EFFICIENT SIMILARITY QUERIES OVER TIME SERIES DATA USING WAVELETS

by

Ivan Popivanov

A thesis submitted in conformity with the requirements for the degree of Master of Science
Graduate Department of Computer Science
University of Toronto

Copyright © 2001 by Ivan Popivanov
The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author’s permission.

L’auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L’auteur conserve la propriété du droit d’auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-62971-6
Abstract

Efficient Similarity Queries over Time Series Data using Wavelets

Ivan Popivanov

Master of Science
Graduate Department of Computer Science
University of Toronto
2001

We consider the use of wavelet transformations as a dimensionality reduction technique to permit efficient similarity search over high dimensional time series data. While numerous transformations have been proposed and studied, the only wavelet that has been shown to be effective for this application is the Haar wavelet.

In this thesis, we prove that a large class of wavelet transformations (not only orthonormal wavelets but also bi-orthonormal wavelets) can be used to support similarity search.

We present a detailed performance study of the effects of using different wavelets on the performance of similarity search for time series data. We include several wavelets that outperform both the Haar wavelet and the best known non-wavelet transformations for this application. We conducted a study including different types of time series data. We define the set of indexable classes of data and we show the performance of our proposed wavelet transformations.
Acknowledgements

There are many people who have helped me with the completion of this thesis. Although I tried to not miss anyone, I would like to apologize for any such omission in advance.

First and foremost, I would like to express my greatest gratitude to my supervisor Renée J. Miller. Her support and inspiring encouragement have been invaluable all the way through my research. Her patience and judgment as a supervisor have made this thesis possible.

I would like to thank Ken Sevcik for reading drafts of this thesis and for providing thoughtful comments that have served to improve the content and the presentation of my thesis.

I like to thank all of my friends for the great time we had together throughout my program at University of Toronto. Thank you Sirish for your friendship and your advice. Thank you Simon for your cheerfulness and for the many laughs we have shared together. Thank you Saman for all the parties we enjoyed together!

Last, but not least, I would like to express my utter gratitude to my parents Valentina and Nedyu Popivanovi for their love and the hard work they did raising me and forming me as a person. Thanks Mom and Dad! I would also like to thank all the other members of my family for their unbounded support through my life. My dearest thanks go to my sister Annie for supporting me, helping me, and being with me through good and bad moments in my life. Thanks Annie! Finally, I would like to thank you Tochka for all your love and encouragement.
## Contents

1 Introduction .................................................. 1  
   1.1 Similarity Search in Time-Series Data ...................... 1  
   1.2 Motivation ................................................... 4  
   1.3 Contributions of the Thesis .................................. 6  
   1.4 Outline of the Thesis ........................................ 6  

2 Query Processing in Time-Series Databases ................. 8  
   2.1 Time-series Databases ........................................ 8  
   2.2 Indexing Time-Series Databases ............................. 9  
   2.3 Contractive Property for Orthonormal Transforms ........... 13  
   2.4 Real Signals .................................................. 14  
   2.5 Data Compression ............................................. 17  
   2.6 Dimensionality Reduction Techniques ....................... 20  
      2.6.1 The Fourier Transform ................................... 20  
      2.6.2 Singular Value Decomposition ............................ 24  
      2.6.3 Piecewise Aggregation Approximation .................... 25  

3 Dimensionality Reduction Using Wavelets ................. 27  
   3.1 Discrete Wavelet Transform ................................ 27  
   3.2 Multiresolution Analysis .................................... 28  
   3.3 Haar Wavelet .................................................. 31
3.4 Benefits of the Discrete Wavelet Transform .................................. 35
3.5 Guarantee of No False Dismissal ...................................................... 36
3.6 Wavelet Comparison ............................................................................. 39

4 Evaluation ................................................................................................. 46
  4.1 Experimental Setup ............................................................................. 46
  4.2 Wavelet Study ..................................................................................... 47
    4.2.1 Dimensionality Study .................................................................... 50
    4.2.2 Filter Length Study ....................................................................... 53
    4.2.3 Data Study .................................................................................... 53

5 Conclusions ................................................................................................ 58

Bibliography .................................................................................................. 60
List of Figures

1.1 Indexing and Querying Strategy for Time Series ...................... 3

2.1 White Noise .................................................. 16
2.2 Brown Noise: Real Stock Data .................................. 17
2.3 Black Noise: Lake Huron Water Level .......................... 18
2.4 Pink Noise: Synthetically Generated Data ....................... 19
2.5 Fourier Representation for the Real Stock Data .................. 20
2.6 Fourier Representation for White Noise .......................... 21
2.7 DFT Reconstruction using 16 out of 1024 Coefficients .......... 22

3.1 The Scaling Function for the Haar Wavelets ....................... 32
3.2 The Mother Wavelet for Haar Wavelets .......................... 33
3.3 Wavelet Decomposition ........................................... 34
3.4 Wavelet Reconstruction .......................................... 34
3.5 Wavelet for the Haar Wavelet Transform ......................... 40
3.6 Ladder-like Approximation for a Smooth Function using the Haar Wavelet and the first 16 Coefficients .................. 41
3.7 Basis Functions for the Daubechies Wavelet of Length 12 ........ 42
3.8 The Same Signal Approximated using Daubechies Wavelet of Length 12 and keeping only Eight Coefficients ..................... 43
3.9 Daubechies Wavelets of Different Length ......................... 44
3.10 Basis functions Daubechies and Symmlet wavelets of length 10 ........ 45

4.1 Precision of wavelets ................................................. 48
4.2 Performance of Wavelets .............................................. 49
4.3 Precision vs. # Dimensions ........................................... 51
4.4 Number of Page Accesses vs. # Dimensions ......................... 52
4.5 Precision vs. Filter Length .......................................... 54
4.6 F-index performance for datasets with different characteristics .... 55
4.7 F-index performance for datasets with different characteristics .... 57
Chapter 1

Introduction

During the past few years, the quantity of data stored in computers has grown rapidly. Much of this data, particularly data collected automatically by sensing or monitoring applications is time series data. A time series (or time sequence) \(^1\) is a real-value sequence, which represents the status of a single variable over time. The monitored activity can be a process defined by some human activity, like the fluctuations in Microsoft stock closing prices, or a purely natural process, like Lake Huron historical water levels. The presence of the time component in data is what unifies such diverse data sets and classifies them as time series.

1.1 Similarity Search in Time-Series Data

During the past two decades, information has been recognized as a vital component of business operations. Huge databases are considered as invaluable sources of information by decision-makers. The accumulation of data has taken place at an explosive rate, hence, information is usually hidden in immense amounts of data and it cannot be extracted efficiently using conventional database management systems. The solution is data mining.

\(^1\)We will use the terms time series and time sequence interchangeably.
Data Mining is a technique to reveal patterns hidden in databases. The use of the hidden patterns as strategic information is application specific. Common examples are:

- Getting better insight into data and analyzing the trends to determine and evaluate the current state of the object of interest.
- Making realistic predictions, thus, advancing the strategic planning process.
- Detecting anomalies.

Whatever the use of time series data, the analysis usually involves queries, which express the notion of similarity as perceived by the user. Therefore, we need a definition of similarity which covers a wide range of applications. One simple, yet universal, approach to determine similarity between two sequences $x$ and $y$ of the same length $N$ is to use the Euclidean distance

$$D(x, y) = \sqrt{\sum_{j=1}^{N} |x_j - y_j|^2}.$$ 

We can define two sequences as similar if their Euclidean distance is less than a user-defined threshold. A query to find all sequences similar to a given sequence is known as an $\epsilon$-query.

Some time series databases are really large. An extreme example is the MACHO project, an astronomical database [24, 22], which contains half a terabyte of data and is updated at the rate of several gigabytes per night. As technology has improved, the time period used for monitoring has been reduced by orders of magnitude. Processes once monitored on a daily basis are measured nowadays every minute or even every second. Hence, the amount of data generated has increased by the same ratio. On the other hand, the importance of the response time to queries has become vital. Therefore, much research has been devoted recently to speeding up data retrieval and, in particular, similarity searches. Indices are often used as important tools in this research.

Typically, interesting queries have reasonable length (for example, tens or hundreds of attribute values). In time series databases, the query length determines the dimension-
The ideal situation, in terms of performance, occurs when the query length matches exactly the dimensionality of the index. Therefore, time series databases have dimensionalities far beyond the practical range of existing multidimensional index structures.

Agrawal, Faloutsos and Sami [1] first proposed to use dimensionality reduction as part of the feature extraction process. Their method appears to be the most promising approach for efficient indexing of high-dimensional, time-series data. The strategy is shown on Figure 1.1. Feature extraction is typically performed by applying a transform to each input sequence and then keeping only a subset of the coefficients. As a result, each sequence of length $N$ is mapped into a point in a $N_f$-dimensional feature space, where $N_f << N$. The original work by Agrawal et al., as well as subsequent research [14, 28], utilizes the Discrete Fourier Transform (DFT) for feature extraction. Later on, the Singular Value Decomposition (SVD) transform has been suggested as a very
accurate, but computationally complex, alternative [35]. Most recently, the Discrete Wavelet Transform (DWT) [8] and other similar techniques [7] have been used to improve various aspects of the query answering process.

1.2 Motivation

Invented in the early 1800s, the Fourier transform has been widely accepted and has become an essential tool in signal analysis. It is based on the simple observation that every signal can be represented by a superposition of sine and cosine waves. Such a superposition, however, works under the assumption that the original signal is periodic in nature. Therefore, it is difficult to model many non-periodic signals with Fourier transforms. Compared to DFT, the Wavelet transform is immature. It was discovered in the early 90s as a reliable tool for signal analysis, yet without the limitations of the Fourier transform. The use of wavelets has led to new developments in the fields of signal analysis, computer graphics, data compression and image processing.

Kanth et al. mentioned DWT as an example transform to speed up similarity search [19]. Chan and Fu, however, were the first to report theoretical, as well as performance evaluation results [8]. For their experimental study they utilize the simplest, yet powerful, wavelet known as the Haar wavelet. However, recent studies using the Haar wavelet (one of many DWTs) for similarity search have not been overly promising [36, 22, 8].

The transforms mentioned so far (DFT, DWT and SVD) share a common property, namely, they are change-of-basis transforms. However, they differ in the procedure used to generate the new basis. DFT utilizes as basis functions the sine and cosine functions. SVD is data dependent, which means it uses the dataset to determine the new basis vectors. The procedure of DWT is to adopt a wavelet prototype function, called a mother wavelet. The basis used in DWT is then generated by applying translation and contraction to the mother wavelet. Therefore, DWT is parameterized by the mother wavelet
function. Different mother wavelets generate different bases, and hence, transforms with different properties. Experience in other areas has shown that DWT can be optimized to a particular application by choosing a wavelet adapted to the data. This feature is based on observation that different wavelets exploit different properties of the dataset. Consider the following examples.

**Example 1:** The JPEG standard for still image compression utilizes the Discrete Cosine Transform to extract features, which can be represented more compactly compared to the original image [26]. The emerging Jpeg 2000 [18] standard replaces the Discrete Cosine Transform used by its predecessor with the Daubechies 9/7 (this is the name of the mother wavelet) wavelet transform [2] and with the symmetric 5/3 wavelet transform [30], for lossy and lossless compression, respectively.

**Example 2:** Recently, the FBI introduced compression support for their fingerprint database. The implemented wavelet compression uses a filter which has been developed by Brislawn [6] and which exploits the properties of fingerprints.

Indeed, the general trend in many areas is to replace DFT with DWT. In this work, we will study the application of DWT as a dimensionality reduction technique for similarity search on time series. We try to answer various questions relating to the framework, performance and configuration. The parameters in our problem include:

- The type of data: The techniques in question are not suitable to any data set.
- The type of the index structure: Many index structure have been developed recently to speed up the query execution in high-dimensional data spaces. Which index configuration is optimal and how does this vary as the index structure changes?
- Wavelet transform parameters: What functions to use?

We try to investigate the relationships among these parameters, and determine the importance of each parameter for efficiency of similarity search over time-series data.
1.3 Contributions of the Thesis

The contributions of this thesis are:

- We prove that the wavelet transform can be used in similarity search based on dimensionality reduction for a large set of wavelets. This set includes, in particular, most of the wavelets used in practice.

- We evaluate several wavelet transforms, comparing them against the other well-known techniques for similarity searches over time series databases.

- We show how versatile the wavelet transform is itself, by conducting a series of experiments using different bases and comparing the performance. Thus, we show that the basis used by the wavelet transform has a substantial influence on the performance. Depending on the data, some wavelets perform better than others. This result is similar to the results in other fields. Notice that the set of wavelets is infinite.

- We identify the classes of data that can be indexed efficiently using wavelet transform.

1.4 Outline of the Thesis

The thesis is organized as follows.

- In Chapter 2, we will give some background and survey the previous work in similarity search for time series data. We will study in detail the process of time series data retrieval, paying particular attention to how the separate parts are combined to work together. We will introduce and discuss all the methods used for dimensionality reduction in similarity search.
In Chapter 3, we briefly discuss the wavelet transform, emphasizing the properties we find vital for its use in the similarity search model. We discuss the intuition of why, by varying the parameters of wavelet transform, one can improve performance significantly. Then we prove that wavelet transforms can be used in the similarity model for a wide class of wavelets. Finally, we discuss the essential properties of wavelets in the context of time series databases.

In Chapter 4, we describe a large set of experiments we have conducted. We empirically compare our method to other techniques and demonstrate its superiority. We discuss the relationships among the parameters in our problem and give practical directions and suggestions on how to find a good configuration for a particular implementation. By studying various wavelets, we determine the best bases for applications which involve similarity search. Finally, we turn our attention to the classification of the signals. We determine the set of real signals suitable for indexing using dimensionality reduction techniques.

In Chapter 5, we give a conclusion and suggest future work in this area.
Chapter 2

Query Processing in Time-Series Databases

2.1 Time-series Databases

Time series reflect the changes of a particular value over a certain time range. A time-series database is a set denoted $DB = X_1, X_2, \ldots, X_M$, where $X_i = [x^{i}_1, x^{i}_2, \ldots, x^{i}_N]$. The dimension of the database is $N$. Each vector $X_i$ contains a sequence of real values or measurements taken at $N$ successive points in time. Alternatively, given a long sequence of values, a time-series database may be derived by using a sliding window of a fixed size over the sequence. So $X_i$ is the $i$th sequence of $N$ values within the full sequence.

Analysis of time-series data is rooted in the ability to find similar series. Similarity is defined in terms of a distance metric, most often Euclidean distance or relatives of the Euclidean distance [1]. Other distance metrics, including the $L_p$ Norms, may also be used [37]. Ignoring the application domain, however, each query belongs to one of the following three classes.

- **Range Query ($\epsilon$-query):** Given a query point $Q$ and a radius $\epsilon$, find all of the points $X \in DB$ such that the Euclidean distance $D(X, Q) \leq \epsilon$. 
• Nearest Neighbor Query: Given a query point \( Q \) find all points \( X \in DB \) such that the Euclidean distance \( D(X, Q) \) is the minimum within \( DB \). A variation, called \( k \)-nearest neighbor query is to find \( k \) closest points in the database to \( Q \).

• All-Pair Query: Given two multidimensional point sets \( s_1, s_2 \subseteq DB \) and a threshold \( \epsilon \), find all pairs of points \( (X, Y) \), where \( X \in s_1 \) and \( Y \in s_2 \), such that \( D(X, Y) \leq \epsilon \).

In this work, we consider only \( \epsilon \)-queries. Solutions to \( \epsilon \)-queries can be adapted to answer nearest neighbor queries [8].

2.2 Indexing Time-Series Databases

Naturally, time series can be considered as points in \( N \)-dimensional space, thus, they could be indexed by a multidimensional index structure. Many of these techniques are efficient for both nearest-neighbor and \( \epsilon \)-queries [3, 5, 7]. However, it is a well-known fact that all techniques for index-based data retrieval fail to scale in high-dimensional spaces. Moreover, various sources report rapid performance deterioration starting as low as 10 dimensions [4, 33]. Most time-series databases typically include a hundred or more dimensions. Thus, their length is far beyond the capabilities of any of the index structures. A common approach to permit the use of multidimensional index structures is to perform dimensionality reduction first. The \textbf{F-index} method[14] can exploit a large class of dimensionality reduction techniques to allow efficient indexing.

To extract object features, the F-index method [14] utilizes a dimensionality reduction technique. The dimensionality is substantially reduced during the process of feature extraction. There are two main issues concerning this process.

• What are the properties that make a dimensionality reduction technique suitable to use in an F-index framework?
The efficiency of the dimensionality reduction technique. That is, what properties are crucial for the performance of a particular technique.

The efficiency is discussed in detail in Section 2.5. Here, we will discuss the F-index in more detail.

Let an F-index be based on a transformation $T$. Then a database $DB$ of dimension $N$ is transformed, by applying $T$ to each item, into a database $DB'$ of dimension $N' \ll N$. We will refer to $DB'$ as the image of $DB$. The benefit is that $N'$ is small enough so that $DB'$ can be indexed efficiently using a multidimensional index structure. To create the index, we usually take the following steps [36].

1. For 1-dimensional time sequence data, the sequence may be chopped into fixed sized subsequence samples using a sliding window.

2. Normalize all the subsequence samples to fit data into a certain range. This process is discussed in detail elsewhere [27].

3. Transform the subsequence samples.

4. Use a few coefficients of the transformed data to represent the original subsequence. These coefficients are used as the key of a multidimensional index that stores all the original data. The leaf level of the index contains the selected coefficients of the transformed data. Each leaf node points to a set of data values in the original data set.

Assuming we have built the index following the above steps, the querying process, for a given query $Q$ and distance $\varepsilon$, proceeds as follows.

1. Normalize and transform $Q$, using the same algorithms as for indexing steps 2 and 3.

2. Create a query $Q'$ with the same dimensionality as the index. Determine the coefficients of $Q'$ extracting them the same way as in step 4 above. Determine $\varepsilon'$, such
that, $D(X', Q') \leq \epsilon'$ implies $D(X, Q) \leq \epsilon$ (as we will see, for many transformations $\epsilon = \epsilon'$).

3. Find all of the points in the index within distance $\epsilon$ of $Q'$. The index is used to locate a (super)set of leaf index pages containing all transformed data points that satisfy $Q'$. This set of pages is brought into memory and pruned to find the set of transformed data points satisfying $Q'$. Note that in this step both index pages and leaf index pages containing transformed data are accessed.

4. For a database $DB$ and a query $Q$, let $Q(DB)$ denote the answer set of $Q$ over $DB$ for a given value of $\epsilon$. Then for each item $q'$, such that $q' \in Q'(DB')$, retrieve the original data pages pointed to by $q'$. This will be the set of data points $\{q : T(q) = q'\}$. This set of pages is brought into memory and pruned to find the set of data points satisfying $Q$. Note that in this step, data pages from the original data set are accessed.

Step 3 uses the index structure to retrieve all the sequences within $\epsilon$ distance to $Q'$. An index structure based on partitioning with spherical or elliptical objects [34] will simplify this step, since only the objects within $\epsilon$-distance will be retrieved at once. However, it is easy to implement this strategy on the top of any structure that uses hyper rectangles for space partitioning. Given the query $Q'$ and the distance threshold $\epsilon$, the query for the index structure is the minimum bounding rectangle for a sphere of radius $\epsilon$ centered at $Q'$. Retrieving this set using the index structure, we can prune it further by discarding all the points that are not within the given distance.

**Definition 1** Given a database $DB$, let $DB'$ denote an image of $DB$, let $Q$ denote a query on $DB$ and $Q'$ denote the query corresponding to $Q$ on $DB'$. A point $X \in DB$ is called a false drop if $X' \in Q'(DB)$ and $X \not\in Q(DB)$.

**Definition 2** Given a database $DB$, let $DB'$ denote an image of $DB$, let $Q$ denote a
query on $DB$ and $Q'$ denote the query corresponding to $Q$ on $DB'$. A point $X \in DB$ is called a false dismissal if $X \in Q(DB)$, but $X' \not\in Q'(DB)$.

Reducing dimensionality of data, inevitably leads to loss of information. Hence, we cannot avoid false drops. We expect the transformation to generate a meaningful low-dimensional representation of the original data, so that after the third step of the algorithm, we will have a set of reasonable size (compared to the size of the real answer set), that is a superset of the answer. During step 4 in the above algorithm, all false drops are pruned, thus the temporary answer set is reduced to the real answer set.

The feature extraction is an essential step in the F-index method. It will not work for any dataset. For example, any transformation-based technique are not helpful when applied to random data. Strong correlations typically found among the coefficients of time series, motivate the application of transforms for this class of data. However, we must ensure that there are no false dismissals. In order to guarantee no false dismissals, the distance measure in the object space ($D_{\text{object}}$) and in the feature space ($D_{\text{feature}}$) must satisfy the following condition [14]:

$$D_{\text{feature}}(T(A), T(B)) \leq D_{\text{object}}(A, B).$$

This property of the transform is often referred to as the contractive property or as the lower bounding lemma [21]. Both DFT and SVD possess the contractive property, when the distance measure is Euclidean distance. A similar property for DWT is discussed in the next chapter.
2.3 Contractive Property for Orthonormal Transforms

Throughout this section, we use matrix notation. For a matrix \( A \), \( A^T \) denotes the transpose of \( A \). Let \( I \) denote the identity matrix

\[
I = \begin{pmatrix}
1 & 0 & \ldots & 0 \\
0 & 1 & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & 1
\end{pmatrix}
\]

then \( A^{-1} \) is the matrix inverse of \( A \), defined by \( AA^{-1} = I \).

All linear transformations can be represented by using a matrix notation: \( X = A^T X \), where \( A \) is \( n \times n \) matrix. The class of orthonormal transformations is defined as follows.

**Definition 3** A transformation is called orthonormal if its transformation matrix \( A \) satisfies \( A^T A = I \). That is, \( A^T = A^{-1} \).

We copy the following Lemma from Fukunaga [17].

**Lemma 1** Orthonormal transformations preserve the Euclidean distance.

**Proof:**

\[
\|X - Y\|^2 = (X - Y)^T (X - Y) = \\
= (A^T (x - y))^T (A^T (x - y)) = \\
= (x - y)^T A A^T (x - y) = \\
= (x - y)^T (x - y) = \\
= \|x - y\|.
\]  

(2.1)

Lemma 1 is significant for dimensionality reduction, since it essentially ensures the existence of a correct query in the feature space. Therefore, the transformations considered in the literature [1, 35, 14, 28, 8, 22] are orthonormal. They preserve Euclidean distance, hence, for \( \epsilon \)-queries, the query transformation algorithm is straightforward.
Observe that for Euclidean distance

\[ D(X, Y) = \left( \sum_{n=1}^{N} |X_n - Y_n|^2 \right)^{\frac{1}{2}} \]

the following holds

\[ \epsilon < D_{N'}(X', Y') \leq D(X', Y') = D(X, Y), \]

where \( D_{N'}(X', Y') \) denotes the distance using only \( N' \) coefficients. The last inequality effectively shows that for the class of orthonormal transformations, there are no false dismissals and hence the existence of a correct transformed query. Note that in this case, the distance threshold for the feature space coincides with the distance threshold of the query.

Both the Fourier transformation matrix and SVD transformation matrix are orthonormal, as are the class of orthonormal wavelets. However, there are wavelets that use a non-orthonormal transformation matrix. This problem will be addressed in more detail in Section 3.5.

### 2.4 Real Signals

Both DFT and DWT transform time series data (or more generally any signal) from the time domain into the frequency domain. The frequency domain representation of data, namely their power spectrum (or energy spectrum) \(^{1}\), is used by Schroeder to classify them into different groups, called noises \([29]\). Functions of the form \( f^{-\beta} \) are adopted to distinguish among noise classes. This classification is summarized in Table 2.1.

White noise has a spectral exponent \( \beta = 0 \). The example shown in Figure 2.1 suggests that the nature of this type of data is completely random. If data is classified as white noise, then its frequency representation is a constant for a finite frequency range. In other words, each \( x[k] \) component of the signal is completely independent of its neighbors.

---

\(^{1}\) We will use the terms power spectrum and energy spectrum interchangeably.
<table>
<thead>
<tr>
<th>$\beta$</th>
<th>Name</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\beta = 0$</td>
<td>White noise</td>
<td>Random data</td>
</tr>
<tr>
<td>$\beta = 1$</td>
<td>Pink noise</td>
<td>Acoustic signals</td>
</tr>
<tr>
<td>$\beta = 2$</td>
<td>Brown noise</td>
<td>Stock prices and exchange rates</td>
</tr>
<tr>
<td>$\beta &gt; 2$</td>
<td>Black noise</td>
<td>Natural phenomenon such as wind speeds and water levels</td>
</tr>
</tbody>
</table>

Table 2.1: Classification of Time-Series Data Characteristics.

$x[k - 1]$ and $x[k + 1]$. Under those circumstances, there are no benefits if we index the frequency domain images of the data. Real signals, and in particular time series data, are not white noise but have skewed energy spectra. Stock prices and exchange rates belong to another group of data, referred to as brown noise (Figure 2.2). It exhibits an energy spectrum of $f^{-\beta}$ with $\beta = 2$. The energy spectrum of the level in water resources (like rivers and lakes) is even more skewed, in fact, it is proportional to $f^{-\beta}$, for $\beta > 2$ and it is called black noise (Figure 2.3). Also, it is worth mentioning, the class called pink noise, which has a power spectrum proportional to $f^{-\beta}$, with $\beta = 1$, and which includes a large number of acoustic signals. Although similar to white noise, pink noise data (Figure 2.4) contains correlations which essentially distinguish it from the completely random signals.

A distinctive feature of all of these signals (pink, brown and black) is that different frequencies, usually the first few, concentrate the energy, therefore, one expects performance benefits from frequency domain indexing. A typical time-series chart of the stock closing prices for a company is shown in Figure 2.2. Such time series are classified as brown noise [29]. Comparing with Figure 2.1 it is apparent that the nature of this signal is not completely random. The signal is further transformed using DFT and the results of the transformed representation are shown on Figure 2.5. Each value along the x-axis gives the magnitude for a particular wave in the sine/cosine superposition for this signal. Hence, the waves close to both ends contribute a lot to the signal, while the waves in the
middle do not. Thus, the first few and the last few frequencies describe the signal almost completely. Keeping them and ignoring the rest, reduces dimensionality enough to allow efficient indexing using a conventional multidimensional index structure. The difference is apparent when we compare to the Fourier representation of the white noise, shown in Figure 2.6. It is similar to the original data, shown in Figure 2.1, which confirms the intuition that random signals are not subject to efficient indexing through dimensionality reduction. The idea of using transformations is to extract meaningful features. Ideally, some of the features are more important than the others (in the case of brown noise, these features are the first few and the last few frequencies), therefore, we can reduce dimensionality substantially. In the case of white noise, however, no subset of the ex-
tracted features is more important than the others, which prevents any efforts to reduce dimensionality further using a transformation to the frequency domain.

2.5 Data Compression

In the context of dimensionality reduction, the most important benefit of using transformations is data compression. Both DFT and DWT, as well as SVD, are change-of-basis transforms and are reversible. The basic idea is that, when applied to non-random signals, the resulting representation is highly skewed, since just a few of the coefficients contribute significantly to the reconstructed signal. Therefore, keeping these coefficients
Figure 2.3: Black Noise: Lake Huron Water Level

and discarding the rest results in data compression without much loss of information. This property is demonstrated using the example from the previous section. The Fourier representation of the Real Stock Data (2.5), strongly suggests that if we keep only the first and the last few coefficients and throw away the rest, then we will have kept most of the information about the data. We took a sequence of length 1024 and we use it as an input for DFT. Then we reconstructed the original signal, but using the first 16 coefficients only. To measure the compression ratio we use the mean relative error.

**Definition 4** Given a signal $x$ of length $N$, let $\tilde{x}$ denote any approximation of $x$. The
mean relative error is defined as the mean of the relative error for each coefficient.

\[
\text{Mean Relative Error} = \frac{1}{N} \sum_{i=1}^{N} \frac{\hat{x}[i] - x[i]}{x[i]}
\]

The original sequence as well as the reconstructed one are shown in Figure 2.7. It is really hard to distinguish which is the original one and which is the reconstructed one. Indeed, the mean relative error for the difference between the original sequence and reconstructed is 0.00077917. Observe that the compression ratio in the case of 16 complex coefficients (which are represented as 32 real numbers) is 32/1024 = 0.0312!
Figure 2.5: Fourier Representation for the Real Stock Data

2.6 Dimensionality Reduction Techniques

2.6.1 The Fourier Transform

The Fourier Transform transforms the time-domain representation of a signal to its frequency domain representation. The original signal is decomposed in terms of the exponential functions $e_N = e^{iN}$, which are orthonormal. The basis functions are sine and cosine waves, hence, every signal is represented by the superposition of finite number of these waves. The Discrete Fourier Transform (DFT) of a finite duration signal $x[k]$ for
$k = 0, 1, \ldots, N - 1$, denoted by $X[k]$, is given by

$$X[k] = \frac{1}{\sqrt{N}} \sum_{l=0}^{N-1} x[l] e^{-\frac{j2\pi kl}{N}} \quad l = 0, 1, \ldots, N - 1$$

where $j = \sqrt{-1}$ is the imaginary unit. The transformation matrix of DFT is orthonormal, therefore, DFT preserves the Euclidean distance. DFT is a complex to complex transformation (even for real signals). Oppenheim [25] proves various unique properties of the DFT when applied to real signals. The following Lemma defines the property exploited by Rafiei and Mendelzon [28] in similarity search.

**Lemma 2** The DFT coefficients of a real-valued sequence of length $N$ satisfy $X_{N+1-i} = X_{i+1}$ for $i = 1, 2, \ldots, N - 1$ where the asterisk denotes complex conjugation.
The square of the distance function is defined as:

\[ D^2(x, q) = D^2(X, Q) = \sum_{i=1}^{N} |X_i - Q_i|^2, \]

and it is the same in the time and in the frequency domain. Using Lemma 2, the square of the distance function becomes [28]:

\[
D^2(X, Q) = |X_1 - Q_1|^2 + \\
\begin{cases}
\sum_{i=2}^{N/2} 2|X_i - Q_i|^2 + |X_{N/2+1} - Q_{N/2+1}|^2 & \text{for even } N \\
\sum_{i=2}^{(N+1)/2} 2|X_i - Q_i|^2 & \text{for odd } N
\end{cases}
\] (2.3)

Equation 2.3 shows that using the first \( k \) coefficients in our index, where \( k \) is smaller than \( n/2 \), we actually have information for \( 2k \) coefficients. That means we have a finer
approximation for the distance. Furthermore, we can use Equation 2.3 to reduce the size of the query rectangle [28]. The conditions

$$2|X_i - Q_i| < \epsilon^2 \quad \text{for } i = 1, 2, \ldots, k$$

(2.4)

are necessary for the left side in 2.3 to be less than $\epsilon^2$. However, they allow us to use a query rectangle of size $2\epsilon$ in one of the dimensions and of size $\sqrt{2}\epsilon$ in all other dimensions. This property of the DFT has been called the symmetric property [21, 36].

For comparison, without using symmetry property, the sum is

$$D^2(X, Q) = \sum_{i=1}^{N} |X_i - Q_i|^2.$$  

(2.5)

and the necessary conditions are

$$|X_i - Q_i|^2 < \epsilon^2 \quad \text{for } i = 1, 2, \ldots, N.$$  

(2.6)

which yields

$$|X_i - Q_i| < \epsilon \quad \text{for } i = 1, 2, \ldots, N.$$  

(2.7)

The last equation lets us use a search rectangle of size $2\epsilon$ for all of dimensions. The results reported by Rafiei and Mendelzon [28], show that using the symmetry property of the DFT, i.e., reducing the search rectangle size in all but the first dimension, improves the performance significantly. According to their analytical results, the symmetry property reduces the search time of the index by 50 to 75 percent. In practice, however, DFT with the symmetry property performs even better, reducing the search time of the index by 73 to 77 percent.

The number of index dimensions is very small compared to the number of dimensions for data. A significant part of the IO is done when accessing the full-dimensional data (in Step 4 of the query answering algorithm), which means that any prior pruning of the result set will improve the number of IOs substantially.

Let $R$ denote the answer set we get from the index for a particular $\epsilon$-query. Assuming that we are using an index structure based on space splitting using bounding rectangles,
we have retrieved points that lie within the query rectangle but out of the sphere defined by the query. We can refine $R$ further using the distance preservation condition. namely

$$D(\tilde{X}_N, \tilde{Q}_N) < \epsilon.$$  

(2.8)

Hence, we decrease the number of page accesses needed to prune the false drops.

### 2.6.2 Singular Value Decomposition

The Singular Value Decomposition (SVD) has been used for indexing images and multimedia objects [35, 19]. For time series indexing, it has been proposed by Chan and Fu [8], however, Keogh et al. [22] were the first to report experimental results.

A unique feature of SVD is that, unlike DWT and DFT, it is a *global* transform. Instead of utilizing particular functions for its basis, it calculates its orthonormal basis from the input data. Examining the whole dataset, it determines a new basis such that the first axis has the maximum possible variance, the second axis has the maximum possible variance orthogonal to the first one, the third axis has the maximum possible variance orthogonal to the first two, etc. This new orthogonal system provides the best least squares fit to the dataset. The transformation is accomplished merely by rotation. therefore, all the lengths are preserved and the contractive property holds.

Mathematically, for a database $DB$ consisting of $M \times N$-dimensional vectors, the Singular Value Decomposition of $DB$ is given by:

$$DB = U \Sigma V^T,$$  

(2.9)

where $U$ is an $M \times N$ matrix and it consists of column-orthonormal row vectors

$$UU^T = I_M \quad U^T U = I_N.$$  

Here $I$ denotes the identity matrix. $V$ is also an orthonormal matrix

$$VV^T = V^TV = I_N.$$
Finally, $\Sigma$ is a diagonal matrix that comprises the non-negative singular values along its diagonal.

$$
\Sigma = \begin{pmatrix}
\sigma_1 & 0 & \ldots & 0 \\
0 & \sigma_2 & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & \sigma_N
\end{pmatrix}
$$

Note that the rotation uses the orthonormal matrix $V$. Therefore, we can multiply both sides in Equation 2.9 by $V$

$$
DBV = U\Sigma
$$

The product $U\Sigma$ contains the rotated set of vectors. Dimensionality is reduced by discarding the least significant singular values in $\Sigma$ and by discarding the corresponding entries in the $U$ and $V$ matrices.

SVD is optimal in several senses and we will expect good performance for indexing. In practice, however, SVD has major drawbacks, which are rooted in its main feature. It requires $O(MN^2)$ to compute, which is quite expensive for large datasets. Moreover, since it is a global transformation, it should be recomputed whenever a database item is updated. Thus, the recomputation time becomes infeasible for practical purposes in the context of a large database.

### 2.6.3 Piecewise Aggregation Approximation

Recently, the Piecewise Aggregation Approximation (PAA) [22] has been proposed as an efficient feature extraction method. In addition to being competitive with the other methods, the original work stresses its ability to support other distance measures including weighted Euclidean queries. Follow-up work by Yi and Faloutsos [37] adopts PAA (also known as segmented means) to support applications that require similarity models based on all $L_p$ norms [37].

The main idea is to divide each sequence of length $N$ into $s$ segments of equal length
n, where s and n satisfy $N = s \times n$. Then, given a sequence $x = [x_1, x_2, \ldots, x_N]$, the corresponding feature sequence comprises mean values of each segment. The new sequence $\bar{x}$ is of length $s$

$$\bar{x} = [\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_s],$$

where

$$\bar{x}_i = \frac{1}{n_i} \sum_{j=n(i-1)+1}^{ni} x_j.$$

For PAA, the dimensionality of the feature space is $s$. The value of $n$ is determined using $s$ together with the length of the input sequences. Zeros are added at the end of each sequence if $N$ is not divisible by $s$. Note that this does not affect query results.

Keogh et al. [22], argued that the feature representation of PAA and of the Haar wavelet transform are similar. However, the segmented means technique gives rough approximation of each sequence, thus, it produces more false drops compared to SVD. DWT and DFT. The IO cost inevitably increases as the number of false drops grows. For short sequences, each false drop increases the number of pages read by one. The ratio is one to one, however, it deteriorates further when the length of the sequences exceeds the page size.
Chapter 3

Dimensionality Reduction Using Wavelets

3.1 Discrete Wavelet Transform

The Fourier transform is based on the simple observation that every signal (a sequence of real values recorded over time) can be represented by a superposition of sine and cosine waves. The Discrete Fourier Transform (DFT) and Discrete Cosine Transform (DCT) are efficient forms of the Fourier transform often used in applications.

Wavelets can be thought of as a generalization of this idea to a much larger family of functions than sine and cosine [12, 31]. Mathematically, a "wavelet" denotes a function $\psi_{j,k}$, defined on the real numbers $\mathbb{R}$, which includes an integer translation by $k$, also called a shift, and a dyadic dilation (a product by the powers of two), often referred to as stretching. The following set of functions, where $j$ and $k$ are integers, form a complete orthonormal system for $L^2(\mathbb{R})$.

$$\psi_{j,k}(t) = 2^{j/2}\psi(2^j t - k) \quad (3.1)$$

Using these functions, we can uniquely represent any signal $f \in L^2(\mathbb{R})$ space by the
following series.

\[ f(t) = \sum_{j,k \in \mathbb{Z}} \langle f, \psi_{j,k}(t) \rangle \psi_{j,k}(t). \]

Here \( \langle f, g \rangle := \int_{\mathbb{R}} f \overline{g} \, dx \) is the usual inner product of two \( L^2(\mathbb{R}) \) functions. The functions \( \psi_{j,k}(t) \) are referred as "basis functions" or as "wavelets". Throughout this text we will use standard notation, with \( \mathbb{Z} \) denoting the set of non-negative numbers, \( \mathbb{R} \) denoting the set of real numbers, \( L^2(\mathbb{R}) \) to be the space of squared integrable real functions, and \( \langle \ldots \rangle \) to be the standard inner product of two real functions of \( L^2(\mathbb{R}) \). The compliment of a set \( V \), will be referred to as \( \overline{V} \), while \( \emptyset \) will denote the empty set. For signals, we will use lower case letters, i.e., \( x \) to denote the time domain representation of a signal, and upper case letters, i.e., \( X \), to denote the corresponding frequency domain representation. For discrete signals \( x \), we will use \( x[t] \) to denote the value of the signal at time \( t \).

### 3.2 Multiresolution Analysis

Manipulating data at various scales has been applied as an efficient approach in many applications. The main idea is to represent a function using a finite number of successive approximations. Then the user is able to work at different levels, each of which corresponds to a particular approximation. Good approximation usually involves more data, thus, such a method allows good control over efficiency. Multiresolution analysis is the general framework for constructing orthonormal bases. Developed by Mallat [23], multiresolution analysis has been widely accepted in wavelet research. By definition, an orthonormal multiresolution analysis of \( L^2 \) is generated by a scaling function \( \phi \) and is a sequence of closed subspaces

\[ \ldots V_{-2} \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \ldots \]

which satisfy:

1. \( V_i \subset L^2(\mathbb{R}), i \in \mathbb{Z} \)
2. $\bigcup_{i \in \mathbb{Z}} V_i = L^2(\mathbb{R})$

3. $\bigcap_{i \in \mathbb{Z}} V_i = \{0\}$

4. $f(\cdot) \in V_0 \iff f(2^i \cdot) \in V_i$, which is the multiresolution condition. In fact, it states that $V_i$ corresponds to a finer resolution when $i$ increases.

5. $f(\cdot) \in V_i \iff f(\cdot - j) \in V_i$. The spaces are shift invariant. In other words, integer translations of any function in the space must still be in the space.

6. The translations $\phi_{ij}$, where

$$\phi_{i,j}(x) = 2^{i/2}\phi(2^i x - j)$$

form an orthonormal basis for $V_i$.

7. For every $j \in \mathbb{Z}$, let $W_j$ denote the orthogonal complement of space $V_j$ in $V_{j+1}$.

$$V_{j+1} = V_j \bigoplus W_j$$

8. For $j \neq j'$, let $W_i \perp W_j$ denote that the spaces $W_i$ and $W_j$ are orthogonal to each other, i.e., each vector in $W_i$ is orthogonal to each vector in $W_j$.

$$W_j \perp W_j'$$

and

$$V_j \perp W_j$$
Multiresolution analysis is a general framework. Wavelets enter the picture, when we extend the above definition by the following property.

9. Each $W_i$ is generated by the translations $\psi_{i,j}$ of the function $\psi$, where

$$\psi_{i,j}(x) = 2^{i/2}\psi(2^i x - j)$$

This last property assures that $W_i$ and the corresponding functions $\psi_{j,k}$, satisfy:

1. $W_i$ form a multiresolution: $f(\cdot) \in W_0 \iff f(2^i \cdot) \in W_i$

2. $\forall i$, $W_i$ is shift invariant: $f(\cdot) \in W_i \iff f(\cdot - j) \in W_i$

3. For $i \neq k$, $W_i$ and $W_k$ are orthonormal: $W_i \perp W_k$

4. All $L^2$ functions are represented as a unique sum:

$$L^2 = \bigoplus W_i.$$  

Therefore, if the extra property is satisfied, it is apparent from the last equality, that an orthonormal multiresolution yields an orthonormal basis. The basis consists of translations and dilations of the wavelet function $\psi$. Wavelet coefficients of function $f$ are defined as the coefficients of $f$ with respect to this basis:

$$w_{i,j}(f) = \langle f, \psi_{j,k} \rangle.$$  

Furthermore, the orthonormality condition can be relaxed. In fact, many wavelets used in practice are not orthonormal. It has been proven that a necessary and sufficient condition for stable reconstruction is that the energy of the wavelet coefficients lies between two positive bounds [9]

$$A \sum_k |X[k]|^2 \leq \|x\|^2 \leq B \sum_k |X[k]|^2,$$  

(3.2)
which will be referred to as *stable reconstruction* condition. This condition guarantees that each function has unique representation, in terms of the wavelet coefficients. Moreover, it guarantees that exact reconstruction is possible. When $A = B$, we have an orthonormal basis. Indeed, the set of wavelet transforms with an orthonormal basis is a proper subset of wavelet transforms that satisfy the stable reconstruction condition. In Section 3.5, we discuss how bi-orthonormal wavelets can be used in an F-index.

How the multiresolution analysis is adopted in practice is the topic of the next section.

### 3.3 Haar Wavelet

The Haar function is the most elementary example of wavelets. Although Haar wavelets have many drawbacks, they still illustrate in very direct way some of the main features of wavelet decomposition. Let’s study them as an example.

The scaling function for the Haar wavelet, i.e., the function whose translations and dyadic dilations generate bases of nested spaces $V_i$ of the multiresolution analysis, is the Box function:

$$
\phi(t) = \begin{cases} 
1, & \text{if } 0 < t < 1 \\
0, & \text{otherwise}
\end{cases}
$$

shown in Figure 3.1. Equation (3.1) implies that the wavelets for a particular basis are built upon a particular function $\psi$, which is often referred to as the *mother wavelet*, and whose translations and dyadic dilations are the bases of the nested spaces $W_i$ of the multiresolution analysis. The mother wavelet for the Haar wavelet is the function

$$
\psi(t) = \begin{cases} 
1, & \text{if } 0 < t < 0.5 \\
-1, & \text{if } 0.5 < t < 1 \\
0, & \text{otherwise}
\end{cases}
$$

It is depicted in Figure 3.2. All the other basis functions, i.e., wavelets, are generated using the mother wavelet and varying $j$ and $k$ in Equation (3.1). It is easy to check that
all the basis functions are orthonormal and satisfy the requirements of multiresolution analysis. Hence, $\psi_{j,k}(t)$ form a complete orthonormal system of $L^2(\mathbb{R})$.

Daubechies [12] discovered that the wavelet transform can be implemented using a pair of Finite Impulse Response (FIR) filters, called a Quadrature Mirror Filter (QMF) pair. These filters are often used in the area of signal processing. Regarding the frequency, the output of a QMF filter pair consists of two separate components: high-pass and low-pass filter output, which correspond to high-frequency and low-frequency output, respectively. The Wavelet transform is considered to be hierarchical since at every single step, the input signal is passed through the QMF filter pair. The output consists of high-pass and low-pass component, each of which is half the length of the input. We
discard the high-pass component, which naturally is associated with details, and we use the low-pass component as further input, thus, reducing the length of the input by a factor of 2. The decomposition process and the corresponding reconstruction process are depicted in Figure 3.3 and in Figure 3.4, respectively. The wavelet coefficients of the input signal consist of all the sequences $d_i$.

The following example demonstrates how the Haar wavelet is used in practice.

**Example** Suppose we have the following one-dimensional signal of length 8:

$$X = [8, 2, 9, 3, 4, 6, -1, 7].$$
Performing the Haar Wavelet transform on it and discarding the details we have:

\[ X = \{5, 6, 5, 3\}. \]

which we use as the input for the next level of the decomposition process. Since the input is 8 it takes two more steps to finish the decomposition process, and we get the following high-pass and low-pass components:

<table>
<thead>
<tr>
<th>Resolution</th>
<th>low-pass</th>
<th>high-pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>[ 8, 2, 9, 3, 4, 6, -1, 7 ]</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>[ 5, 6, 5, 3 ]</td>
<td>[ -3, -3, 1, 4 ]</td>
</tr>
<tr>
<td>2</td>
<td>[ 5.5, 4 ]</td>
<td>[ 0.5, -1 ]</td>
</tr>
<tr>
<td>1</td>
<td>[ 4.75 ]</td>
<td>[ -0.75 ]</td>
</tr>
</tbody>
</table>

The original signal is presented by the last low-pass coefficient and all the details coefficients:
Notice that we can reconstruct uniquely the original signal by applying the reverse process.

In the case of the Haar wavelet, the coefficients at different resolution levels are meaningful. The low-pass filter output is actually the pairwise average of the input. That is, the first two values of the original signal (8 and 2) average to 5. Similarly, 9 and 3 average to 6. and 4 and 6 average to 5. Notice that having the average it is not possible to reconstruct the original values, since an average value of 5 is the result of both pairs (8, 2) and (9, 3). The high-pass filter stores the information needed for the exact reconstruction of the original signal. It stores the distance between the average and the two values in the original signal. The first value of −3 in the above table identifies that the original pair is (8, 2) = (5 − (−3), 5 + (−3). Notice that the sign controls whether it is (8, 2) or (2, 8) = (5 − 3, 5 + 3).

3.4 Benefits of the Discrete Wavelet Transform

The Fourier transform has been widely used as a reliable tool for signal analysis. Nevertheless, it suffers from certain limitations. Practical experience has been shown that for many applications wavelet transforms are as powerful and versatile as the Fourier transform, yet without some of its limitations. The following properties suggest that wavelet transforms may be useful in a scalable dimensionality reduction technique.

- Wavelet transforms have what is called compact support. That means that the basis functions are non-zero only on a finite interval. For comparison, the sine and cosine waves of the Fourier transform are infinite in extent. As a result, unlike the Fourier transform, which maps signals from the time domain into the frequency domain,
wavelet transforms map signals from time domain into time-frequency domain. The Fourier transform gives the set of frequency components that exist in our signal. Wavelet transforms give a gradually refined representation of the signal of different scales, which correspond to basis functions of different length.

- The efficiency of the wavelet transforms is superior even compared with the Fast Fourier transform. In general, the speed of wavelet transforms is \textit{linear} in the length of the signal. Another performance benefit of using wavelet transforms is that they are real to real transforms, hence, they are much easier to implement. This reduces the pre-processing and post-processing of data.

- Wavelet transforms are also hierarchical transformations, thus, compared to the Fourier transform, or to any existing transform used for frequency domain analysis, they allow much finer tuning for a variety of applications.

- The Fourier transform utilizes the sine and cosine functions only. The wavelet transforms, instead, have an infinite set of possible basis functions. Thus, they provide access to information that can be obscured by other methods.

In the next sections, we will show how these properties can be exploited to improve the existing methods for similarity search.

### 3.5 Guarantee of No False Dismissal

The contractive property for the distance in the object and feature space

\[ D_{\text{feature}}(T(A), T(B)) \leq D_{\text{object}}(A, B). \]

is essential to guarantee that an F-index does not result in any false dismissals. In a previous chapter, we have already shown that it holds for any linear, orthonormal
transform. The Haar wavelet belongs to the class of orthonormal wavelets. In other words, all orthonormal wavelets can be used in an F-index method.

Unlike other transforms applied in practice, the DWT basis functions are not always orthonormal. Indeed, the majority of wavelets used in the area of image compression belong to the class of bi-orthonormal wavelets (defined below). As we will see, the class of orthonormal wavelets (for which the basis is orthonormal), is a subset of the class of bi-orthonormal wavelets. The contractive property has not been shown for the class of bi-orthonormal wavelets.

The wavelets are implemented using a pair of FIR filters. Actually, there are two such pairs of filters: the analysis filter and the synthesis filter, which implement the forward and inverse wavelet transform, respectively. All the wavelets, created following a particular approach, are grouped into a family, and each family is named after its creator. Typically, the family members are distinguished from each other by the length of the analysis and synthesis filters. For example, Coiflet 4 (or Coiflet wavelet of length 4), refers to the wavelet of the Coiflet family, with both analysis and synthesis filters of length 4.

Not every function generates a wavelet basis. There are certain requirements that a function should satisfy to be a candidate wavelet basis. To assure proper reconstruction, the stable reconstruction condition (3.2) must be satisfied:

\[ A \sum_k |X[k]|^2 \leq \|x\|^2 \leq B \sum_k |X[k]|^2. \]

Note that (3.2) turns into an equality

\[ \|x\|^2 = \sum_k |X[k]|^2 \]

for \( A = B \). This equality shows that the energy is preserved, and it holds in the case of orthonormal transforms, hence, the class of bi-orthonormal wavelets is a superset of the class of orthonormal wavelets. Indeed, there are wavelets where \( A \neq B \). In such cases,
the transformation is not orthonormal, hence, it is not clear whether we can use them in an F-index.

Now we will prove that the exact reconstruction property [3.2] is essential to convert an \( \varepsilon \)-query over the database to a corresponding \( \varepsilon \)-query against the feature space with guarantee of no false dismissals. First, notice that for \( \varepsilon \)-queries of radius \( \varepsilon \) we have

\[
A \sum_k |X[k]|^2 \leq \|x\|^2 < \varepsilon^2
\]

(3.3)

The last inequality implies that a necessary condition for the norm to be less than \( \varepsilon^2 \) is that every magnitude on the left be less than \( \varepsilon^2 \). Rewriting it for vector \( z = x - y \) we have

\[
A \sum_k |Z[k]|^2 \leq \|z\|^2
\]

Let \( W_{nn} \) denotes the transformation matrix. Since \( Z[k] \) is the transformation of \( z \) and, since, the transformation is linear we have

\[
\sum_k |Z[k]|^2 = \sum_k \left| \sum_l z[l]W_{kl} \right|^2 =
\]

\[
= \sum_k \left| \sum_l (x[l] - y[l])W_{kl} \right|^2 =
\]

\[
= \sum_k \left| \sum_l x[l]W_{kl} - \sum_l x[l]W_{kl} \right|^2 =
\]

\[
= \sum_k |X[k] - Y[k]|^2.
\]

(3.4)

Finally, we get

\[
A \sum_k |X[k] - Y[k]|^2 \leq \|x - y\|^2.
\]

For the norm to be less than \( \varepsilon^2 \), a necessary condition is that each term on the left be less than \( \varepsilon^2 \). This yields

\[
A |X[k] - Y[k]|^2 \leq \varepsilon^2
\]

or, equivalently

\[
|X[k] - Y[k]| \leq \frac{1}{\sqrt{A}} \varepsilon
\]
The last inequality defines the size of the search rectangle to be $\frac{2}{\sqrt{\lambda}}\epsilon$. For each point $X \in S'$, where $S'$ is the answer set after the first stage of the algorithm, if

$$\epsilon^2 < A \sum_k |X[k] - Q[k]|^2 \leq \|x - q\|^2$$

holds, then $r$ does not belong to the answer set. This proves, that with slight modifications, bi-orthogonal wavelets can be used in similarity search for answering $\epsilon$-queries. However, the query transformation must be modified.

### 3.6 Wavelet Comparison

Chan and Fu [8] were first to propose the wavelet transform as a dimensionality reduction technique. Their work is based on the simple, but powerful Haar wavelet. Although, the Haar wavelet shares most of the properties of bi-orthogonal wavelets, there is a major drawback. The basis functions for Haar wavelet, are not smooth (i.e., they are not continuously differentiable), and they have the shape shown in Figure 3.5. It has been shown [36] that wavelets capture the shape of time series better than the Fourier transform, however, the Haar wavelet approximates any signal by a ladder-like structure. This undesirable effect when approximating close to smooth functions is illustrated in Figure 3.6. Hence, the Haar wavelet is not likely to approximate a smooth function well using few coefficients. The number of coefficients we must add is high (slow convergence).

For comparison, we approximated the same function using the Daubechies wavelet with a filter length of 12. Figure 3.7 depicts the shape of the Daubechies basis functions and Figure 3.8 gives the reconstructed signal. Notice that we used half the coefficients, namely eight, to reconstruct the signal. The results are plotted in Figure 3.8. We observed a mean relative error (see Section 2.5) of 0.3732 for the Haar wavelet approximation. The mean relative error improves up to 0.0111 when we use the Daubechies wavelet to approximate the original signal. The compression ratios are $16/1024 = 0.0156$ and $8/1024 = 0.0078$ for the Haar and for the Daubechies wavelet approximations, respectively.
It is apparent that even though it uses only half of the coefficients used by our Haar wavelet reconstruction (eight for Daubechies versus sixteen for Haar), the Daubechies wavelet approximates the original signal much better. Thus, by changing the wavelet used for the transform, we can achieve substantially better approximation using fewer coefficients.

Development of wavelets with more continuous derivatives has attracted the interest of researchers in many areas and spawned the active area of wavelet theory [11, 13]. The Daubechies wavelet families were among the first wavelets with compact support, and are still the most intensively used in practical applications. An essential property of the Daubechies wavelet families is that they have a high number of continues derivatives [32].
Indeed, the longer the filter, the greater the number of derivatives, which is illustrated in Figure 3.9. At the same time, the compact support property ensures that Daubechies wavelets capture local trends in the signal. This feature is unique for wavelet transforms and it has been the main motivation behind the wide application of wavelets in many areas.

Also designed by Daubechies, the class of Symmlet wavelets, presents a further improvement. Symmetry of wavelets appears to be an essential property for compressibility. At the same time, it has been proven in signal analysis that symmetry is not compatible with the stable reconstruction property. The additional advantage of Symmlet wavelets
is that they are nearly symmetric. For comparison, the shape of Daubechies and Symmlet basis functions is depicted in Figure 3.10. Apparently, the Symmlet wavelet is much more symmetric towards its center. Indeed, we have observed minor improvement when using Symmlet wavelets versus Daubechies wavelets.

In Section 2.4, we considered the characteristics of time series data. We showed in particular, that such data is indeed smooth. By using a wavelet which better captures the smooth characteristics of time series data, the same number of coefficients will contain a much better description of the original data. Unlike the Haar wavelet, most other wavelets are smooth. Throughout our experiments, we observe various wavelets outperforming the Haar wavelet. However, some wavelets with smooth basis functions, show inferior
Figure 3.5: The Same Signal Approximated using Daubechies Wavelet of Length 12 and keeping only Eight Coefficients.

performance compared to the Haar wavelet. Our intuition is that their basis functions are not similar to the shape of the time series.
Figure 3.9: Daubechies Wavelets of Different Length
Figure 3.10: Basis functions Daubechies and Symmlet wavelets of length 10
Chapter 4

Evaluation

4.1 Experimental Setup

All the experiments use the same multidimensional index structure. namely Norbert Beckman's Version 2 implementation of the $R^*$-tree [3]. For the wavelet transformations, we used the "Imager Wavelet Library" which is a research library [15]. This library, while perhaps not containing the fastest implementations of specific wavelets, did permit us to experiment with a large suite of dozens of wavelet functions. For comparison with DFT, we used one of the best known implementations [16]. Because of this choice, comparison of CPU times for the transformations would be very biased and unreliable. While the complexity of DFT is $O(n \log n)$ compared to $O(n)$ for wavelets, we actually observed CPU times that were about 6 times faster for DFT than for the Haar wavelet. We do not believe this is a fundamental constraint of wavelets but merely an artifact of these implementations. Furthermore, with the tremendous success of wavelets in numerous applications, we expect to see optimized libraries for wavelets emerging soon.

Different approaches have been proposed to compare the effectiveness of different transformations. We will use the query precision which we define as follows.

\[
Precision(Q) = \frac{\text{Number of sequences in answer of } Q}{\text{Number of sequences retrieved using the index}} \tag{4.1}
\]
Dimensionality reduction implies information loss. Therefore, the size of the data set retrieved using the index is always greater than or equal to the size of the actual result set and the range for the precision is \([0, 1]\).

The precision, however, only compares the pruning power of a particular technique, and it does not measure overall performance. To ensure our results are not biased by implementation details (such as our choice of transformation libraries), we report the performance in terms of the number of physical page accesses. Both for the index and for the database we use 4,096 bytes as the default page size.

### 4.2 Wavelet Study

Our first experiment is designed to determine whether any of the more robust wavelets can be effective for similarity search. To address this question, we copied an experimental set up from Wu et al. which was designed to compare the use of the DFT (using the symmetry optimization \([28]\)) with the Haar wavelet \([36]\).

The data set consists of 1,647 stocks and their historical quotes in several time frames (daily, weekly, etc.)\(^1\) We selected a set of 100 stocks and for each stock, we extracted the closing prices for the last 360 days, ending at September 29, 2000. Then we used a 128-day sliding window, starting at the beginning and taking a sample at each data point. When the window reached the end of the 360 days sequence, we wrapped the beginning of the 360-day sequence to the end. Since we started with 100 stocks and for each sequence we have 360 subsamples, we ended up with 36000 time-series, each of which is 128 days long. The size of the database is approximately 37Mbs, stored in 9,000 data pages. Notice that this database is created using only a fraction of the original data, which we obtained from Yahoo. We used the remaining data to generate the queries for the experiments. To generate a query, we took a random sequence from the remaining

\(^1\)This data set was obtained from Yahoo ("http://chart.yahoo.com/").
dataset. Queries of particular selectivity were generated by using the dataset and the random query point to determine the radius, i.e. $\epsilon$, of the query. All reported results are averaged over the execution of 100 queries.

Our first aim was to study the pruning power of the competitive techniques. Therefore, we fixed the number of index dimensions to 9, but we varied the query selectivities from 0.02 to 0.1. Figure 4.1 shows the precision over different query selectivities. Note that the results for DFT and Haar are very close as is suggested by Wu et al. [36].\(^2\) However, the precision of the Daubechies wavelet is significantly better. This confirms our

\(^2\)Chan and Fu did find that Haar showed improvement over DFT, but we believe this is because they did not use DFT with Rafiei and Mendelzon’s symmetry optimization [28].
prediction that the precision can be improved using wavelets that model the characteristics of the dataset better. PAA is the Piecewise Aggregation Approximation (also known as segmented means) that has been used in several recent studies [21, 22, 37]. We did not expect PAA to perform well on this pruning-power test. However, we have included it since it has been shown to have good performance in increasing classification accuracy for weighted Euclidean distance [22]. Furthermore, it is the only technique suitable for applications that require similarity models based on all $L_p$ norms [37].

![Figure 4.2: Performance of Wavelets](image)

To understand the impact on performance of this improvement, we study the relationship between precision and IO (expressed as the number of page accesses). The page access performance over the range of selectivities is shown in Figure 4.2. For reference,
the database has 9,000 pages so even at the highest selectivity of 10%, the page IO is only 13% of the total database size. The improvement of Daubechies 12 over Haar is about 15%.

We performed this experiment with a large class of wavelets from our library. For this data set, which is an example of brown noise ($\mathcal{J} = 1.99$), we found the Daubechies wavelet with filter length 12 outperformed all other wavelets in the library. This included the Coiflet family and two families of Bathlet wavelets (around 20 wavelets of different filter lengths) among many others. We expect the reason for this performance has to do with the length of the Daubechies filter. Most of the other wavelets in our library had filter lengths less than 10. Intuitively, the length of the filter reflects the length of the patterns that the wavelet is able to capture and use in compressing the data. Given the success of the wavelets with longer filters (the Daubechies 16 performed comparably to the Daubechies 12), we studied in detail the effects of filter length in a separate experiment (Section 4.2.2).

### 4.2.1 Dimensionality Study

To study the impact of the index structure on the overall performance, we analyzed the number of page accesses over a range of dimensionalities. For this study, we used the same stock data as in the previous study and the best transformation for this type of data, the Daubechies 12. The query selectivity was 0.01. To understand the influence of the number of dimensions used in indexing, we plot both the precision for different dimensions (Figure 4.3) and the number of page accesses (Figure 4.4).

Figure 4.3 confirms our intuition that more index dimensions improve query precision. Increasing the number of dimensions also improves search performance, up to a point. To explain this, first notice that the page accesses can be broken down into two components.

\[
\text{Page accesses} = \text{Index page accesses} + \text{Database page accesses}.
\]
An improvement in precision means that the number of database page accesses in the above formula is decreasing. At the same time, the number of index page accesses is steadily growing as the performance of all multidimensional index structures decreases with added dimensions. Before the minimum (in this experiment, 9 dimensions), the growth in the number of index page accesses is less than the reduction in the number of database page accesses, caused by the improved precision. Hence, the search performance is improved. After this minimum, the situation is reversed. This same trade-off will be present with any index structure although the optimal point may be reached at a different number of dimensions.

For the index structure we chose, we found 9 dimensions to be optimal for most
wavelets, with only a few having an optimum at 8 or 10. We also found the same optimum for other data sets that were Brown or midway between Brown and Pink (for example, the River data set described in Section 4.2.3). Hence, we believe this number is influenced primarily by the index structure. Considering the interaction of the index and the choice of wavelet, the wavelets that we found to perform the best were those that were able to concentrate most of the energy to within 9 features.
4.2.2 Filter Length Study

For this experiment, we studied the effect of the filter length on the precision of the query and the performance of the search. We used the Daubechies family of wavelets for two reasons.

- It includes a larger set of filter lengths than other families. For example, the Coiflet family is represented by filters of three different lengths (2, 4, 6), for the Daubechies family we have filters of eight different lengths (4, 6, 8, 10, 12, 14, 16, 20).

- It performs well for all types of data.

We varied the filter length from 4 to 20 and studied the precision. The results are shown in Figure 4.5.

The precision of the wavelets increases with increased filter length to a point, then begins to decrease. For the Daubechies family, both the filter lengths 12 and 16 performed comparably, with only a slight variation depending on the data class. Furthermore, these were the best of all the filter lengths. Note that the filter of length 20 is worse than length 12. We believe the explanation for this is that time series data exhibits strong patterns of length 12 – 16 so shorter filters are not able to take advantage of these trends in compression. For longer filters, these trends are obscured by additional data.

4.2.3 Data Study

White noise (completely random) data, with a spectral exponent $\beta = 0$ is not a subject for our indexing method. On the other hand, brown noise, with spectral exponent $\beta = 2$, has been shown very suitable for indexing using dimensionality reduction techniques. A natural question is how an F-index performs for signals in the gap between white noise and brown noise, or for signals classified as black noise. This question is essential, since there are many real signals exhibiting a spectral exponent in the range $\beta \in [1 - \epsilon, 1 + \epsilon]$, also known as pink noise. At the same time, many natural phenomenon, for example
the level in water resources (like rivers and lakes) yield time series that are even more skewed. This data has a spectral exponent $\beta > 2$, hence, these signals are classified as black noise.

To explore the application extent of the F-index for pink noise and black noise, we use the following datasets.

- **Lakes** - The historical water levels of the following lakes: Erie, Michigan-Huron, Ontario, Superior and Lake St. Clair. Records span the period 1918 - present. The spectral exponent is $\beta = 2.68$.

- **Stocks** - The stock dataset used in the previous two studies. The energy spectrum
is $\beta = 1.99$.

- **Pink** - Synthetically generated dataset. We observe an average spectrum of $\beta = 1.01$ for the output of our data generator.

The Lakes dataset was the smallest with only 3,936 sequences. Therefore, we adjusted the size of the other two datasets to be the same. Throughout this experiments the query selectivity was fixed at 0.01. Data dimensionality is 128, while we vary the index dimensionality between 16, 32 and 64. All the results are averaged over 100 queries.

![F-index performance for datasets with different characteristics](image)

**Figure 4.6:** F-index performance for datasets with different characteristics

In this study, we tried to understand which classes of data can be indexed using an F-index technique. The results are plotted in Figure 4.6. The vertical axis is the
fraction of the database that is scanned throughout the query answering. Thus, linear scan corresponds to a horizontal line at 1.0. Considering the results, we observe that for pink noise data we need to scan more of the dataset even using 64 dimensions than for brown and black noise data using 16 dimensions. Dimensionality of 64 is far beyond the practical range of almost all multidimensional indexing methods. Furthermore, a common rule of thumb in indexing is that if more than 20% of the data needs to be retrieved using the index, then a linear scan is better [33]. Hence, for pink noise data, an index that performed efficiently for 32 dimensions (at least) would be required. Therefore, we conclude that it is unlikely that "pure" pink noise data has strong enough correlations to enable efficient indexing using dimensionality reduction. So the data suitable for an F-index method must have spectral exponent greater than 1.0.

Our experiment (and conclusion) relied on "pure" synthetically generated pink noise. Hence, we decided to perform an additional experiment to try to determine whether any data less correlated than brown noise could be indexed effectively with this technique. We used the Rivers dataset, which we obtained from the Hydro-Climatic Data Network (HCDN).\textsuperscript{3} It consists of stream flow records for 1,659 sites throughout the United States and its Territories. Records cumulatively span the period 1874 through 1988. For this dataset we observe, a spectral component of $3 = 1.4$. Notice that the Lake dataset is excluded in Figure 4.7, in order to compare datasets of reasonable size. Hence, we are able to use our default settings throughout this experiment (rather than a small sample of the data as was used for Figure 4.6). The results for the River data set is much closer to Brown noise, than it is to our pure Pink data set. These results motivate using an F-Index as long as the spectral characteristic of the data surpasses 1.5. We expect many data sets to fit into this category.

Previous work on similarity search using F-Indices has speculated that standard transformations (like the DFT) concentrate the energy of the data into few enough dimensions

\textsuperscript{3}ftp://ftprvaes.er.usgs.gov/hcdn92/
Figure 4.7: F-index performance for datasets with different characteristics that standard indices (like the R*-tree) can be used [1]. However, these studies focus exclusively on Brown noise data. We believe that our results show that less correlated "tan" data can be indexed effectively particularly if we exploit more sophisticated index structures that perform well at higher dimensions.
Chapter 5

Conclusions

In this work, we studied the use of the wavelet transform for similarity search over time series data. Inspired by the success of wavelets in many areas and, in particular, in data compression, we have proposed using full-featured wavelet transforms as an efficient and flexible dimensionality reduction technique.

The F-index model has been used as a de facto standard approach for similarity search over time series data. There are certain requirements in order to adopt a particular transformation in an F-index framework. Therefore, we have first considered the problem of whether wavelets can be used for this application. We showed that a large class of wavelets, in fact all wavelets with a stable reconstruction (the class of bi-orthonormal wavelets), can be used for similarity search. Next, we presented a study of how different wavelets perform in this application. Finally, we experimentally confirmed our theoretical analysis showing the following:

- Indeed wavelets can outperform all well-known transformations used for similarity search.
- We determined the optimal filter length to use for similarity search in time series. We showed that longer filters yield better performance up to a point. The intuition is that longer filters tend to relate more data points. Time series data is strongly
related but over a short time interval. Therefore, the performance improves when extending the filter length up to the limits of this interval.

- Throughout our experiments, we used various real and synthetic datasets, which we classified according to their power spectrum [29]. We investigated the practical limits of similarity search based on dimensionality reduction. We found that wavelets can be effectively used for data that is significantly less correlated than brown noise. In fact, datasets having a spectral exponent, \( \beta \) of 1.5 or higher can be indexed efficiently using an F-index.

Since an F-index comprises different components, we studied the performance when varying the components as parameters. For instance, the dimensionality of the multidimensional index and the length of the filter used for the wavelet transform. Thus, we generalized our results over various F-index configurations.

There are many feasible ways to extend the work presented in this thesis. Because we limited our study to only the \( R^* \)-tree, it is possible that we were not able to take best advantage of some wavelets with an inherent dimensionality that exceeds the capabilities of this index. We are currently studying the use of wavelets with more sophisticated, higher dimensional index structures including the M-tree [10] and the SR-tree [20]. We plan to expand our experiments to other indexing structures given the availability of robust implementations of them. Intrigued by the success of wavelets with long (12-16) filter lengths, we also plan to expand our wavelet library to include additional families with long filters.
Bibliography


