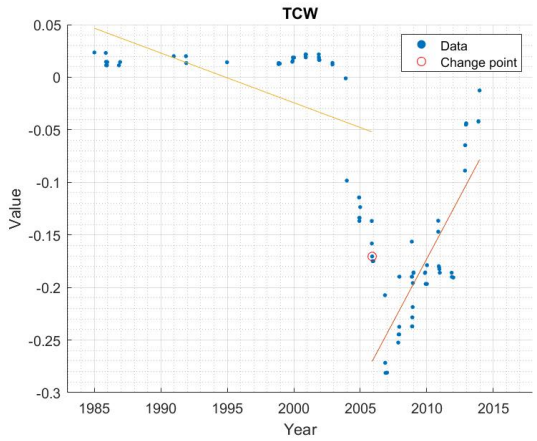

Figure 9.1: Data of TCW segmented with linear RSS as cost function.

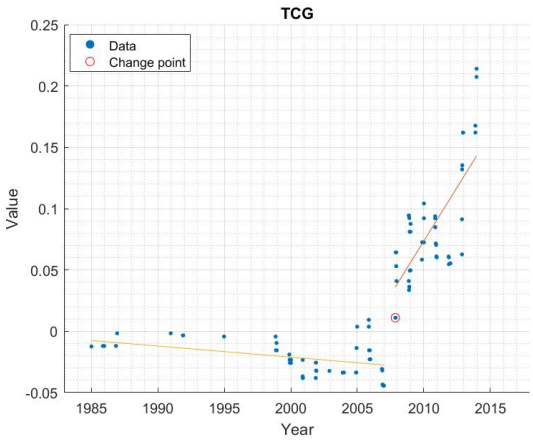

Figure 9.2: Data of TCG segmented with the linear RSS cost function.

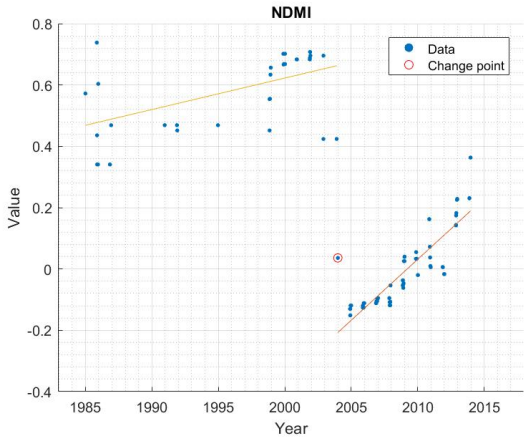

Figure 9.3: Data of NDMI segmented with the linear RSS cost function.

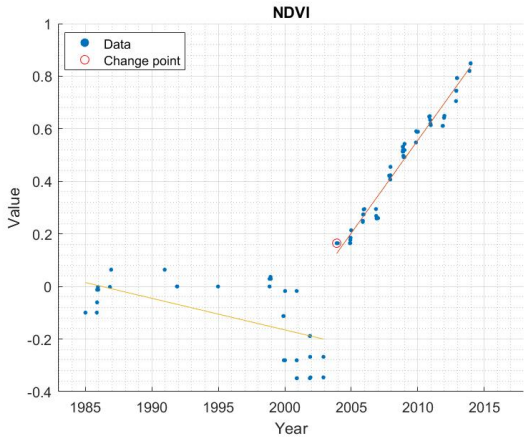

Figure 9.4: Data of NDVI segmented with the linear RSS cost function.

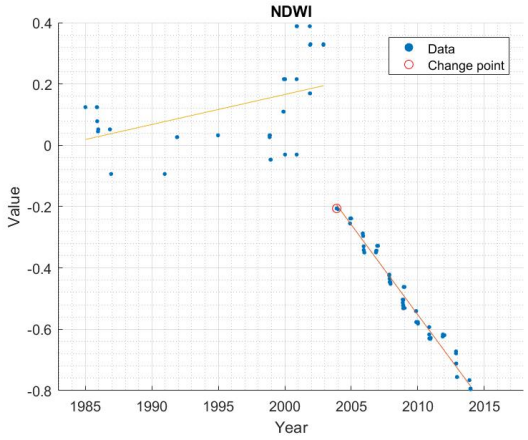

Figure 9.5: Data of NDWI segmented with the linear RSS cost function.

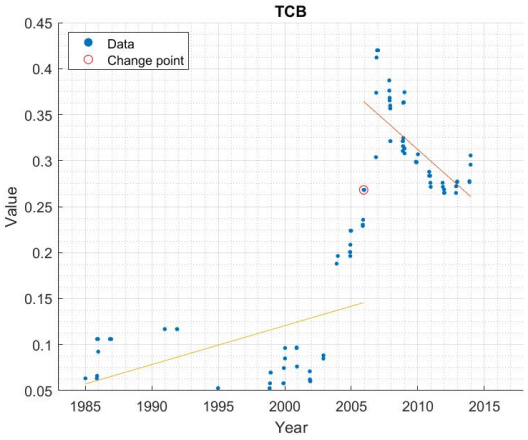

Figure 9.6: Data of TCB segmented with the linear RSS cost function.

The likelihood regression function

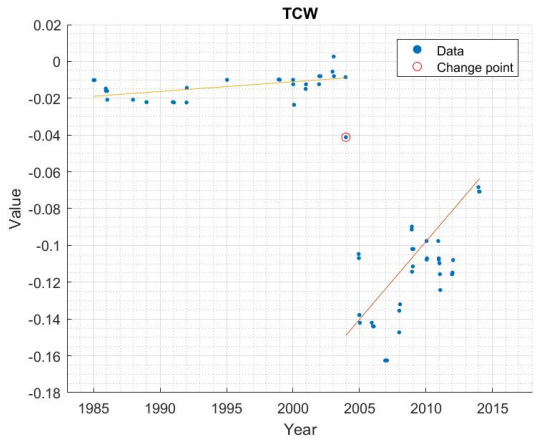

Figure 9.7: Data of TCW segmented with the likelihood regression cost function.

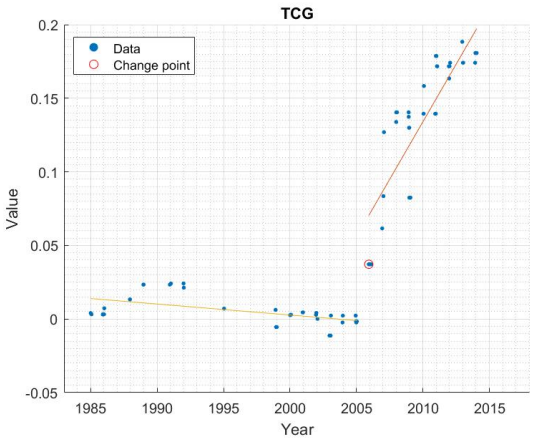

Figure 9.8: Data of TCG segmented with the likelihood regression cost function.

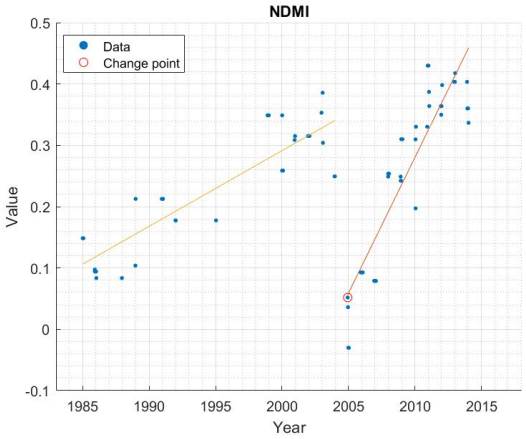

Figure 9.9: Data of NDMI segmented with the likelihood regression cost function.

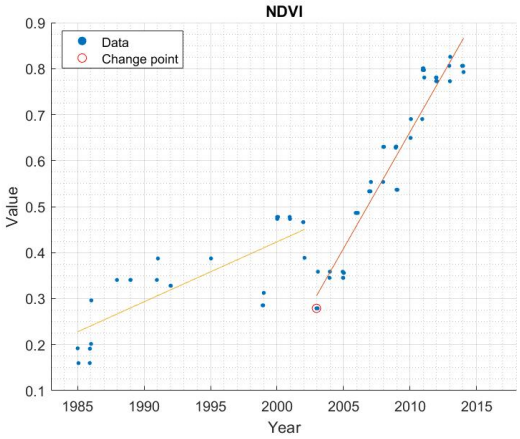

Figure 9.10: Data of NDVI segmented with the likelihood regression cost function.

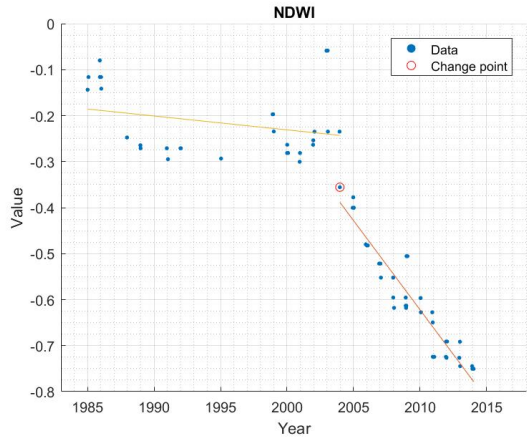

Figure 9.11: Data of NDWI segmented with the likelihood regression cost function.

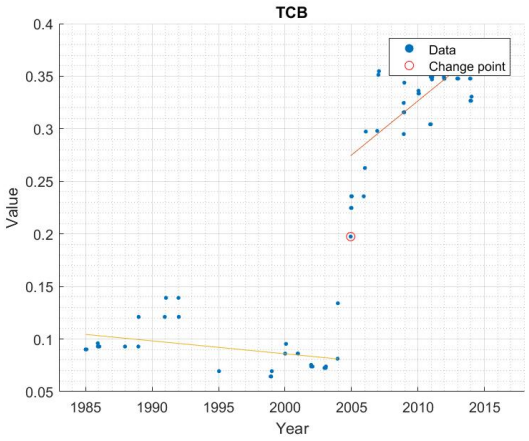

Figure 9.12: Data of TCB segmented with the likelihood regression cost function.

The likelihood mean function

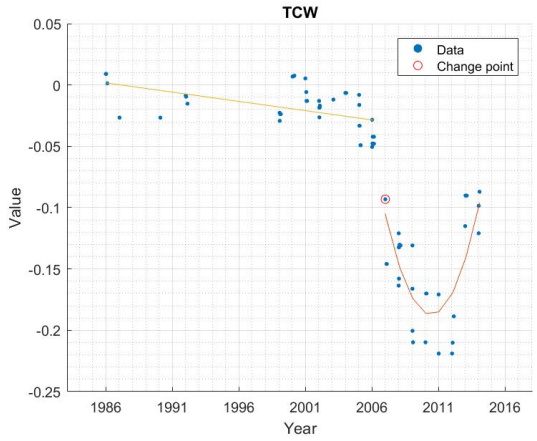

Figure 9.13: Data of TCW segmented with the likelihood mean cost function.

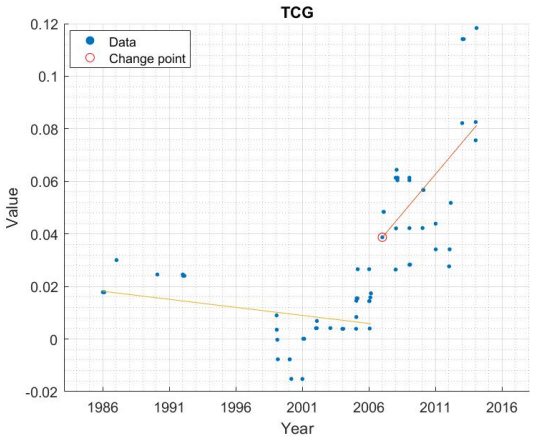

Figure 9.14: Data of TCG segmented with the likelihood mean cost function.

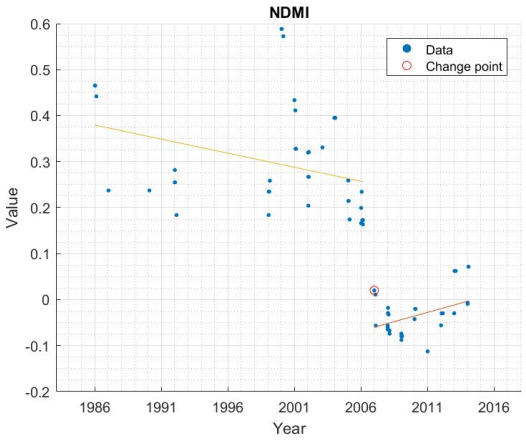

Figure 9.15: Data of NDMI segmented with the likelihood mean cost function.

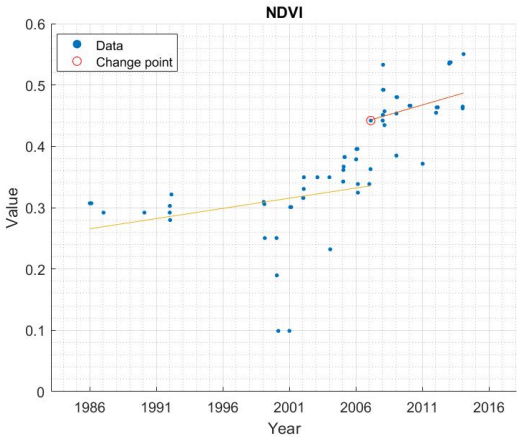

Figure 9.16: Data of NDVI segmented with the likelihood mean cost function.

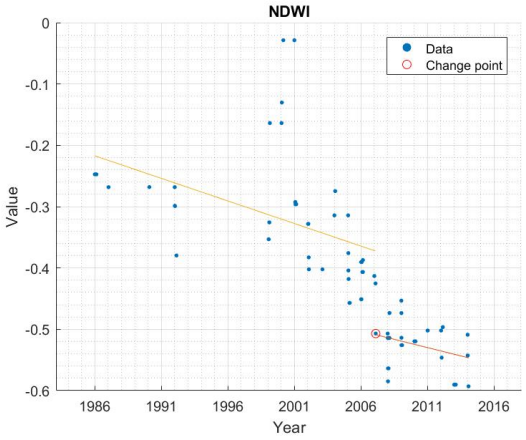

Figure 9.17: Data of NDWI segmented with the likelihood mean cost function.

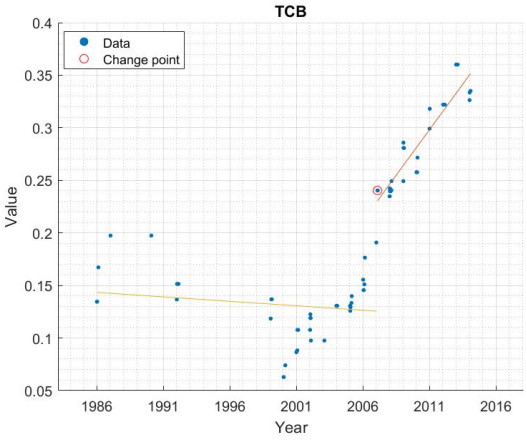

Figure 9.18: Data of TCB segmented with the likelihood mean cost function.

The absolute value function

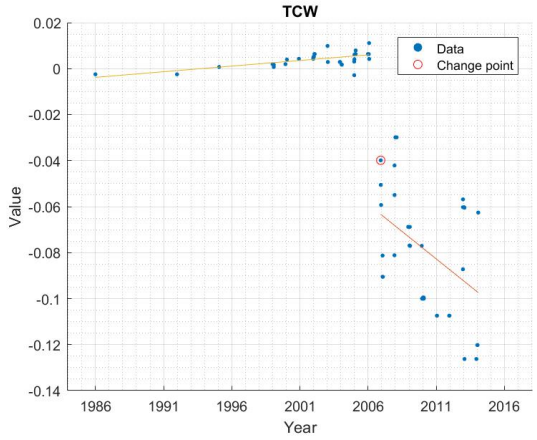

Figure 9.19: Data of TCW segmented with the absolute value cost function.

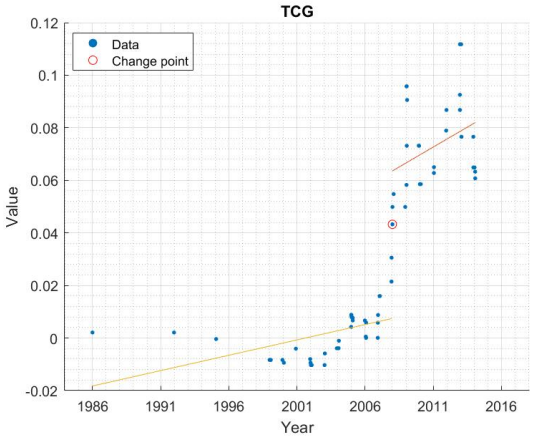

Figure 9.20: Data of TCG segmented with the absolute value cost function.

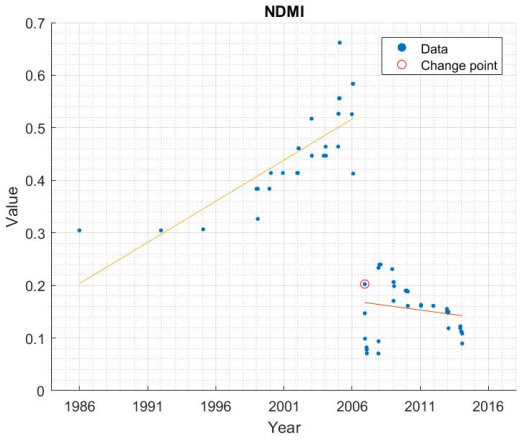

Figure 9.21: Data of NDMI segmented with the absolute value cost function.

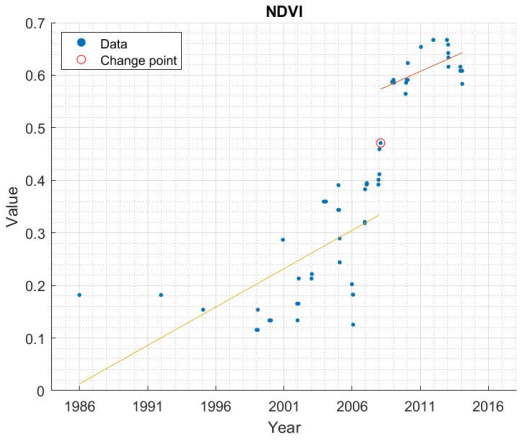

Figure 9.22: Data of NDVI segmented with the absolute value cost function.

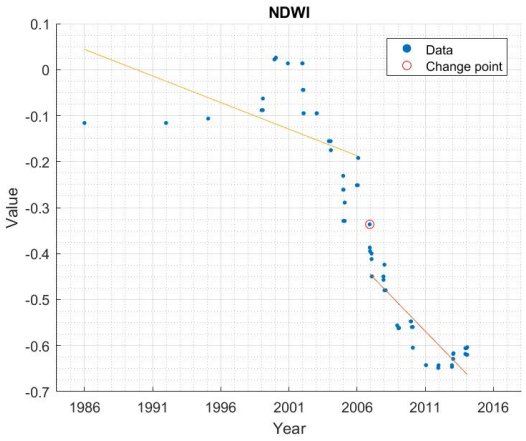

Figure 9.23: Data of NDWI segmented with the absolute value cost function.

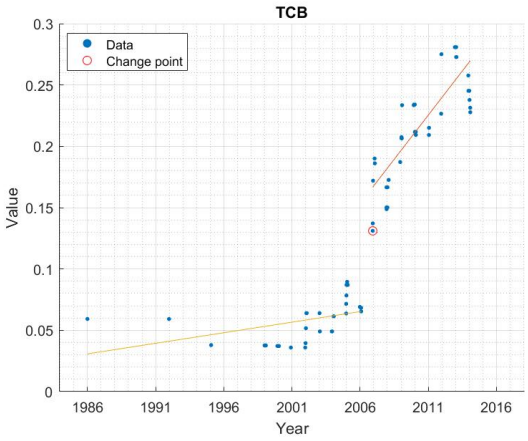

Figure 9.24: Data of TCB segmented with the absolute value cost function.

9.2 Trend analysis

As described in the introduction, after having found change points, trends describing the data should be removed from the time series. It is desirable that the remaining component is stationary. The definition of stationarity follows below and is further described in [2, p. 15].

Definition 9.2.1. Stationarity
A time series $\{X_t\}$ is (weakly) stationary if $\mu_X(t)$ is independent of t and $\gamma_X(t+h,t)$ is independent of t for each h, where $\mu_X(t) = E(X_t)$ is the mean function of $\{X_t\}$ and $\gamma_X(t+h,t) = Cov(X_{t+h}, X_t)$ is the covariance function of $\{X_t\}$ for all h.

After finding change points in the data, the idea is to find trends that describe the data. Thereafter, the trend components are removed from the data and it is investigated whether the data is stationary or not. A failure to detect stationary properties might be because an inaccurate change point has been found, too few change points have been chosen or a trend that is to simple has been removed from the data. If stationary properties are found, we can conclude that we have found trends describing the data. A test for stationarity was done on the time series that are shown in Figures 9.13-9.18 and that have been analysed with the likelihood mean cost function. The MATLAB function *kpsstest()*, see [19], was used to check for stationarity. Fitting polynomials with degree zero to the original data sets, the hypothesis of stationarity was rejected at a 5 % confidence level for the indices TCW and TCG but not for the other indices. However, the result from removing linear trends with degree one from the data sets, was that the hypothesis of stationarity was not rejected at a 5 % confidence level for any of the six indices. Therefore we conclude that by change point analysis and trend removal we have been successful in creating stationary time series.

9.3 Discussion

We observe that some of the trends that have been fitted to the data sets above have a very gentle slope, see for example Figure 9.16. This is because the linear properties are almost none, and it would be more suitable to fit a straight line with no slope to the segments. The likelihood mean function, that models the data with a piecewise constant function, finds all correct change points in the time series presented in Figures 9.13-9.18, whereas the likelihood regression function, that models the data with a piecewise linear function, does not. However, we have seen in this chapter, that the time series in Figures 9.13 and 9.14 are better modelled with linear trends, since they did not become stationary when constant functions were removed but when linear trends were removed. We conclude that even though a time series is best modelled by polynomials of a certain degree, methods for detecting change points that model the data with polynomials of a different degree could be better. We have shown that this is the case with the time series presented in Figures 9.13 and 9.14. What can also be done to investigate whether a trend has been successfully found in a time series is to make confidence intervals for the trend parameters, but we do not show any examples of that in this study.

Part III

Concluding discussion

Chapter 10

Discussion

Rejected methods

We have concluded that DBEST, TIMESAT, the MATLAB function *findchangepts()* and the optimal partitioning algorithm with the quadratic RSS cost function are not suitable for our purposes. The segmentation method in DBEST assumes equidistant time steps between each value which could be complicated when dealing with time series of remote sensing data. One solution is to reduce the time series to have equidistant time steps between each point. That is, however, not a good solution since for most lake drainage data sets the number of observations will then be too small to find any accurate patterns. TIMESAT also assumes equidistant time step between the observations and the program also requires the user to specify the number of observations each year [5], which must be the same for all years. This cannot be done with our data. Also, TIMESAT primarily investigates seasonal components, which is not in our main interest. We further conclude that the MATLAB function *findchangepts()* does not take the time interval into account, and therefore fails to detect the correct change point in the constructed data set in Chapter 6. Last, we conclude that the quadratic RSS cost function can reduce the RSS more than the linear RSS cost function, if the data is noisy. However, the quadratic RSS is more flexible than the linear RSS and thus more sensitive to variability in the data. This may result in overfitting the data, and we have therefore rejected the optimal partitioning algorithm with the quadratic RSS cost function as a method to find change points in optical remote sensing lake drainage data. A short comparison of the rejected methods is shown in Table 10.1. 'Nr. of parameters chosen by the user' is the minimum number of parameters that has to be specified by the user. For DBEST, TIMESAT and the MATLAB function *findchangepts()* additional parameters can be specified.

Property	DBEST	TIMESAT	Quadratic RSS	*findchangepts*()
Considering time	No	No	Yes	No
Sensitive to outliers	Yes	-	Yes	-
Nr. of parameters chosen by the user	4	0	1	0

Table 10.1: Comparison of the methods that are presented and left out in Part I.

Impact of the median filter

The data sets that we are dealing with are noisy and have lots of so called missing data points. The number of points are in general somewhere between 50 and 150 and the change points are in many cases not obvious to the eye. Because of the few and missing data points, there is a high risk that the noise component influences the choice of change points, even though methods that account for missing values are used. Filtering the series reduces the noise component but does not fill in missing data values. Comparing the performance of the methods on the original data sets and on the filtered data sets however, we draw the conclusion that filtering the data sets have not improved the performance of the methods.

Comparison of approved methods

We have concluded that the optimal partitioning algorithm with the likelihood mean function, best finds change points in optical remote sensing lake drainage data, compare Tables 7.1 and 8.1. The second best method for finding change points in our data is the absolute value cost function followed by the likelihood regression cost function and the linear RSS cost function. We also conclude that simpler models are to prefer when searching for change points in optical remote sensing lake drainage data. A short comparison between the methods is shown in Table 10.2, where k corresponds to the number of segments that the time series is divided into.

Property	Linear RSS	Likelihood regression	Likelihood mean	Linear absolute val.
Considering time	Yes	Yes	Yes	Yes
Nr. of estimated parameters, k segments	$2 \cdot k + 1$	$3 \cdot k$	$2 \cdot k$	$2 \cdot k + 1$
Nr. of parameters chosen by the user	1	1	1	1

Table 10.2: Comparison of the methods that are evaluated in Part I and Part II.

Future research

Since we have concluded that the optimal partitioning algorithm with the likelihood mean function performs best on our data, it would be interesting to investigate whether even simpler cost functions with fewer parameters, e.g. least-squares regression that models the data with a piecewise constant function but with a fixed variance, performs even better. Furthermore, there are faster partitioning algorithms that can be used instead of the optimal partitioning algorithm. Using the same cost functions, the results will be the same but the execution time faster. For example, the Pruned Exact Linear Time (PELT) method is presented in [16]. It is a partitioning algorithm that uses pruning, i.e. removing values that cannot be minima in (3.1). In the worst case scenario the complexity will be $\mathcal{O}(n^2)$, which is the complexity of the optimal partitioning algorithm described in equation (3.1).

Bibliography

[1] Blom, G., Enger, J., Englund, G., Grandell, J., & Holst, L. (2008). *Sannolikhetsteori och statistikteori med tillämpningar.* Studentlitteratur.

[2] Brockwell, P. J., & Davis, R. A. (2016).*Introduction to time series and forecasting.* Springer.

[3] Burnham, K. P., Anderson, D. R. (2002). *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach.* 2nd Ed. Springer-Verlag.

[4] Chen, J. & Gupta, A. K. (2000). *Parametric statistical change point analysis.* Birkhäuser.

[5] Eklundh, L. & Jönsson, P. (2015). Timesat 3.2 Software Manual. Lund and Malmö University, Sweden.

[6] Gao, B. C. (1996). NDWI - a normalized difference water index for remote sensing of vegetation liquid water from space. *Remote Sensing of Environment*, 58(3), 257-266.

[7] Grosse G. Intro to remote sensing of permafrost landscapes and dynamics. hdl:10013/epic.48618.d001. Retrieved July 14, 2017.

[8] Grosse G., Jones B., & Arp C. (2013). Thermokarst lakes, drainage, and drained basins. *Treatise on Geomorphology*, 8, Glacial and Periglacial Geomorphology, San Diego: Academic Press, 325-353.

[9] Jackson, B., Scargle, J. D., Barnes, D., Arabhi, S., Alt, A., Gioumousis, P., Gwin, E., Sangtrakulcharoen, P., Tan, L., & Tsai, T. T. (2005). An algorithm for optimal partitioning of data on an interval. *IEEE Signal Processing Letters*, 12(2), 105–108.

[10] Jamali, S., Jönsson, P., Eklundh, L., Ardö, J., & Seaquist, J. (2015). Detecting changes in vegetation trends using time series segmentation. *Remote Sensing of Environment*, 156, 182-195.

[11] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning with applications in R.* Springer Texts in Statistics.

[12] John, G. H. Robust decision trees: Removing outliers in databases. (1995). First International Conference on Knowledge Discovery and Data Mining, Menlo Park, CA, AAAI Press, 174-179.

[13] Jönsson, P., & Eklundh, L. (2002). Seasonality extraction and noise removal by function fitting to time-series of satellite sensor data. *IEEE Transactions of Geoscience and Remote Sensing*, 40(8), 1824 – 1832.

[14] Jönsson, P., & Eklundh, L. (2004). TIMESAT—a program for analyzing time-series of satellite sensor data. *Computers & Geosciences*, 30(8), 833-845.

[15] Kandasamy, S., Baret, F., Verger, A., Neveux, P., & Weiss, M. (2013). A comparison of methods for smoothing and gap filling time series of remote sensing observations-applications to MODIS LAI products. *Biogeosciences*, 10, 4055-4071.

[16] Killick, R., Fearnhead, P., & Eckley, I.A. (2012). Optimal detection of change-points with a linear computational cost. *Journal of the American Statistical Association*, 107(500), 1590-1598.

[17] Lavielle, M. (2005). Using penalized contrasts for the change-point problem. *Signal Processing*, 85(8), 1501-1510.

[18] Matlab. (2017). Statistics and machine learning toolbox user's guide. Mathworks, Inc.

[19] Matlab. (2017). Econometrics toolbox. Mathworks, Inc.

[20] Nitze, I., & Grosse, G. (2016). Detection of landscape dynamics in the Arctic Lena Delta with temporally dense Landsat time-series stacks. *Remote Sensing of Environment*, 181, 27-41.

[21] Pratt, W. K. (2007). *Digital Image Processing*. 4th Ed. John Wiley & Sons.

[22] Westermann, S., Duguay, C. R., Grosse, G. & Kääb, A. (2015). Remote sensing of permafrost and frozen ground. *Remote Sensing of the Cryosphere*, Wiley.

[23] Measuring Vegetation (NDVI & EVI). earthobservatory.nasa.gov. Retrieved June 8, 2017.

[24] The Landsat Program. landsat.gsfc.nasa.gov. Retrieved June 8, 2017.

TRITA -MAT-E 2017:51
ISRN -KTH/MAT/E--17/51--SE