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. In this paper, we propose a novel approach to address the problem of change detection in time series data. 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 capture discontinuities in a signal. We show that transforming data using footprints generates nonzero coefficients only at the change points. Exploiting this property, we propose a change detection query processing scheme which employs footprint-transformed data to identify change points, their amplitudes, and degrees of change efficiently and accurately. Our analytical and empirical results 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 of the number of change points 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 of 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 [7,8,12,21,24]. 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.

C. Bauzer Medeiros et al. (Eds.): SSTD 2005, LNCS 3633, pp. 127–144, 2005.

For the past year, we have been working with Chevron 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), 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, these points 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 the *degrees* of change 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 ith 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 tool 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. 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. Our query response time is independent of the number of change points in the data. This is while both our methods demonstrate a significant increase in accuracy.

The remainder of this 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 first illustrate the idea of using footprints to capture discontinuities by focusing only on piecewise linear time series. We then generalize our change detection approach and propose our lazy and absolute methods for footprint transformation in Section 5. In Section 6, we show how our footprint-based approach can be incorporated within systems where time series data is stored in wavelet domain. Section 7 includes our experimental results, and Section 8 discusses the conclusion and our future plans.

2 Related Work

Change detection in time series has recently 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 endpoints of the segments as change points. 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. [24] reduce the problem of change point detection in time series into that of outlier detection from time series of moving-averaged scores. Ge et al. [7] extend hidden semi markov model for change detection. Both these solutions are applicable to different data distributions using different regression functions; however, they are not scalable to large size datasets due to their 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, the user can focus only on the change points detected by the 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 $(x_1, ..., x_n)$ 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 $\mathcal{X} : [1, n] \to \mathbb{R}$ that models time series $\mathbf{X}_n$ as follows:

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

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 segment with P_i. Our ultimate goal is to identify $\theta_1, ..., \theta_K$ when 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.

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

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

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 $p_{i,j} \neq p_{i+1,j}$.

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., $B = \{B_1, B_2, ..., B_n\}$ 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. Such collection B where $n > d$ is an overcomplete collection for $\mathbb{R}^d$.For ease of use we call it an "overcomplete basis" for the rest of the paper.*

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

$$\langle B_i, \tilde{B}_j \rangle = \begin{cases} 1 \; if \; i = j \\ 0 \; if \; i \neq j \,. \end{cases} \tag{3}$$

*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 $B = \{B_1, B_2, ..., B_n\}$ 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 $B = \{B_1, B_2, ..., B_n\}$ 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 (or amplitude) of change.

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. Consider the time series $\mathbf{X}_8 = (0, 0, 0, 0, 0, 1, 1, 1)$. The transformation starts by computing the pairwise averages and differences of data (also multiplying by a normalization factor at each level) 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 repeats by applying the same computation on the vector of summary coefficients. The last summary coefficient followed by all $n-1$ detail coefficients form the transformed data, i.e., $(1.06, -1.06, 0, -0.5, 0, 0, -0.7, 0)$.

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. Suppose $|X|$ denotes the length of a vector X, the wavelet transformation[1] of $\mathbf{X}_n$ is $\hat{\mathbf{X}}_n = (\hat{x}_1, ..., \hat{x}_n)$ where $\hat{x}_i = \langle \mathbf{X}_n, \psi_i \rangle . \frac{1}{|\psi_i|}$ and the term $\frac{1}{|\psi_i|}$ is a

[1] Throughout 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.

normalization factor (notice that Haar wavelet basis is orthogonal, and hence self-dual). Moreover, time series $\mathbf{X}_n$ can be represented as $\mathbf{X}_n = \sum_{i=1}^{n} \hat{x}_i \psi_i$. Figure 1 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 $1 \le i \le n$. The first vector ψ_1 consists of n 1's. The remainder $n-1$ vectors corresponding to the detail coefficients are defined as follows:

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

where $0 \le j \le \log n$, $k = 0, ..., 2^j - 1$, and $1 \le l \le n$. We now define the term *support interval* that we will use throughout the paper.

Definition 5. *Let $\hat{\mathbf{X}}_n = (\hat{x}_1, ..., \hat{x}_n)$ 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.*

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 $x_j \in Sup(\hat{x}_i)$.

$$\begin{matrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{matrix}$$

Fig. 1. Haar wavelet basis of size 8

4 Footprints

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 discontinuities in the data. To illustrate the problem, consider the example time series $\mathbf{X}_8 = (0, 0, 0, 0, 0, 1, 1, 1)$. Although there exists only one discontinuity point at fifth position of $\mathbf{X}_8$, 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 at 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. We now explain the idea behind the footprints, assuming for simplicity piecewise constant data only. Consider $\mathbf{X}_n$ with only one discontinuity at position θ:

$$\mathbf{X}_n(i) = \begin{cases} a & 1 \le i \le \theta \\ b & \theta < i \le n\,. \end{cases} \tag{5}$$

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:

$$\mathbf{X}_n = \hat{x_1}\psi_1 + \sum_{i=2}^{n} \hat{x_i}\psi_i\,. \tag{6}$$

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

$$\mathbf{X}_n = \hat{x_1}\psi_1 + \sum_{i \in Sup^{-1}(\theta)}^{i \neq 1} \hat{x_i}\psi_i\,. \tag{7}$$

Now, if we define a new vector f_θ as the linear combination of the multiplication of nonzero coefficients to their corresponding wavelet basis vectors (i.e., the second term in Equation 7), we obtain a representation of $\mathbf{X}_n$ as follows:

$$\mathbf{X}_n = \hat{x_1}\psi_1 + \alpha_\theta f_\theta\,. \tag{8}$$

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 $\mathbf{X}_8$. 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.

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

Theorem 1. *Let $\hat{X}$ denote wavelet transformation of vector $\mathbf{X}$ using orthogonal wavelets (e.g., Haar). If $\mathbf{X}_n = \mathbf{X_1}_n + ... + \mathbf{X_m}_n$, we have $\hat{\mathbf{X}}_n = \hat{\mathbf{X}}_{\mathbf{1}n} + ... + \hat{\mathbf{X}}_{\mathbf{m}n}$.*

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 $\mathbf{X_i}_n$. 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. This idea can easily be generalized to generate all vectors f_i of degree $d = 0, ..., D$ each with only one discontinuity point i ($i = 1, ..., n$) of degree d. Using these vectors as a basis enables us to capture polynomial changes up to degree D in time series data. Interested readers can find the details in [23]. 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., $f_i^{(0)}$,...,$f_i^{(D)}$).

4.1 Properties

In this section, we enumerate different properties of footprints.

- The set of footprints together with the *summary vector* constitutes a basis.
- 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 $f_i^{(D)}$ is $n \times (D+1)$, and hence the resulting basis is overcomplete.
- The footprints efficiently model discontinuities in time series data; a piecewise polynomial time series with K discontinuities, can be represented with only $K \times (D + 1)$ footprints together with the summary coefficients. Each footprint coefficient also contains
  1. The amplitude of the discontinuity that it represents.
  2. The characteristics of the two polynomials right before and after the discontinuity point.

5 Change Detection with Footprints

We showed that nonzero coefficients in footprint transformed time series data are representatives of 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. 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:

- $\mathbf{Q_d}$: Return change points of all degrees.
- $\mathbf{Q_{da}}$: 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. We use footprint basis F_D as our wavelet basis. Therefore, our approach answers change detection queries by returning the nonzero coefficients stored in its database. Figure 2 illustrates the process flow of our approach. We describe each part in details.

5.1 Insert/Update

Upon receiving the new data (i.e., time series), we transform it using F_D and then store it in the database. The transformation will further be explained in

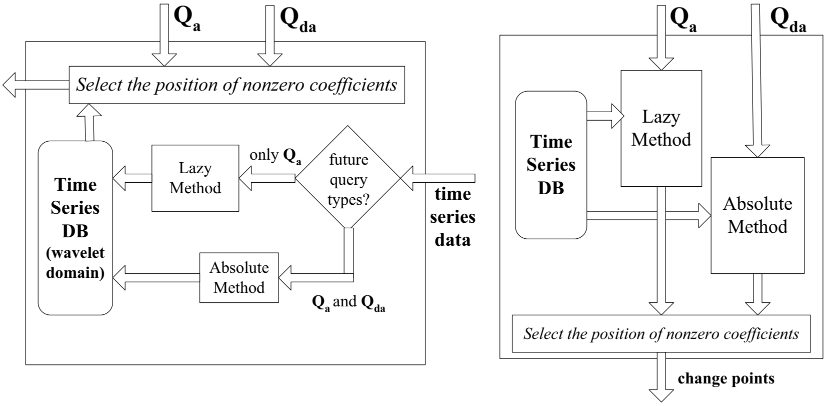


Fig. 2. Query processing in wavelet domain *(on the left)* and Ad hoc query processing *(on the right)*

Section 5.3. 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.

5.2 Query Processing

On receiving a change detection query on a portion of the data, we retrieve the nonzero coefficients corresponding to that portion of the data from the database. For each nonzero coefficient $f_i^{(d)}$, we return a change of degree d at point i. If the user is interested in changes greater than a given threshold, we return only those coefficients that are greater than the threshold. The time complexity of this approach is $O(n)$, since we only need a single scan over the data.[2]

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 the algorithms are highly dependent on either threshold value or degree.

5.3 Footprint Transformation

The challenge here is to transform the data using the footprint basis. We propose two different methods for footprint transformation. In both these methods, we assume that F_D and $\tilde{F}_D$ are pre-computed. Note that the computation of these vectors is completely data-independent.

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.

[2] This can be improved further (perhaps to O(logn)) by using an index-structure on the coefficients.

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 positions as change points. Because of possible noise in the data, both methods may employ thresholds to select nonzero coefficients.

The Lazy Footprint Transformation. 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 5.2). The universal threshold $u = \sigma\sqrt{2 \ln N}$ suggested in [4] is an appropriate candidate for the threshold value.

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. However, in Section 7 we show that the lazy transformation performs very effectively for detecting the change points. For each time series of size n, it is easy to see that the time complexity of the lazy transformation is $O(n^2)$.[3]

The Absolute Footprint Transformation. As mentioned before, 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. Due to lack of space, we omit the details of the technique and corresponding algorithm. Interested readers can find the details in [23].

The set of coefficients returned using the matching pursuit technique are the change points 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 $\mathbf{X_n}$. Since each iteration of matching pursuit algorithm takes $O(n^2)$, the overall time complexity of the absolute transformation becomes $O(\lceil \frac{K}{2} \rceil .n^2)$. [4]

5.4 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.

[3] This can be improved further by utilizing the fact that the $f_i^{(d)}$ matrices are very sparse.

[4] The same as what we had for the Lazy transformation the time complexity can be improved.

6 Customizing Footprints for Wavelet-Based Applications

In this section, we show that our approach can be incorporated within systems where the time series data is maintained in the wavelet domain (e.g., ProDA [10]). 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 basis.

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:

$$\langle \mathbf{X}_n, Q_n \rangle = \langle \hat{\mathbf{X}}_n, \hat{Q}_n \rangle \ . \tag{9}$$

Therefore, ProPolyne evaluates $\langle \hat{\mathbf{X}}_n, \hat{Q}_n \rangle$ as the answer to the query Q_n.

However, it is easy to see that Equation 9 does not hold for wavelet footprints since the footprint basis is not an orthogonal basis. Here, we extend Equation 9 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 9 it is easy to see that

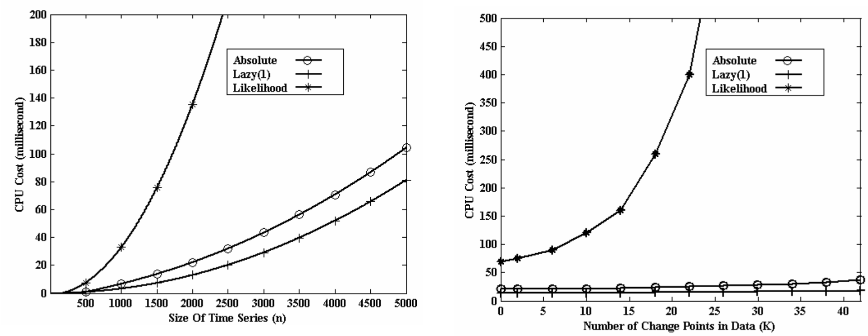
$$\langle \mathbf{X}_n, Q_n \rangle = \langle \hat{\mathbf{X}}_n, \tilde{Q}_n \rangle \tag{10}$$

$$\langle \mathbf{X}_n, Q_n \rangle = \langle \tilde{\mathbf{X}}_n, \hat{Q}_n \rangle \ . \tag{11}$$

In practice, we use Equation 10 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. Hence, we are still 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.

7 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 5.4 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.



a. Query cost vs. size of data b. Query cost vs. number of change points K

Fig. 3. Performance Comparison

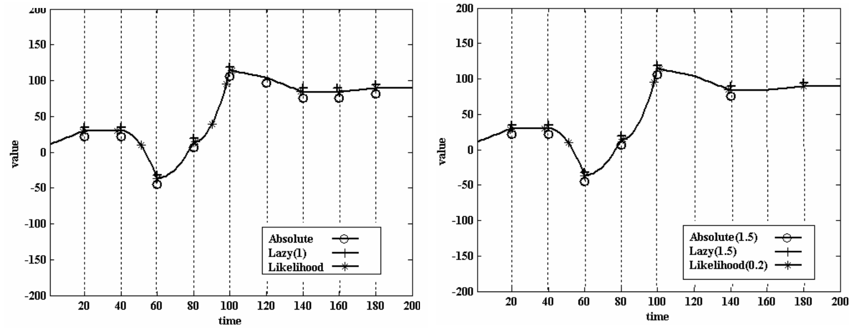
We also evaluated our lazy and absolute methods by investigating the effect of the following parameters on their accuracy: 1) the minimum distance between two consecutive change points (MinDist), 2) the maximum degree of change points in the data (MaxDeg), 3) the maximum degree D of footprint basis (MaxDegF), and 4) the amount of noise in the data (Noise). We represent the accuracy of each method in terms of the average number of false negatives (AFN) and the average number of detected 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. Our real-world dataset include oil and gas time series for different oil wells in California.

Notice that for the absolute method, we used a modified version of matching pursuit introduced in [3] which terminates after $\lceil \frac{K}{2} \rceil$ iterations. For the likelihood method, we use the threshold value with which the method computes the most accurate result. Sections 7.1 and 7.2 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 7.3 discusses our experiments with the real-world data.

7.1 Performance

In our first set of experiments, we compared the performance of our ad hoc change detection query processing. As the Likelihood method uses the original time series data as input, it is only fair to compare it with our ad hoc approach. That is, the CPU time reported for the lazy and absolute methods include both the time for the footprint transformation of time series 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. That is, MaxDeg = MaxDegF. Also, we used polynomials of up to degree 3 in the Likelihood method.



a. lower threshold values

b. higher threshold values

Fig. 4. Detected change points by the absolute, lazy and Likelihood methods with the actual change points in the data *(the vertical lines)*

We varied the size of time series data from 100 to 5,000 and measured the CPU cost of each method. In Figure 3a, 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. We also compared the performance of all the mentioned methods by running the algorithms on overlapping chunks of size 256 for the subset of time series with size larger than 1000. However, since all the methods benefit from the input with smaller sizes the result diagram shows approximately the same trend.

The theoretical time complexities of the absolute and Likelihood methods depend on K. On the other hand, the lazy method is a series of simple projections which are independent of the characteristics of the data. To study the effect of K on the performance of each method, we varied K and measured the CPU cost. Figure 3b 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.

7.2 Accuracy

Our second set of experiments were [2] 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). Figure 4 shows a small time series of size 200 generated with polynomial segments of maximum degree 2 as well as the true change points and those detected by each method. We used footprint basis of up to degree 2 to transform the data and for the Likelihood method we used polynomials of up to degree 2.[5]

[5] Notice that the performance of the likelihood method can be improved further by adapting the technique described in [2]. As part of our future work, we plan to compare the performance of the improved version of the likelihood method with our improved versions of Lazy and Absolute methods.

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

F3 Method	*AFN*	*F2* 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

There are ten actual change points as shown in Figure 4 and the minimum distance between each two change points is 20. In Figure 4a, the likelihood and lazy(1) methods both miss the change point at $t = 120$. Also, Likelihood method detects two false hits at points $t = 51$ and $t = 90$. And 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 points at their exact positions without any false hit. It is interesting to note that the increased threshold values result in ignoring the minor change points at $t = 120$ and $t = 160$ by all the methods as shown in Figure 4b. However the likelihood method still has a false hit at $t = 51$ and for the change that occurs at point $t = 100$, it also detects two change points at $t = 98$ and $t = 101$.

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. Also at point $t = 140$, we have a discontinuity caused by a constant segment following a linear segment.

We repeated the previous experiment on 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 all the methods. Column $F3$ shows the results for the experiment where we used footprints of degrees 0, 1, 2, and 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. The Likelihood method also used polynomials of degrees up to 2 (resp. 3) for case $F2$ (resp. $F3$).

The Effect of MinDist. As Table 1 depicts, our footprint methods always outperform the Likelihood approach in terms of accuracy with *absolute* being the superior method. With small values of MinDist, the accuracies of all methods dramatically downgrade. 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.

The Effect of Noise. In our third set of experiments, 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. Table 2 shows the accuracy results of applying all three methods on both datasets.

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

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

7.3 Experiment with Real-World Datasets

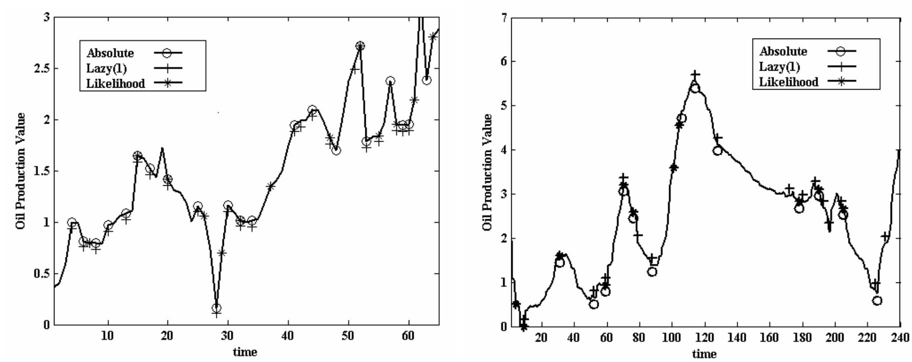
Finally, the last set of experiments focuses on real-world time series data. We tested our methods on different time series generated within the oil industry. Here, we only report the results on three time series OIL1, OIL2, and GAS obtained from Petroleum Technology Transfer Council[6] due to lack of space. These time series are collected from wells in active oil fields in 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 5a, 5b and 6 depicts the change points detected in time series OIL1, OIL2 and GAS, respectively.

Notice that our absolute method does not identify any change at points such as $t = 235$ in Figure 5b. The reason here is that the segment corresponding to the range $[228, 240]$ can be modelled by a polynomial of degree 3. Therefore, $t = 235$ is not a discontinuity of degree 3 in OIL2.

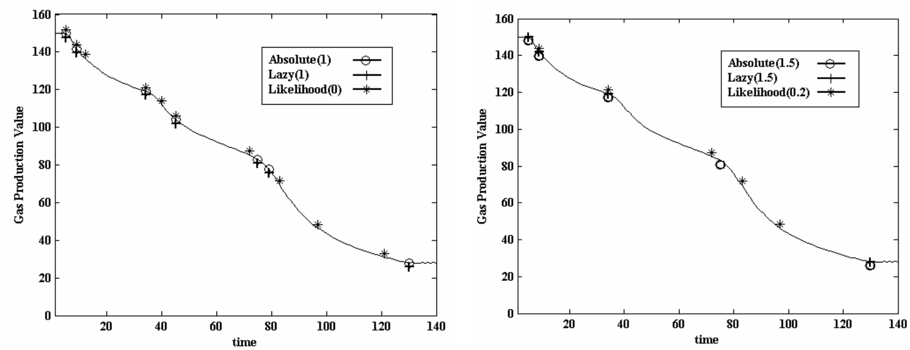
[6] http://www.westcoastpttc.org/.

Figure 6b illustrates the detected change points in GAS data by each of the three methods when they use higher threshold values as compared to those used in Figure 6a. Comparing Figures 6a and 6b 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 a given threshold value, changes above different thresholds can be identified only by performing a simple scan over all the 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 6a and 6b.



a. OIL1 b. OIL2

Fig. 5. Detected change points by the absolute, lazy and Likelihood methods



a. lower threshold values b. higher threshold values

Fig. 6. Detected change points by the absolute, lazy and Likelihood methods

8 Conclusions and Future Work

We studied the problem of detecting changes in time series data. 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 polynomial functions fitting two adjacent segments of data. We then described our novel approach which employs wavelet footprints for defining discontinuities of different degrees. We proposed lazy and absolute methods to transform the data using footprint basis. 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. The results show that our approaches are faster, more accurate and return more information about the changes. In addition their performances are less sensitive to user defined parameters such as threshold values and number of changes.

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.

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 - Chevron initiative.

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. Y. Chen, G. Dong, J. Han, B. W. Wah, and J. Wang. Multi-dimensional regression analysis of time-series data streams. In *VLDB*, pages 323–334, 2002.
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. (demostration).
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. Schafer. *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. M. Sharifzadeh, F. Azmoodeh, and C. Shahabi. Change detection in time series data using wavelet footprints. Technical Report 05-855, Department of Computer Science, University of Southern California, 2005.
24. 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.