

Change Detection in Time Series Data Using Wavelet Footprints

Mehdi Sharifzadeh, Farnaz Azmoodeh and Cyrus Shahabi
Computer Science Department
University of Southern California
Los Angeles, CA 90089-0781
[sharifza, azmoodeh, shahabi]@usc.edu

ABSTRACT

Detecting changes in time series data is an important data analysis task with application in various scientific domains. In this paper, we propose a novel approach to address the problem of change detection in time series data, which can find both the amplitude and *degree* of changes. Our approach is based on *wavelet footprints* proposed originally by the signal processing community for signal compression. We, however, exploit the properties of footprints to efficiently capture discontinuities in a signal. We show that transforming time series data using footprint basis up to degree D generates nonzero coefficients only at the change points with degree up to D . Exploiting this property, we propose a novel change detection query processing scheme which employs footprint-transformed data to identify change points, their amplitudes, and degrees of change efficiently and accurately. We also present two methods for exact and approximate transformation of data. Our analytical and empirical results with both synthetic and real-world data show that our approach outperforms the best known change detection approach in terms of both performance and accuracy. Furthermore, unlike the state of the art approaches, our query response time is independent from the number of change points in the data and the user-defined change threshold.

1. INTRODUCTION

Time series data are generated, maintained, and processed within a broad range of application domains in different fields such as economics, meteorology, or sociology. Moreover, recent advances in the manufacturing of modern sensory devices have caused several applications to utilize these sensors towards better understanding of the physical world. These sensors when deployed in an environment generate large amount of measurement data streams which can be stored as time series data.

Mining such time series data becomes vital as the applications demand for understanding the underlying processes/phenomena that generate the data. There has been an explosion of interest within the data mining community in indexing, segmenting, clustering, and classifying time series [13, 14, 15, 16]. A specific interesting mining task is to detect change points in a given time series [21, 12, 23, 7, 8]. These are the time positions in the original data where the local trend in the data values has changed. They may indicate the points in time when external events have caused the underlying process to behave differently.

The problem of detecting change in time series has been mostly studied in the class of segmentation problems [14, 6] where each portion of the data is modelled by a known function. Subsequently, change points are defined as the points in data where two adjacent segments of the time series are connected. However, there are real-world applications in which only the position of the change is required and not the fitting functions.

For the past year, we have been working with Chevron-Texaco on mining real data generated during oil well tests. This is a real-world petroleum engineering application studied within the USC's Center of Excellence for Research and Academic Training on Interactive Smart Oilfield Technologies (CiSoft). Petroleum engineers deploy sensors in oil wells to monitor different characteristics of the underlying reservoir. Here, the underneath pressure values measured by sensors form a time series. When the second derivative in the pressure vs. time plot becomes fixed (i.e., a radial flow event in their terminology), they estimate the "permeability" of the reservoir [11]. At the same time, they would like to know if the first derivative is changing. To us, the points where the second derivative becomes fixed are the positions in the pressure time series where a change of degree 1 or 2 occurs. In this paper we focus on identifying both the change points and degrees in time series data. While the definition of change is highly application-specific, we focus on points in data where discontinuities occur in the data or any of its i th derivatives. Moreover, we consider the notion of *degree* of change as the degree of the changing derivatives at the change point. This general definition of change has been broadly used in many scientific application areas such as petroleum engineering [11]. However, its significance has been ignored within the data mining community.

We propose a novel efficient approach to find change points in time series data. Our approach utilizes *wavelet footprints*, a new family of wavelets recently introduced by the signal processing community for signal compression and denoising [5]. While footprints are defined to address a different problem in a different context, we exploit their interesting properties that make them a powerful data analysis tools for our change detection problem. Our contribution starts with employing the idea of wavelet footprints in the context of a data mining problem. This is yet another example of adapting signal processing techniques for the purpose of data mining which started by Vitter et. al proposing the use of wavelets in answering OLAP range-sum queries [19], and Chakrabarti et. al using multi-dimensional wavelets for

general approximate query processing [1].

We show that footprints efficiently capture discontinuities of any degree in the time series data by gathering the change information in the corresponding coefficients. Motivated by this property, we make the following additional contributions:

- We propose two database-friendly methods to transform the time series data using footprints up to degree D . These methods enable us to detect all the change points of degree 0 to degree D , their corresponding amplitude and degree of change. While our *lazy* method only approximates the changes, our *absolute* method computes the exact changes together with their degree and amplitude. To the best of our knowledge, this is the first change detection approach that captures all the above parameters at the same time.
- While we transform the data using footprints, our methods can work with any user-defined threshold value. That is, there is no need to rerun our algorithms each time the user-defined threshold value changes; we answer any new query via a single scan over the transformed data to return the coefficients greater than the user threshold. This is a considerable improvement over the best change detection algorithms which are highly dependent on this threshold value.
- Both analytically and empirically, we show that our query processing schemes significantly outperform the state of the art change detection methods in terms of performance. Employing either our transformation methods, our query response time is independent from the number of change points in the data. This is while both methods demonstrate a dramatical increase in accuracy.

The remainder of the paper is organized as follows. Section 2 reviews the current data mining research on change detection in time series data. Section 3 provides the background on linear algebra and wavelet theory. In Section 4, we describe the idea of using footprints to capture discontinuities in piecewise linear time series. Section 5 generalizes the concept of footprints of degree zero to provide a formal definition for wavelet footprints of arbitrary degrees. We propose our change detection approach and the lazy and absolute methods for footprint transformation in Section 6. In Section 7, we show how our footprint-based approach can be incorporated within systems where time series data is stored in wavelet domain. Section 8 includes our experimental results, and Section 9 discusses the conclusion and our future plans.

2. RELATED WORK

Change detection in time series has newly received considerable attention in the field of data mining. Change detection has also been studied for a long time in statistics literature where the main purpose is to find the number of change-points first and identify the stationary model to fit the dataset based on the number of change points.

In the data mining literature, change detection has mainly been studied in the time series segmentation problems. Most of these studies use *linear* interpolation to approximate the

signal with a series of best fitting lines and return the end-points of the segments as change points in the time series. However, there are many examples of real-world time series which fitting a linear model is inappropriate. For example, Puttagunta et al. [21] use incremental LSR to detect the change points and outlier points with the assumption that the data can be fit with linear models. Also Keogh et al. [12] use probabilistic methods to fit the data with linear segments in order to find patterns in time series.

Yamanish et al. [23] reduce the problem of change point detection in time series into that of outlier detection from time series of moving-averaged scores. Although they consider a wide group of models for fitting the data, their approach is time intensive and hardly scales to high volume datasets. Ge et al. [7] extend hidden semi markov model for change detection. Their solution is applicable to different data distributions using different regression functions; however, it is not scalable to large size datasets due to its time complexity.

Guralnik et al. [8] suggest using maximum likelihood technique to find the best function to fit the data in each segment. Their method is mainly based on the trade-off between the data fit quality and the number of estimated changes. They also consider a wider group of curve fitting functions; however, they do not consider the possible disagreement among different human observers on the actual change points. Also their approach lacks enough flexibility in the sense that they have to rerun the algorithm for different change thresholds asserted by the user.

The method described in this paper is mostly similar to the work done by Guralnik et al. in [8]; however, we return all the possible change points for several polynomial curve fitting functions since then users have the flexibility to focus only on the interesting change points. For example, only the change points found by using quadratic and linear models. Moreover, after we find the change points once, there is no need to rerun the algorithm for different change thresholds asserted by the user.

3. PRELIMINARIES

We consider time series \mathbf{X}_n of size n as a vector (x_1, \dots, x_n) where each x_i is a real number (i.e., $x_i \in \mathbb{R}$). Given F , a class of functions (e.g., polynomials), one can find the piecewise segmented function $\mathcal{X} : [1, n] \rightarrow \mathbb{R}$ that models time series \mathbf{X}_n as follows:

$$\mathcal{X}(t) = \begin{cases} P_1(t) + e_1(t) & 1 < t \leq \theta_1 \\ P_2(t) + e_2(t) & \theta_1 < t \leq \theta_2 \\ \cdot & \cdot \\ \cdot & \cdot \\ P_{K+1}(t) + e_{K+1}(t) & \theta_K < t \leq n \end{cases} \quad (1)$$

Each function P_i is a member of class F that best fits the corresponding segment of the data in \mathbf{X}_n and each $e_i(t)$ is the amount of error introduced when fitting the data with F . Our ultimate goal is to identify $\theta_1, \dots, \theta_K$ where P_i 's are not known a priori. We refer to these points as the *change points* in data where discontinuities occur in the data or its derivatives. We use change point and discontinuity interchangeably in this paper. For example, Figure 1 illustrates a time series with one change point when modelled with quadratic polynomials. In this case $P_1(t)$ is $t^2 + 10$, $P_2(t)$ is

$6t^2 - 1990$, and θ_1 is 20.

Throughout the paper, F is the class of polynomial functions of maximum degree D . That is, each $P_i(t)$ in Equation 1 can be represented as

$$P_i(t) = p_{i,D}t^D + p_{i,D-1}t^{D-1} + \dots + p_{i,2}t^2 + p_{i,1}t + p_{i,0}$$

We call a change point θ_i , a *change point of degree j* if the corresponding coefficients $p_{i,j}$ and $p_{i+1,j}$ differ in the polynomial representations of $P_i(t)$ and $P_{i+1}(t)$. Notice that θ_i is a change point of all degrees j where we have $p_{i,j} \neq p_{i+1,j}$. For example, $\theta_1 = 20$ is a change point of degrees 0 and 2 in Figure 1. This is because $p_{1,2} = 1$ and $p_{1,0} = 10$ while $p_{2,2} = 6$ and $p_{2,0} = -1990$.

3.1 Linear Algebra

In this section, we present some background linear algebraic definitions. We use these definitions in Section 4 when we discuss transforming time series with wavelet footprints.

Definition 1. A finite basis B for a vector space \mathbb{R}^d is a set of vectors $B_i \in \mathbb{R}^d$ (i.e., $B = \{B_1, B_2, \dots, B_n\}$ where $n = d$) such that any vector $V \in \mathbb{R}^d$ can be written as a linear combination of B_i 's, i.e., $V = \sum_{i=1}^n c_i B_i$. Note that given a basis B , the set of coefficients c_i is unique for a vector V . However, if the number of vectors in B is greater than d (i.e., $n > d$) then the vector V can be represented as the linear combination of B_i 's in infinite number of ways. We call such basis B where $n > d$ an *overcomplete* basis for \mathbb{R}^d .

Definition 2. Suppose $B = \{B_1, B_2, \dots, B_n\}$ is a finite basis for vector space \mathbb{R}^d and there exists a basis $\tilde{B} = \{\tilde{B}_1, \tilde{B}_2, \dots, \tilde{B}_n\}$ such that

$$\langle B_i, \tilde{B}_j \rangle = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \quad (2)$$

where $\langle X, Y \rangle$ denotes the inner product of vectors X and Y . The “*unique*” basis \tilde{B} is known as the *dual basis* of B .

Definition 3. A basis $B = \{B_1, B_2, \dots, B_n\}$ is a *biorthogonal* basis if we have $\langle B_i, \tilde{B}_j \rangle = 0$ for any $B_i \neq B_j$ and $\langle B_i, \tilde{B}_j \rangle = 1$ otherwise.

Definition 4. A basis $B = \{B_1, B_2, \dots, B_n\}$ is an *orthogonal* basis if for any $B_i \neq B_j$, we have $\langle B_i, B_j \rangle = 0$. According to Definition 2, an orthogonal basis is the dual basis of itself (i.e., it is *self-dual*).

To find each coefficient c_i where $1 \leq i \leq n$ for a vector V given a basis B (as in Definition 1), we simply compute $\langle V, \tilde{B}_i \rangle$. For orthogonal bases, due to their self-duality, c_i is computed by the inner product of V to the basis itself, i.e., $\langle V, B_i \rangle$.

The basic idea of compression is to find the basis B and then for each given vector V , only store the coefficients c_i 's. The main question is what is the best basis for a given application and dataset, such that several of c_i 's become zero or take negligible values. In our case, wavelet footprints would result in c_i 's that would take non-zero values only if a change occurs in the vector V . The value of c_i then corresponds to the amount of change.

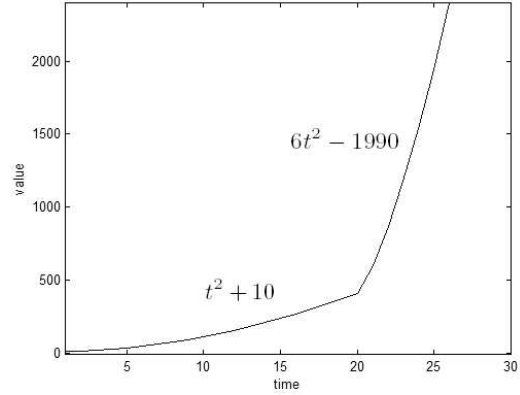


Figure 1: time series with one quadratic change point

3.2 Wavelets

We develop the background on the wavelet transformation using an example. We use Haar wavelets to transform our example time series into the wavelet domain. Although Haar wavelets are the simplest and oldest members of the wavelet family, they capture the essential elements of the wavelet theory. Consider the time series $\mathbf{X}_8 = (0, 0, 0, 0, 0, 1, 1, 1)$. The transformation process starts by computing the pairwise averages and differences of data. These computed values are multiplied by a normalization factor at each level ($\sqrt{2}$ for Haar) to produce two vectors of *summary* coefficients $H_1 = (0, 0, 0.7, 1.4)$ and *detail* coefficients $G_1 = (0, 0, -0.7, 0)$, respectively. This process continues by applying the same computation on the vector of summary coefficients to get $H_2 = (0, 1.5)$ and $G_2 = (0, -0.5)$. Finally, from H_2 we get $H_3 = (1.06)$ and $G_3 = (-1.06)$. The last summary coefficient (i.e., single element of H_3) followed by all $n - 1$ (7) detail coefficients computed during $\log n$ (3) iterations form the transformed data. The total number of iterations to get the last summary vector of size one (e.g., H_3) is called the level of decomposition. Figure 2 shows the complete transformation on \mathbf{X}_8 . Note that both the transformation from original representation to wavelet representation and re-transformation to the original presentation from wavelet view can be performed very efficiently in linear time.

We can conceptualize the process of wavelet transformation as the projection of the time series vector of size n to n different vectors ψ_i termed as *wavelet basis* vectors. Assume that $|X|$ represents the length of a vector X . Therefore, the wavelet transformation¹ of \mathbf{X}_n is $\hat{\mathbf{X}}_n = (\hat{x}_1, \dots, \hat{x}_n)$ where $\hat{x}_i = \langle \mathbf{X}_n, \psi_i \rangle \cdot \frac{1}{|\psi_i|}$ where the term $\frac{1}{|\psi_i|}$ is the normalization factor (notice that Haar wavelet basis is orthogonal, and hence self-dual). Moreover, time series \mathbf{X}_n can be represented as the linear combination:

$$\mathbf{X}_n = \sum_{i=1}^n \hat{x}_i \psi_i \quad (3)$$

¹Throughout the paper, we assume that the size of the time series we work with is always a power of 2. This can be achieved in practice by padding the time series with zeroes.

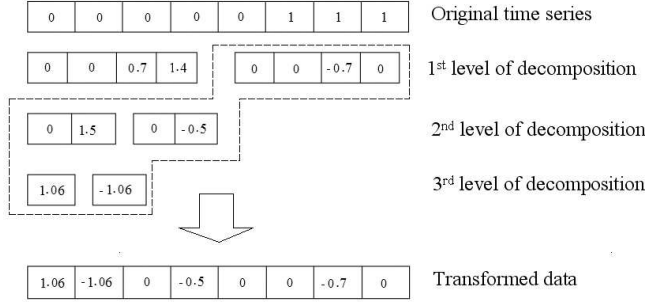


Figure 2: Wavelet example (traditional view of wavelets)

1	1	1	1	1	1	1	1
1	1	1	1	-1	-1	-1	-1
1	1	-1	-1	0	0	0	0
0	0	0	0	1	1	-1	-1
1	-1	0	0	0	0	0	0
0	0	1	-1	0	0	0	0
0	0	0	0	1	-1	0	0
0	0	0	0	0	0	1	-1

Figure 3: Haar wavelet basis of size 8

Figure 3 shows Haar wavelet basis vectors of size 8 as different rows of an 8×8 matrix.

In general, we identify Haar wavelet basis vectors of size n as ψ_i where $1 \leq i \leq n$. The first vector ψ_1 consists of n 1's. The remainder $n - 1$ vectors corresponding to the detail coefficients are defined as follows:

$$\psi_{2^j+k+1}(l) = \begin{cases} 1 & k \cdot \frac{N}{2^j} \leq l \leq k \cdot \frac{N}{2^j} + \frac{N}{2^{j+1}} - 1 \\ -1 & k \cdot \frac{N}{2^j} + \frac{N}{2^{j+1}} \leq l \leq k \cdot \frac{N}{2^j} + \frac{N}{2^j} - 1 \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

where $0 \leq j \leq \log n$ (j is the level of decomposition), $k = 0, \dots, 2^j - 1$, and $1 \leq l \leq n$.

We now define the term *support interval* that we will use throughout the paper.

Definition 5. Let $\hat{\mathbf{X}}_n = (\hat{x}_1, \dots, \hat{x}_n)$ be the wavelet transformation of the time series \mathbf{X}_n . The *support interval* of a wavelet coefficient \hat{x}_i is the range of indices $j \in [1, n]$ such that \hat{x}_i is derived from x_j 's, i.e., the value of \hat{x}_i is calculated from x_j 's.

For example, the support interval of the first coefficient \hat{x}_1 is the entire time series (i.e., $[1, n]$), while that of the last coefficient \hat{x}_n is the last two elements of \mathbf{X}_n (i.e., $[n - 1, n]$). We use $Sup(\hat{x}_i)$ to denote the support interval of coefficient \hat{x}_i . Similarly, we use $Sup^{-1}(j)$ to refer to the set of all wavelet coefficients which are derived from x_j , i.e., all \hat{x}_i 's such that $x_j \in Sup(\hat{x}_i)$.

4. FOOTPRINTS AT A GLANCE

Wavelets have been widely used in different data mining applications due to their power in capturing the trend of the data as well as their approximation property [17]. However, wavelets in their general form do not efficiently model

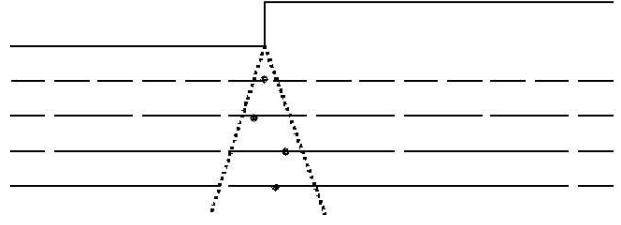


Figure 4: The top line shows the original data and the following lines are the wavelet representations at different levels. Notice how nonzero coefficients are scattered among different levels due to the overlapping support intervals.

discontinuities in the data. To illustrate the problem, consider the example of Figure 2. Although there exists only one discontinuity point at fifth position of our example time series \mathbf{X}_8 , we get three nonzero coefficients (other than the average) in the final transformed vector $\hat{\mathbf{X}}_8$. The reason is that there is a great amount of overlap among the support intervals of different coefficients in different levels. Figure 4 shows how wavelet transform scatters the effect of a single discontinuity point among wavelet coefficients of different levels. Therefore, to benefit from the approximation power of wavelets and efficiently model the change points in the underlying data at the same time, a new form of basis is required.

Dragotti et al. [3] introduce a new basis which removes the overlap among the support intervals of corresponding wavelet coefficients at different levels. They call this basis *wavelet footprints* or *footprints* for short, since footprints are basically the traces left by the discontinuities (i.e., singularities). We now explain the idea behind the footprints, assuming for simplicity piecewise constant data only for now. Consider \mathbf{X}_n with only one discontinuity at position θ :

$$\mathbf{X}_n(i) = \begin{cases} a & 1 \leq i \leq \theta \\ b & \theta < i \leq n \end{cases} \quad (5)$$

where a and b are two real values. In our example \mathbf{X}_8 , we have $a = 0$, $b = 1$, and $\theta = 5$.

We transform \mathbf{X}_n using Haar wavelet basis vectors ψ_i as:

$$\mathbf{X}_n = \hat{x}_1 \psi_1 + \sum_{i=2}^n \hat{x}_i \psi_i \quad (6)$$

where $\hat{x}_i = \langle \mathbf{X}_n, \psi_i \rangle$.

Considering the procedure of transformation discussed in Section 3.2, only those coefficients whose support interval include the point of discontinuity θ (i.e., x_θ) will be nonzero (i.e., \hat{x}_i if $i \in Sup^{-1}(\theta)$). In other word, we get

$$\mathbf{X}_n = \hat{x}_1 \psi_1 + \sum_{i \in Sup^{-1}(\theta)} \hat{x}_i \psi_i \quad (7)$$

Now, if we define a new vector f_θ as the linear combination of the multiplication of nonzero coefficients on their

corresponding wavelet basis vectors (i.e., the second term in Equation 7), we get a representation of \mathbf{X}_n as follows:

$$\mathbf{X}_n = \hat{x}_1 \psi_1 + \alpha_\theta f_\theta \quad (8)$$

where $\alpha_\theta = 1$.

We refer to f_θ as the footprint for discontinuity point x_θ of degree zero as it captures a change point of degree zero.

Here, we apply the same scenario to our example time series \mathbf{X}_8 . As Figure 3 shows, $\hat{\mathbf{X}}_8 = (1.0607, -1.0607, 0, -0.5, 0, 0, -0.7071, 0)$. If we rewrite \mathbf{X}_8 in terms of ψ_i 's we get $\mathbf{X}_8 = 1.0607 \frac{\psi_1}{|\psi_1|} - 1.0607 \frac{\psi_2}{|\psi_2|} - 0.5 \frac{\psi_4}{|\psi_4|} - 0.7071 \frac{\psi_7}{|\psi_7|}$. Let $f_5 = -1.0607 \frac{\psi_2}{|\psi_2|} - 0.5 \frac{\psi_4}{|\psi_4|} - 0.7071 \frac{\psi_7}{|\psi_7|}$. Similar to Equation 8 we get $\mathbf{X}_8 = 1.0607 \times \psi_1 + 1 \times f_5$. Therefore, we can represent \mathbf{X}_8 with one summary coefficient and only one nonzero detail coefficient at position 5.

Up to this point, we showed using an illustrative example that any piecewise constant time series data with one discontinuity can be represented using a footprint vector and the summary vector. Now, we need to show that the above scenario is extendable to time series data with m discontinuities of degree zero (piecewise constant). It is easy to see that any piecewise constant time series with m discontinuities can be represented as linear combination of m time series each with only one discontinuity. For example, the time series $(0, 0, 0, 1, 1, 0, 0)$ with two discontinuities at positions 4 and 6 is equal to the sum of $(0, 0, 0, 1, 1, 0, 0)$ and $(0, 0, 0, 0, 0, -1, -1)$ with one discontinuity each at positions 4 and 6, respectively. As the example shows, the position of the single discontinuity of each of these time series is the same as that of one and only one of the discontinuities of the original time series.

Now, we use Parseval's theorem [20] to extend the scenario we developed so far:

Theorem 1. Let $\hat{\mathbf{X}}$ denote wavelet transformation of vector \mathbf{X} using orthogonal wavelets (e.g., Haar). If $\mathbf{X}_n = \mathbf{X}_{1n} + \dots + \mathbf{X}_{mn}$, we have $\hat{\mathbf{X}}_n = \hat{\mathbf{X}}_{1n} + \dots + \hat{\mathbf{X}}_{mn}$.

Assume that the piecewise constant time series \mathbf{X}_n includes m discontinuities and each \mathbf{X}_{in} includes its i th discontinuity ($1 \leq i \leq m$) as we showed before. We also showed that one can represent each of these later time series \mathbf{X}_{in} in terms of a footprint f_i . The direct conclusion of Parseval's theorem is that $\hat{\mathbf{X}}_n$ is represented in terms of the set of summary vector and footprints f_i each used in representing \mathbf{X}_{in} . That is, any piecewise constant time series with m discontinuities can be represented with the summary vector together with m footprints.

As the previous example shows, we get a much sparser representation of the data if we use wavelet footprints as our basis instead of regular wavelets (e.g., Haar). We generalize this idea in the next section where we generate all vectors f_i of degree $d = 0, \dots, D$ each with only one discontinuity point i ($i = 1, \dots, n$) of degree d . Using these vectors as a basis enables us to capture polynomial changes up to degree D in time series data.

5. FOOTPRINTS

In Section 4, using the simple case of piecewise constant data we showed that wavelet footprints remove the dependency among the support intervals of wavelet coefficients. Therefore, they prevent from scattering information about

$$\begin{aligned} A^{(0)} &= 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ V_1^{(0)} &= 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ V_2^{(0)} &= 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ V_3^{(0)} &= 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ V_4^{(0)} &= 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ V_5^{(0)} &= 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ V_6^{(0)} &= 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ V_7^{(0)} &= 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{aligned}$$

Figure 5: Summary vector and discontinuity vectors of degree 0 and size 8

discontinuity points among different nonzero wavelet coefficients. Keeping this information in a single coefficient, any time series which is well approximated by piecewise polynomial functions can be represented with very few nonzero coefficients. In this section, we define footprints of arbitrary degrees to model changes of those degrees.

First, we define $V_\theta^{(d)}$, the θ th discontinuity vector of degree d and size n as

$$V_\theta^{(d)}(j) = \begin{cases} (j - \theta)^d & \theta < j \leq n \\ 0 & 1 \leq j \leq \theta \end{cases}$$

where $j \in [1, n]$, $\theta \in [1, n - 1]$. We also define $A^{(d)}$, the summary vector of degree d and size n , as

$$A^{(d)}(j) = j^d$$

where $j \in [1, n]$. Figure 5 shows discontinuity vectors and summary vector of degree zero. As illustrated discontinuity vectors contain all the possible discontinuities of degree zero.

Now, consider a time series \mathbf{X}_n in the class of piecewise polynomial time series with one discontinuity point of degree d at x_θ . \mathbf{X}_n can be represented using wavelet vectors as Equation 6. Similar to Section 4, we define footprints in terms of the θ th discontinuity vectors and summary vector and show that \mathbf{X}_n can be represented as the linear combination of footprints.

We introduce a new "fake" discontinuity point ² at x_0 . Hence, the nonzero values will now show up in coefficients whose support intervals include positions 0 or θ (i.e., \hat{x}_j where $j \in \text{Sup}^{-1}(0) \cup \text{Sup}^{-1}(\theta)$). For now, we assume that there is no commonality between these two sets of coefficients (i.e., $\text{Sup}^{-1}(0) \cap \text{Sup}^{-1}(\theta) = \emptyset$). Therefore, time series \mathbf{X}_n can now be written as

$$\mathbf{X}_n = \hat{x}_1 \psi_1 + \sum_{j \in \text{Sup}^{-1}(0)}^{j \neq 1} \hat{x}_j \psi_j + \sum_{j \in \text{Sup}^{-1}(\theta)}^{j \neq 1} \hat{x}_j \psi_j. \quad (9)$$

We transform summary and discontinuity vectors of degree d and size n to obtain:

²This is due to the periodic extension of wavelet basis. For the constant piecewise case, we were using Haar wavelets (with size 2) over time series of size 2^m . Therefore, we did not add any fake change point. However, for the general case we require to use padding (e.g., periodic) to transform the data. It is clear that this padding introduces a fake discontinuity at point zero of the time series.

$$A^{(d)} = \hat{a}_{1,d}\psi_1 + \sum_{j \in \text{Sup}^{-1}(0)}^{j \neq 1} \hat{a}_{j,d}\psi_j \quad (10)$$

and

$$V_\theta^{(d)} = \hat{v}_{1,d}\psi_1 + \sum_{j \in \text{Sup}^{-1}(0)}^{j \neq 1} \hat{v}_{j,d}\psi_j + \sum_{j \in \text{Sup}^{-1}(\theta)}^{j \neq 1} \hat{v}_{j,d}\psi_j \quad (11)$$

The set of $A^{(d)}$ and $V_\theta^{(d)}$ for all $d = 0, \dots, D$ constitutes a basis for the polynomial space of maximum degree D . Therefore, \mathbf{X}_n can be represented as the linear combination of the θ th discontinuity vectors and summary vectors:

$$\mathbf{X}_n = \sum_{d=0}^D \alpha_0^{(d)} A^{(d)} + \sum_{d=0}^D \alpha_\theta^{(d)} V_\theta^{(d)} \quad (12)$$

We combine Equations 9, 10, and 11, and considering only the coefficients \hat{x}_j where $j \in \text{Sup}^{-1}(\theta)$ we get

$$\sum_{j \in \text{Sup}^{-1}(\theta)}^{j \neq 1} \hat{x}_j \psi_j = \sum_{d=0}^D \alpha_\theta^{(d)} \sum_{j \in \text{Sup}^{-1}(\theta)}^{j \neq 1} \hat{v}_{j,d} \psi_j \quad (13)$$

In Equation 13, $\sum_{j \in \text{Sup}^{-1}(\theta)}^{j \neq 1} \hat{v}_{j,d} \psi_j$ contains all the coefficients generated by $V_\theta^{(d)}$ (i.e., all the coefficients generated by a discontinuity of degree d at point x_θ). We call this vector $f_\theta^{(d)}$.

We can generate all other $f_i^{(d)}$'s (for all $1 \leq i \leq n-1$ and $d \leq D$) in the same way. Dragotti et al. [3] prove that each discontinuity of degree D at x_i can be represented with the summary vector together with maximum $D+1$ footprints (i.e., $f_i^{(0)}, \dots, f_i^{(D)}$).

Finally, we can provide the precise definition of footprints:

Definition 6. Footprints are generated by gathering all the coefficients affected by discontinuity points in the θ th discontinuity vectors and summary vectors, normalizing them (i.e., $|f_i^{(d)}| = 1$), and enforcing bi-orthogonality (i.e., $\langle f_i^{(d)}, \tilde{f}_j^{(d)} \rangle = 0$ if $i \neq j$).

5.1 Properties

In this section, we enumerate different properties of footprints. These properties are our key motivations for using footprints in change point detection for time series data.

- The set of footprints together with the *summary vector* constitutes a basis. The reason is that a) each footprint vector f_i is a linear combination of the wavelet basis vectors ψ_j , and b) since each wavelet coefficient is affected at least once by a discontinuity point, each wavelet vector ψ_j contributes to at least one footprint vector f_i .
- The footprint basis is an overcomplete basis (except for the case of constant piecewise where we have footprints of degree 0 only). Notice that when the length of the data is n the number of footprint vectors in $f_i^{(D)}$ where $D > 0$ is $n \times (D+1)$, and hence the resulting basis is overcomplete.

- The footprints efficiently model discontinuities in time series data; a piecewise constant time series with K discontinuities, can be represented with only K footprints together with the summary coefficients. Footprints contain the following information about the discontinuity point generating them:
 1. The amplitude of the discontinuity.
 2. The characteristics of the two polynomials right before and after the discontinuity point.

1. The amplitude of the discontinuity.
2. The characteristics of the two polynomials right before and after the discontinuity point.

Also in piecewise polynomial case, time series with K discontinuities of maximum degree D can be represented with the summary coefficients and $K \times (D+1)$ footprint coefficients.

6. CHANGE DETECTION WITH FOOTPRINTS

We showed that nonzero coefficients in footprint transformed time series data are representatives for the change points in the data. Therefore, a novel change detection approach emerges by employing footprints. Throughout this section, we assume that we have pre-computed the biorthogonal footprint basis F_D (and \tilde{F}_D) in terms of the i th discontinuity vectors and the summary vector of degree up to D as we showed in Section 5. Note that this is a one time process independent of either the data or the queries.

We would like to answer two major categories of change detection queries. These queries can be issued on a single time series or on a database of time series data:

- **Q_d**: Return change points of all degrees.
- **Q_{da}**: Return change points of all degrees, their corresponding degrees and change amplitudes.

Similar to any general SQL query, user can enforce restrictions on degree of change point or its change amplitude. For example, user can ask for change points of degree d where the change amplitude is greater than a threshold T .

Our approach stores the time series data in the wavelet domain. That is, instead of the original data, its wavelet transformation is stored in the database. We use footprint basis F_D as our wavelet basis. Now, our approach answers change detection queries by returning the nonzero coefficients stored in its database. Figures 6a and 6b illustrate the process flow of our approach. We describe each part in details.

6.1 Insert/Update

Upon receiving the new data (i.e., time series), we transform it using F_D and then store it in the database. Since transforming data using footprints may be a time consuming task, we propose a *lazy* and an *absolute* method for the transformation in Sections 6.3.1 and 6.3.2, respectively. The lazy transformed data does not contain the exact information about the change amplitude and hence cannot be used to answer **Q_{da}** queries. However, the absolute transformed data can be processed to answer both **Q_d** and **Q_{da}** queries. To update the transformed data, approaches such as Shift-Split [9] can be used to update the transform data stored in the wavelet domain efficiently.

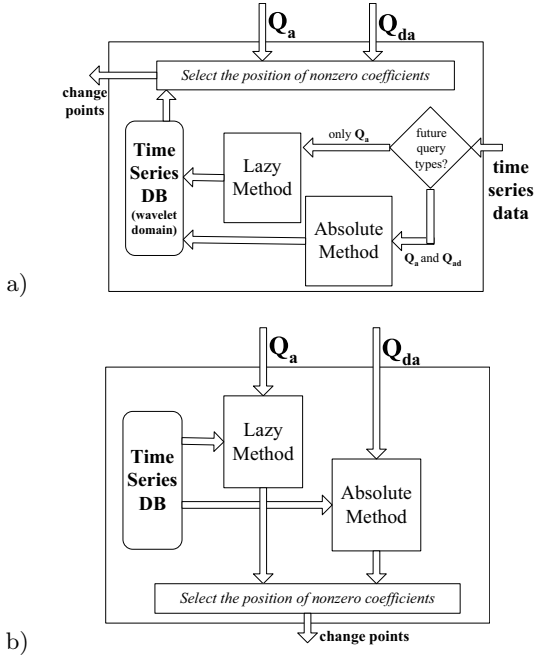


Figure 6: a) Query processing in wavelet domain and b) Ad hoc query processing

6.2 Query Processing

On receiving a change detection query on a specific portion of the data, we retrieve the nonzero coefficients corresponding to that portion of the data from the database. For each nonzero coefficient corresponding to footprint vector $f_i^{(d)}$, we only return a change of degree d at point i . If the user is interested in changes greater than a given threshold, we return the coefficients greater than that threshold. The time complexity of our approach to change detection is $O(n)$ for each time series of size n , since we only need a single scan over the data.

Trivially, the pre-transformation of the data eliminates the need to restart the entire process whenever user specifies a new degree and/or threshold for the change values. This makes our approach faster and more practical than the other change detection approaches where their entire algorithm is highly dependent on either threshold value or degree.

6.3 Footprint Transformation

The challenge here is to transform the data using the footprint basis. We propose two different methods for footprint transformation. The first *lazy* method is mainly based on *approximating* the footprint coefficients by projecting the time series on the dual basis of the footprint basis. This method is highly efficient in terms of performance but it is not tolerant to the existence of high-amplitude noise. Also the coefficients returned by this method are not the *exact* footprint coefficients due to the overlap among the footprint basis of different degrees.

The second *absolute* method is based on a greedy iterative algorithm termed *matching pursuit* [18] which is a proven approach in signal processing for representing signals in terms of an overcomplete basis. The outputs of both methods enable us to answer change detection queries by retrieving the nonzero coefficients and reporting their posi-

tions as change points. The absolute method has the extra advantage that it returns the amplitude of change as well since the coefficients are the exact footprint coefficients. Because of possible noise in the data, both methods may employ thresholds to select nonzero coefficients.

6.3.1 The Lazy Footprint Transformation

We assume that F_D , the footprint basis of degree up to D , and its dual basis \tilde{F}_D are pre-computed. Note that the computation of vectors of F_D is completely data-independent. We would like to find the change points in a given time series \mathbf{X}_n . The lazy method approximates the coefficients of \mathbf{X}_n by simply computing the $\alpha_i^{(d)} = \langle \mathbf{X}_n, \tilde{f}_i^{(d)} \rangle$ for all $f_i^{(d)}$'s in the basis. During the query processing, it returns i as a change point if $\alpha_i^{(d)}$ is greater than the user defined threshold in each footprint basis of degree d (see Section 6.2). The universal threshold $u = \sigma\sqrt{2\ln N}$ suggested in [4] is an appropriate candidate for the threshold value.

Since our basis is overcomplete, the dependency among footprint vectors can result into false hits and false negatives. To reduce the effect of this dependency, we make the footprint vectors corresponding to each discontinuity point k locally orthogonal by applying orthogonalization techniques such as Gram Schmidt [2] to $f_k^{(d)}$. Notice that the coefficients computed by the lazy method are not the exact footprint coefficients due to the overcompleteness of the basis. They only approximate the discontinuity points. This is the reason that the lazy transformed data cannot be used to answer Q_{da} queries. However, in Section 8 we show that the lazy transformation performs very effectively for detecting the change points.

For each time series of size n , the time complexity of the lazy transformation is $O(n^2)$. The reason is that this method requires projecting \mathbf{X}_n on each of the vectors of footprint basis F_D . The number of these vectors is $O(n)$ where n is the size of the time series data.

6.3.2 The Absolute Footprint Transformation

As mentioned in Section 5.1, the footprint vectors constitute an overcomplete basis. This overcomplete basis gives us more power and flexibility in modelling changes in the data. As a drawback, transformation of the time series \mathbf{X}_n (i.e., coefficients $\alpha_i^{(d)}$) becomes a more challenging task. Here, in order to compute the exact values for $\alpha_i^{(d)}$'s we use the *matching pursuit technique* to find the nonzero $\alpha_i^{(d)}$ coefficients. The following iterative procedure computes the coefficients for time series \mathbf{X}_n :

1. Find the projection of \mathbf{X}_n on all vectors $f_i^{(d)}$ in the footprint basis, i.e., compute all the coefficients α_i such that $\alpha_i = \langle \mathbf{X}_n, f_i^{(d)} \rangle$.
2. Let $\alpha_j = \max(\alpha_i)$; the coefficient corresponding to the highest energy component of \mathbf{X}_n .
3. Let $\mathbf{X}'_n = \mathbf{X}_n - \sum_{d=0}^D \langle \mathbf{X}_n, f_j^{(d)} \rangle \cdot f_j^{(d)}$ where $f_j^{(d)}$ is the footprint corresponding to α_j .
4. If all the values in \mathbf{X}'_n are zero (or less than a predefined threshold in case of noisy data) exit the procedure.
5. Let $\mathbf{X}_n = \mathbf{X}'_n$ and goto 1.

The set of coefficients selected by step 2 of the above procedure form the nonzero coefficients of the transformation of time series \mathbf{X}_n using footprint basis of degree d . Therefore, their corresponding positions are change points of degree d in the time series \mathbf{X}_n . Notice that the coefficients computed by the absolute approach are the exact coefficients of the footprint transformation. We can modify matching pursuit such that the algorithm terminates after maximum $\lceil \frac{K}{2} \rceil$ iterations where K is the number of change points in \mathbf{X}_n . Hence, the overall time complexity of the absolute transformation becomes $O(\lceil \frac{K}{2} \rceil \cdot n^2)$.

An important advantage of the absolute method over the lazy method is that with the former we get a) all the exact change points of degrees 0 to D , and b) the exact amplitude of the change at each change point. This enables us to answer both \mathbf{Q}_d and \mathbf{Q}_{da} queries which focus on the amplitude and degree of change at each point. For example, the absolute method can return the change points higher in value than 10 with the change of degree 3 in a given time series.

6.4 Ad Hoc Query Processing

If the data cannot be stored in the wavelet domain, we must transform it in real-time when we receive a query. We choose between the lazy or absolute method based on the type of query. The time complexity of this ad hoc change detection approach is equal to the time complexity of the transformation using footprints which is $O(n^2)$ (see Sections 6.3.1 and 6.3.2).

7. CUSTOMIZING FOOTPRINTS FOR WAVELET-BASED APPLICATIONS

In the previous sections, we developed a novel approach for change detection based on the transformation of the data using wavelet footprints. In this section we show that our approach can be incorporated within systems where the time series data is maintained in the wavelet domains (e.g., ProDA [10]). That is, instead of the original data, its wavelet transformation is stored.

An example of an approach dealing with the data directly in the wavelet domain is ProPolyne introduced in [22]. ProPolyne is a wavelet-based technique for answering polynomial range-aggregate queries. It uses the transformed data from the wavelet domain to generate the result. We show that ProPolyne’s approach to answer polynomial range-aggregate queries is still feasible when the data is transformed using footprint wavelet basis instead of general wavelet bases such as Haar.

With ProPolyne, a polynomial range-aggregate query (e.g., SUM, AVERAGE, or VARIANCE) is represented as a query vector Q_n . Then, the answer to the query is $\langle \mathbf{X}_n, Q_n \rangle$. The family of wavelet basis used by ProPolyne each constitutes an orthogonal basis. Thus, according to *Parseval’s theorem*, they preserve the energy of the data after the transformation and hence we have:

$$\langle \mathbf{X}_n, Q_n \rangle = \langle \hat{\mathbf{X}}_n, \hat{Q}_n \rangle. \quad (14)$$

Therefore, ProPolyne evaluates the inner product of $\hat{\mathbf{X}}_n$ and \hat{Q}_n as the answer to the query Q_n .

Now, we show that if we transform the data using the footprint basis F_D and the query using the dual basis of F_D , we

are able to answer polynomial range-aggregate queries proposed by ProPolyne. Therefore, ProPolyne can transform data using footprint basis and still benefit from its unique properties.

It is easy to see that Equation 14 does not hold for wavelet footprints since the footprint basis is not an orthogonal basis. Here, we extend Equation 14 to hold for the biorthogonal bases. Assume that $\hat{\mathbf{X}}_n$ and $\tilde{\mathbf{X}}_n$ are the transformations of \mathbf{X}_n using the footprint wavelet basis, and its dual basis, respectively. Now, according to Definitions 3 and 4, and Equation 14 it is easy to see that

$$\langle \mathbf{X}_n, Q_n \rangle = \langle \hat{\mathbf{X}}_n, \tilde{Q}_n \rangle \quad (15)$$

$$\langle \mathbf{X}_n, Q_n \rangle = \langle \tilde{\mathbf{X}}_n, \hat{Q}_n \rangle \quad (16)$$

In practice, we use Equation 15 where the data is transformed to wavelet in advance and the dual of the query will be computed on the fly to perform the dot product at the query time.

8. EXPERIMENTAL RESULTS

We conducted several experiments to evaluate the performance of our proposed approach for change detection in time series data. We compared the query response time of our ad hoc query processing approach described in Section 6.4 with the maximum likelihood-based algorithm proposed by Guralnik et. al [8]. We chose their approach for comparison because it is the fastest change detection algorithm that considers different degrees of change. Throughout this section, we refer to their method as the *Likelihood* method.

We studied how the size of the time series (n) and the total number of its change points (K) affects the performance of each method.

We also evaluated our lazy and absolute methods by investigating the effect of the following parameters on their accuracy: 1) the minimum distance between two consecutive change points (MinDist), 2) the maximum degree of change points in the data (MaxDeg), 3) the maximum degree D of footprint basis (MaxDegF), and 4) the amount of noise in the data (Noise). We represent the accuracy of each method in terms of the average number of missed change points (false negatives AFN) and detected spurious change points (false hits AFH).

We used both synthetic and real-world datasets. We generated a synthetic dataset **D3** of 80 time series each with a size in the range of 100 to 5,000. Each time series of the dataset **D3** is a concatenation of several segments each modelled by a polynomial of degree up to 3. The average number of change points in each time series is 10. Our real-world dataset include oil and pressure time series for different oil wells in California.

The experiments were performed using MATLAB on a DELL Precision 470 with Xeon 3.2 GHz processor and 2GB of RAM. Notice that for the absolute method, we used a modified version of matching pursuit introduced in [3] which is guaranteed to terminate after $\lceil \frac{K}{2} \rceil$ iterations where K is the number of change points. For the likelihood method, we use the threshold value using which the method computes the most accurate result. Sections 8.1 and 8.2 focus on our synthetic dataset as we already know the exact characteristics of the change in their time series. This enables us to measure the accuracy of our approach. Section 8.3 discusses our experiments with the real-world data.

8.1 Performance

In our first set of experiments, we compared the performance of our ad hoc change detection query processing using both lazy and absolute methods with the Likelihood method. As the Likelihood method uses the original time series data as input, we must compare it only with our ad hoc approach where the footprint transformation takes place at the query time. That is, the CPU time reported for the lazy and absolute methods include both the time for the footprint transformation of data and that of detecting the change points. We used footprint basis of up to degree 3 to transform the data in the lazy and absolute methods (i.e., $f^{(0)}$, $f^{(1)}$, $f^{(2)}$ and $f^{(3)}$). That is, $\text{MaxDeg} = \text{MaxDegF}$. Also, we used polynomials of up to degree 3 for finding the change points in the Likelihood method.

We varied the size of time series data from 100 to 5,000 and measured the CPU cost of each method. Figure 7 depicts the performance of our lazy and absolute methods as compared to that of the Likelihood method. In the figure, $\text{Lazy}(i)$ denotes the measurements of the lazy method in which the threshold value is $i \times u$ (u is the universal threshold and i is simply a factor multiplied by u). As shown in the figure, our lazy and absolute methods outperform the Likelihood method by a factor of 2 to 8 when the size of data increases from 100 to 5,000. It also shows that the lazy method is faster than the absolute method because it trades the performance for accuracy.

Note that we showed in Sections 6.3.1 and 6.3.2 that the time complexity of our methods is $O(n^2)$ if we transform the time series at the query time. Moreover, the time complexity of the Likelihood method is also $O(n^2)$ [8]. However, as our first experiment shows, our methods perform practically better than the Likelihood method in terms of CPU cost.

The theoretical time complexities of the absolute and Likelihood methods depend on K , the number of change points in the data. On the other hand, the lazy method is a series of simple projections which are independent from the characteristics of the data. To study the effect of K on the performance of each method, we varied K from 0 to 42 on time series of size 2,048 and measured the CPU cost. Figure 8 illustrates that while the performance of both our methods remains almost fixed for different number of change points, the CPU cost of Likelihood method dramatically increases when the number of change points in data increases. The intuition here is that the number of Likelihood’s iterations to fit the functions on the segments of data is identical to the number of change points and each iteration performs expensive computations. This is while the computation time of our lazy method only depends on the size of data (n). As shown in the figure, the CPU cost of lazy method is fixed for different values of K . The absolute method performs slightly slower when K increases but its performance downgrade is negligible. The figure does not show the CPU cost of the Likelihood method for $K > 25$ but the method computes 42 change points in 1.7 seconds, almost 68 times slower than our methods.

Notice that even when there is no change in the data (i.e., $K = 0$), our methods are performing at least twice better than the Likelihood method. The reason is that the Likelihood method still requires to apply one regression iteration on the data. To conclude, we showed that unlike the likelihood method both our methods are scalable with respect to the number of change points in the data.

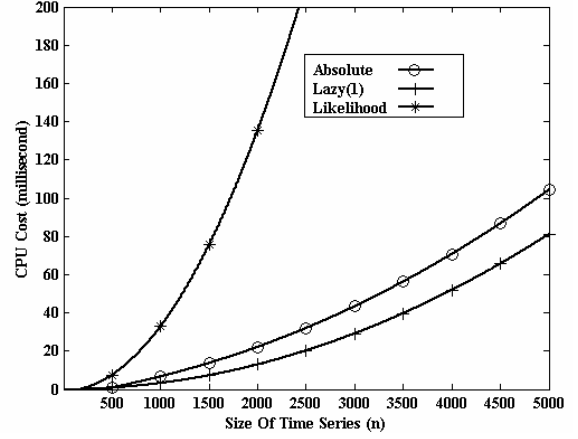


Figure 7: Query cost vs. size of data (n)

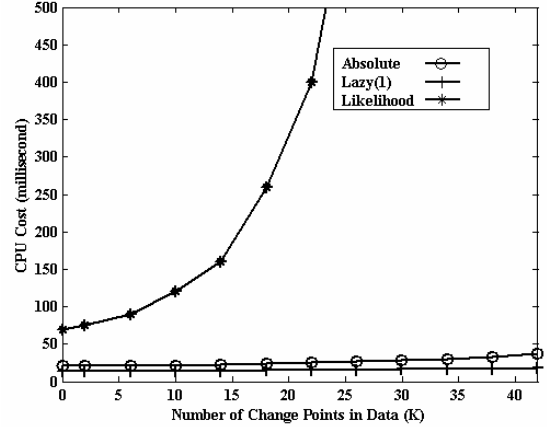


Figure 8: Query cost vs. number of change points (K)

8.2 Accuracy

Our second set of experiments were aimed to evaluate the accuracy of each method in terms of number of missed change points and the detected spurious change points (i.e., precision and recall). First, we illustrate how each method performs on an example time series data. Figure 9 shows a small time series of size 200 generated with polynomial segments of maximum degree 2. It also shows the true change points and those detected by each method. We used footprint basis of up to degree 2 to transform the data in the lazy and absolute methods (i.e., $f^{(0)}$, $f^{(1)}$ and $f^{(2)}$). Also for the Likelihood method we used polynomials of up to degree 2 for finding the change points.

There are ten actual change points as shown in Figure 9 and the minimum distance between each two change points is 20. The likelihood and lazy(1.5) methods both miss the change point at $t = 120$. Also, Likelihood method detects two false hits at points $t = 51$ and $t = 90$. Also for the change that occurs at point $t = 100$, it detects two change points at $t = 98$ and $t = 101$. The lazy method returns no false hit. The absolute method returns all 10 actual change

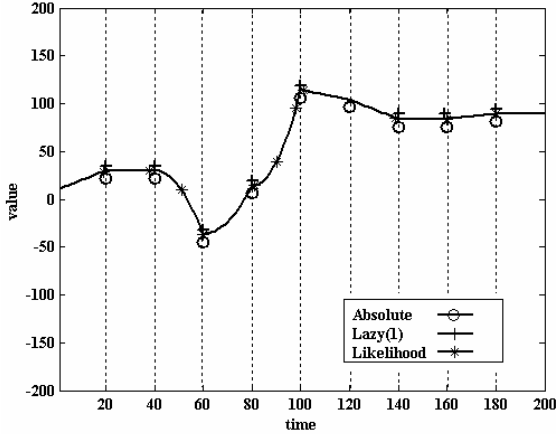


Figure 9: Detected change points by the absolute, lazy and Likelihood methods. The vertical lines show the actual change points in the data.

points at their exact positions without any false hit (i.e., 100% precision and recall).

Notice that using our footprint-based approach, we also acquire valuable information about the degree and amplitude of each change point. For example, at point $t = 40$, we have a discontinuity caused by a quadratic segment following a constant segment. Hence, we get large coefficients for $f_{40}^{(0)}$ and $f_{40}^{(2)}$. Also at point $t = 140$, we have a discontinuity caused by a constant segment following a linear segment. Hence, we get large coefficients for $f_{140}^{(0)}$ and $f_{140}^{(1)}$ and zero for $f_{140}^{(2)}$.

We repeated the previous experiment on all time series of the dataset **D3** for which $K = 10$. We varied the minimum distance between each two consecutive change points in the data (MinDist) from 5 to 50. Table 1 shows the average number of false negatives (AFN) for the lazy, absolute and Likelihood methods. Column **F3** shows the results for the experiment where we used footprints of degrees 0, 1, 2, and 3 for the lazy and absolute methods. That is, we used $F_3 = \{f^{(0)}, f^{(1)}, f^{(2)}, f^{(3)}\}$. Note that in this case MaxDegF is identical to MaxDeg. Likewise, column **F2** shows the case where we used footprints of degrees 0, 1, and 2. For this case, MaxDegF < MaxDeg. That is, we used footprint basis of a degree less than the degree of change points in the data. The Likelihood method also used polynomial functions of degrees up to 2 (resp. 3) for case **F2** (resp. **F3**).

8.2.1 The effect of MinDist

As Table 1 depicts, both lazy and absolute methods always outperform the Likelihood approach in terms of accuracy with absolute being the superior method. With the small values of MinDist, the accuracies of all methods dramatically downgrade. That is, if the change points are too close to each other, all methods are unable to detect some of the true change points. However, even for close change points, the absolute method misses only one of 10 change points on average. This yields that the absolute method is resilient to the effect of closeness of change points. As the distances between change points are increasing, the accuracy of all methods improves. When MinDist is 20, the absolute method correctly detects all change points when MaxDeg =

MaxDegF (i.e., case **F3**). Note that in this experiment, the absolute method detected no false change points (i.e., AFH = 0). However, AFH = 0.5 for the Likelihood method and AFH = 0.4 for our lazy method for all the values of MinDist. This shows that while all methods are performing accurately for large enough MinDist values, our absolute method can effectively detect even very close change points.

8.2.2 The effect of MaxDeg and MaxDegF

In this section, we investigate the accuracies of all the three methods for the cases where the degree of footprints in our methods or that of polynomials in the Likelihood method (MaxDegF) is less than, equal, or greater than the maximum degree of change in data (MaxDeg). We focus on the results shown in Table 1 for MinDist = 20. It is clear that when MaxDegF is less than MaxDeg (i.e., case **F2**), all methods miss more true change points as compared to the case **F3** where the footprints and polynomials of appropriate degrees have been used (i.e., MaxDegF = MaxDeg). However, the average number of false hits of each method does not change.

We repeated our experiments on only those time series of **D3** which consist of polynomial segments of up to degree 2. We used footprint basis F_3 and polynomials of degree up to 3 in the methods. While we do not report the detailed results for this case where MaxDegF > MaxDeg, we report that no method generated false hits in this case. It shows that using footprint basis of a degree greater than the maximum degree of change in the data does not result into detecting false change points by our methods. Therefore, if we guarantee that the degree D of our footprint basis is large enough, our approach is able to accurately detect all the change points in the data.

8.2.3 The effect of Noise

In our third set of experiments, we studied the effect of noise on the accuracy of the change detection. Here, we fixed the minimum distance between change points to 10, threshold value equal to $1.5 \times u$, and $F_D = F_2$. Using polynomials of degree up to 2, we generated two noisy datasets. We added noise with the standard deviation of about 1/15 to 1/30 and 1/150 to 1/300 of the average of values in time series to generate two noisy datasets **Noisy(1)** and **Noisy(0.1)**, respectively. The number of change points in each time series is 10. Table 2 shows the accuracy results of applying all three methods on both datasets.

Trivially, the presence of noise reduces the performance of all methods. Our lazy method which detects the change points based on approximations of the footprint transformation of the data starts missing some of the true change points in the presence of noise. On the other hand, the Likelihood method generates more false hits in this case. However, our absolute method still generates the most accurate results for different amount of noise in data.

8.3 Experiment With Real-World Datasets

Finally, the last set of experiments focuses on real-world time series data. We tested our methods on different time series generated within the oil industry. Here, we report the results on three time series **OIL1**, **OIL2**, and **GAS** obtained from Petroleum Technology Transfer Council³. These time series are collected from wells in active oil fields in

³<http://www.westcoastpttc.org/>

F3		F2	
Method	AFN	Method	AFN
MinDist= 5		MinDist= 5	
Lazy(1.5)	3	Lazy(1.5)	3.5
Lazy(1)	2.1	Lazy(1)	3.1
Absolute	0.9	Absolute	1
Likelihood	4.5	Likelihood	4.1
MinDist= 10		MinDist= 10	
Lazy(1.5)	1.9	Lazy(1.5)	2.9
Lazy(1)	0.9	Lazy(1)	2.7
Absolute	0.2	Absolute	0.5
Likelihood	1.8	Likelihood	3.0
MinDist= 20		MinDist= 20	
Lazy(1.5)	0.5	Lazy(1.5)	1.1
Lazy(1)	0.2	Lazy(1)	0.6
Absolute	0	Absolute	0.1
Likelihood	0.2	Likelihood	1
MinDist= 50		MinDist= 50	
Lazy(1.5)	0.3	Lazy(1.5)	0.9
Lazy(1)	0.2	Lazy(1)	0.4
Absolute	0	Absolute	0.1
Likelihood	0.2	Likelihood	0.8

Table 1: Accuracy results of all methods for cases F2 and F3

Noisy(1)			Noisy(0.1)		
Method	AFN	AFH	Method	AFN	AFH
Lazy(1)	5	1.2	Lazy(1)	3.5	1
Absolute	2	0.9	Absolute	1.5	0.8
Likelihood	2.8	3	likelihood	2.6	3

Table 2: Accuracy results of all methods for datasets Noisy(1) and Noisy(0.1)

California. **OIL1** and **OIL2** include oil production during 1985-1995 and 1974-2002, respectively. **GAS** includes the gas production rate measured in a 2300 day period, sampling once every 15 days.

Unlike synthetic time series, here we do not know where the *exact* change points are. Therefore, we evaluated our methods visually based on the position of their detected change points. Figure 10 depicts the change points detected in time series **OIL1** by the absolute, lazy and Likelihood method. While both our methods perform absolutely perfect on this data, the Likelihood method ignores 60% of the change points. For example, there are local minimum and maximum points at $t = 28$ and $t = 57$, respectively. The Likelihood method misses these points. Moreover, it returns several false points such as $t = 29$ and $t = 47$. Figures 11 and 12 show similar results for time series **OIL2** and **GAS**, respectively.

Notice that our absolute method does not identify any change at points such as $t = 235$ in Figure 11. The reason here is that the segment corresponding to the range $[228, 240]$ can be perfectly modelled by a polynomial of degree 3. Therefore, $t = 235$ is not a discontinuity of degree 3 in **OIL2**.

Figure 13 illustrates the detected change points in **GAS** data by each of three methods when they use higher threshold values as compared to those used in Figure 12. Comparing Figures 12 and 13 shows that the former detects all small changes while the later identifies only major changes in the data. Notice that once our methods detect changes using

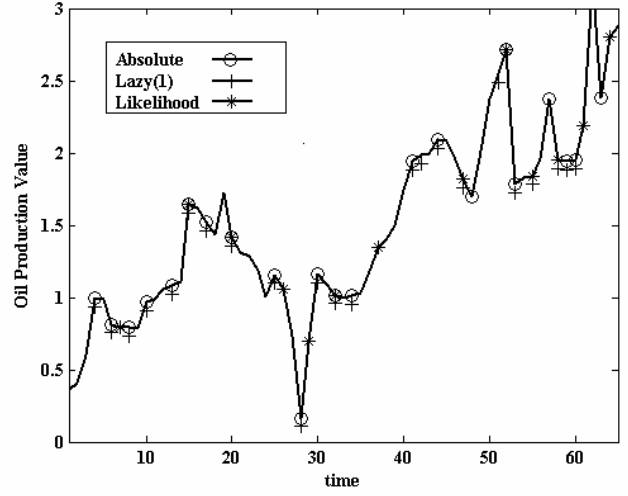


Figure 10: Detected change points by the absolute, lazy and Likelihood methods on OIL1

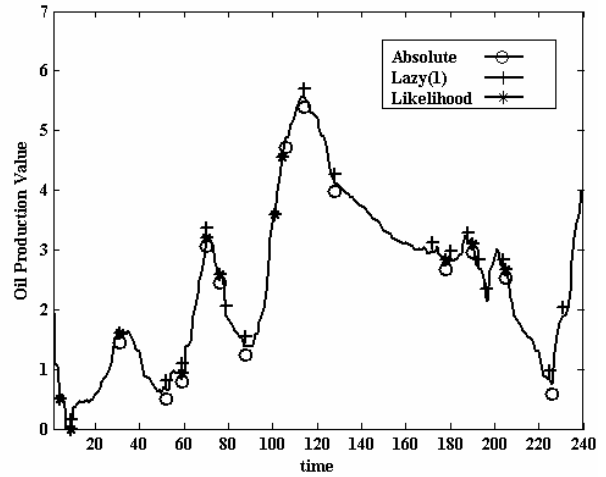


Figure 11: Detected change points by the absolute, lazy and Likelihood methods on OIL2

a given threshold value, changes above different thresholds can be identified only by doing a scan over all coefficients. However, with the likelihood method, we need to rerun the whole process when the user changes her threshold value. For example, we ran the likelihood method separately for the results of each of Figures 12 and 13. For our methods, we only ran our approach once to generate the results of Figure 12 and only checked all the coefficients for the different thresholds of Figure 13.

9. CONCLUSIONS AND FUTURE WORK

We studied the problem of detecting changes in time series data. Using a real-world motivating application, we showed that different scientific data analysis tasks might have different definitions for change. Therefore, we formally defined the degree of change for change points in the time series data. Our definition is closely related to the difference in two

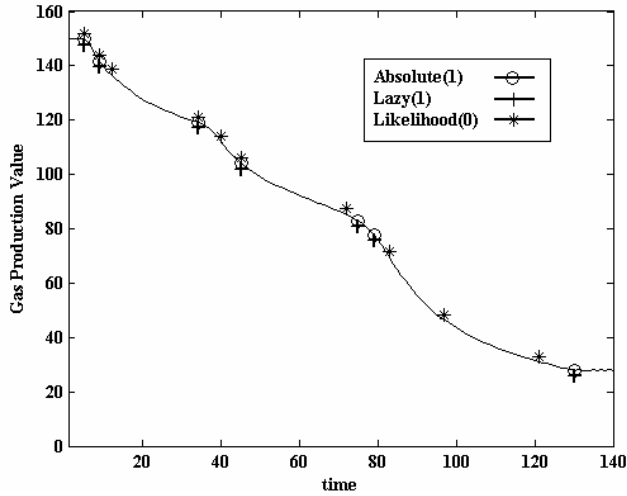


Figure 12: Detected change points by the absolute, lazy and Likelihood methods on GAS

polynomial functions fitting the two adjacent segments of data. We then described our novel wavelet-based approach which employs wavelet footprints for defining discontinuities of different degrees.

First, we provided preliminary background on footprints and showed their property of capturing discontinuities in the time series data. We showed that the footprint transformation of the time series data using footprints of degree d contains nonzero coefficients only at the change points of degree d . Subsequently, we proposed our approach for change detection in time series data describing its data transformation and query processing modules. We proposed lazy and absolute methods to transform the data using footprint basis. Our lazy method approximates the coefficients based on which change points can be identified efficiently. However, our absolute method computes the accurate coefficients and benefits from the ability to provide the amplitude of the change at each change point. We also showed that our approach can be efficiently incorporated within the systems such as ProDA [10] where the time series data is stored in the wavelet domain.

Finally, we compared the performance and accuracy of our footprint-based approach with the maximum likelihood method [8] through exhaustive sets of experiments with both synthetic and real-world data. Our empirical results showed the followings:

- Our ad hoc query processing approach with both lazy and absolute methods outperforms the maximum likelihood method specially for large time series data. Furthermore, while the performance of maximum likelihood method dramatically downgrades with increasing number of change points in the data, both of our methods are scalable with respect to this factor. This is a significant improvement over the best known method specially for real-world applications where the characteristics of the data is not known a priori.
- Our change detection approach is highly accurate. Even the lazy method which approximates the change points performs as accurate as the maximum likelihood method.

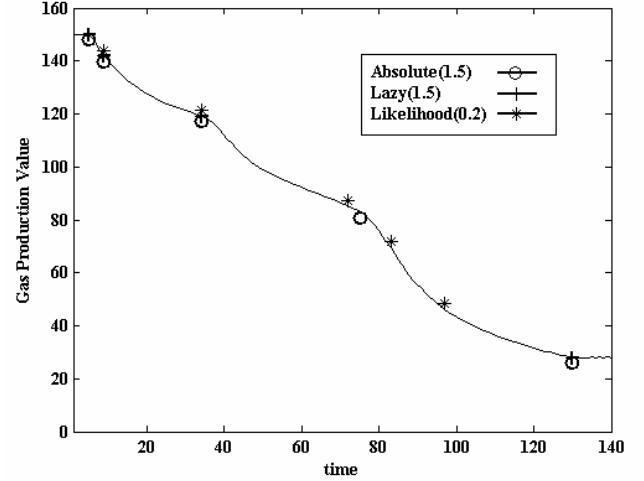


Figure 13: Detected change points by the absolute, lazy and Likelihood methods on GAS with threshold values $1.5 \times u$ for the absolute and lazy methods and $0.2 \times u$ for the likelihood method

The absolute method detects exactly all discontinuities and computes their amplitudes and degree of change when the data is noiseless. Even with the noise in data, it is considerably more precise than the other methods.

In this paper, for the first time we exploited the interesting characteristics of footprints for change detection in time series data. Motivated by our results, we plan to develop a footprint-based tool for real-time change detection on data streams.

10. ACKNOWLEDGEMENT

We would like to acknowledge Dr. Antonio Ortega and his student En-Shuo Tsau for their valuable help in our understanding of the theory of wavelet footprints.

This research has been funded in part by NSF grants EEC-9529152 (IMSC ERC) and IIS-0238560 (PECASE), unrestricted cash gifts from Microsoft, an on-going collaboration under NASA's GENESIS-II REASON project and partly funded by the Center of Excellence for Research and Academic Training on Interactive Smart Oilfield Technologies (CiSoft); CiSoft is a joint University of Southern California - ChevronTexaco initiative.

11. REFERENCES

- [1] K. Chakrabarti, M. N. Garofalakis, R. Rastogi, and K. Shim. Approximate query processing using wavelets. In A. E. Abbadi, M. L. Brodie, S. Chakravarthy, U. Dayal, N. Kamel, G. Schlageter, and K.-Y. Whang, editors, *VLDB 2000, Proceedings of 26th International Conference on Very Large Data Bases, September 10-14, 2000, Cairo, Egypt*, pages 111–122. Morgan Kaufmann, 2000.
- [2] H. Cohen. *A course in computational algebraic number theory*. Springer-Verlag New York, Inc., 1993.
- [3] P. L. Dragotti and M. Vetterli. Wavelet transform footprints: Catching singularities for compression and denoising. In *ICIP*, 2000.

- [4] P. L. Dragotti and M. Vetterli. Deconvolution with wavelet footprints for ill-posed inverse problems. In *IEEE Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 1257–1260, Orlando, Florida, USA, May 2002.
- [5] P. L. Dragotti and M. Vetterli. Wavelet footprints: Theory, algorithms and applications. *IEEE Transactions on Signal Processing*, 51(5):1306–1323, May 2003.
- [6] L. Firoiu and P. R. Cohen. Segmenting time series with a hybrid neural networks - hidden markov model. In *Eighteenth national conference on Artificial intelligence*, pages 247–252. American Association for Artificial Intelligence, 2002.
- [7] X. Ge. *Segmental semi-markov models and applications to sequence analysis*. PhD thesis, 2002. Chair-Padhraic Smyth.
- [8] V. Guralnik and J. Srivastava. Event detection from time series data. In *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 33–42. ACM Press, 1999.
- [9] M. Jahangiri, D. Sacharidis, and C. Shahabi. SHIFT-SPLIT: I/O Efficient Maintenance of Wavelet-Transformed Multidimensional Data. In *Proceedings of the 24th ACM SIGMOD International Conference on Management of Data*, 2005.
- [10] M. Jahangiri and C. Shahabi. ProDA: A Suit of WebServices for Progressive Data Analysis. In *Proceedings of 24th ACM SIGMOD International Conference on Management of Data*, 2005. (demonstration).
- [11] R. Jr. *Advances in Well Test Analysis*, volume 5. Society of Petroleum Engineers, 1977.
- [12] E. Keogh and P. Smyth. A probabilistic approach to fast pattern matching in time series databases. In D. Heckerman, H. Mannila, D. Pregibon, and R. Uthurusamy, editors, *Third International Conference on Knowledge Discovery and Data Mining*, pages 24–30, Newport Beach, CA, USA, 1997. AAAI Press, Menlo Park, California.
- [13] E. J. Keogh, K. Chakrabarti, S. Mehrotra, and M. J. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. In *SIGMOD Conference*, 2001.
- [14] E. J. Keogh, S. Chu, D. Hart, and M. J. Pazzani. An online algorithm for segmenting time series. In *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 289–296. IEEE Computer Society, 2001.
- [15] J. Lin, E. Keogh, and W. Truppel. Clustering of streaming time series is meaningless. In *DMKD '03: Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, pages 56–65. ACM Press, 2003.
- [16] J. Lin, M. Vlachos, E. J. Keogh, and D. Gunopulos. Iterative incremental clustering of time series. In *EDBT '04: Proceedings of the 8th International Conference on Extending Database Technology*, pages 106–122, 2004.
- [17] S. Mallat and W. L. Hwang. Singularity detection and processing with wavelets. *IEEE Trans. Inf. Th.*, 38:617–643, 1992.
- [18] S. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.
- [19] Y. Matias, J. S. Vitter, and M. Wang. Wavelet-based histograms for selectivity estimation. In *SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 448–459. ACM Press, 1998.
- [20] A. V. Oppenheim and R. W. Schaffer. *Digital Signal Processing*. Prentice Hall, Englewood Cliffs, NJ, USA, 1975.
- [21] V. Puttagunta and K. Kalpakis. Adaptive methods for activity monitoring of streaming data. In *ICMLA*, pages 197–203, 2002.
- [22] R. R. Schmidt and C. Shahabi. Propolyne: A fast wavelet-based algorithm for progressive evaluation of polynomial range-sum queries. In *EDBT '02: Proceedings of the 8th International Conference on Extending Database Technology*, pages 664–681. Springer-Verlag, 2002.
- [23] K. Yamanishi and J. ichi Takeuchi. A unifying framework for detecting outliers and change points from non-stationary time series data. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 676–681. ACM Press, 2002.