Mining for similarities in aligned time series using wavelets

Ykä Huhtala[a], Juha Kärkkäinen[a], Hannu Toivonen[a,b]

[a]Department of Computer Science, P.O. Box 26
FIN-00014 University of Helsinki, Finland

[b]Rolf Nevanlinna Institute, P.O. Box 4
FIN-00014 University of Helsinki, Finland

ABSTRACT

Discovery of non-obvious relationships between time series is an important problem in many domains, such as financial, sensory, and scientific data analysis. We consider data mining in aligned time series, which arise, e.g., in numerous online monitoring applications, and we are interested in finding time series that reflect the same external events. The time series can have different vertical positions, scales and overall trends, but still show related features at the same locations. The features can be short term such as small peaks and turns, or long term such as wider mountains and valleys.

We propose using a wavelet transformation of a time series to produce a natural set of features for the sequence. Wavelet transformation yields features that describe properties of the sequence both at various locations and at varying time granularities. In the proposed method, these features are processed so that they are insensitive to changes in the vertical position, scaling, and overall trend of the time series.

We discuss the use of these features in data mining, in tasks such as clustering. We demonstrate how the features allow a flexible analysis of different aspects of the similarity: we show how one can examine how the similarity between time series changes as a function of time or as a function of time granularity considered.

We present experimental results with real financial data sets. Experiments indicate that the proposed method can produce useful results. For instance, important similarities can be found in time series that would be considered unrelated by visual inspection. Experiments with compression give encouraging results for the application of the method in mining massive time series data sets.

Keywords: Time series, wavelets, similarity measure, clustering, financial data

1. INTRODUCTION

Large amounts of time series data is produced routinely in a number of application areas, such as finance, economy, process industry, experimental sciences, and medicine. The analysis of such data is obviously an important tool for people working in these areas. Consequently, data mining in time series data has been recognized as a promising and useful task (for a sample of work in this area, see Refs. 1–5).

We consider data mining in databases of aligned time series, such as currency exchange or stock market data. The major problem we consider is measuring the similarity between time series. Similarity of objects or patterns is a central concept in knowledge discovery: whether mining for regularities or exceptions, a measure of similarity (or dissimilarity) is usually needed. The essential problem here is: when should two time series be considered similar?

We apply a feature-based notion of similarity. We propose a method where one extracts features from time series, and compares the feature vectors for similarity. We show how a wavelet transformation of a time series can be used to produce a natural set of features for the sequence. Unlike most other methods proposed for mining time series data, wavelet transform yields features that describe properties of the sequence both at various locations and at varying time granularities. We propose a method for further processing these features so that they are insensitive to changes in the vertical position, scaling, and overall trend of the time series. Figure 1 illustrates these properties on two currencies. Note, in particular, the effect of trend insensitivity in discovering similarities between time series: with a suitable trend change, Swiss franc is quite similar to South African rand.

Email: {*Yka.Huhtala, Juha.Karkkainen, Hannu.Toivonen*}*@cs.Helsinki.FI*
This research has been supported by the Academy of Finland.

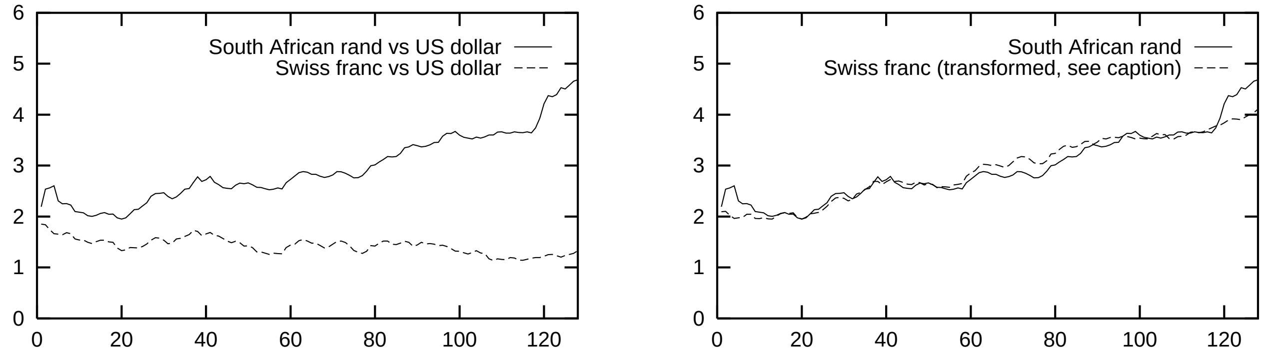


Figure 1. The panel at left shows the exchange rates of two currencies, South African rand and Swiss franc, vs. U.S. dollar during a period of 128 days. The panel at right shows the effect of suitable changes in the vertical position, scaling, and trend of Swiss franc.

Representing time series as feature vectors gives a number of benefits. There are a multitude of methods and tools for processing feature vectors, and we discuss some of these. Dimensionality reduction techniques, such as principal component analysis, are useful for compressing vectors and thus reducing both storage and computational demands. Clustering and related methods, for example the self-organizing maps,[6] can be applied on (potentially compressed) feature vectors. There are also indexing methods for high-dimensional spaces, visualization methods for multi-dimensional data, and so on.

For explorative and flexible data mining it is convenient to use features that have a location and a granularity. It is easy, for instance, to examine the similarity a shorter time interval at a time, by just selecting a subset of the features. By comparing the similarities at different time periods one obtains information about how the similarity changes as a function of time; this could be useful, e.g., for the discovery of deviations.

Related work Much of the effort on time series analysis in data mining has concentrated on finding good alignments of (fragments of) time series.[2–5,7–15] Our problem is different: we are interested in similarities in synchronized time series. The proposed method is less flexible in this sense, but more expressive in terms of invariance with respect to vertical position, scaling, and overall trend. Due to this, the method is able to discover significant similarities in apparently different time series, such as illustrated in Figure 1.

Wavelets have been widely used in statistical time series analysis[16] but, to our knowledge, they have not been used in similarity computations for data mining as described in this paper. Fourier transform, a related technique, has been used in a number of methods but mainly to achieve efficient indexing.[8,17] Fourier transform does not suit well problems where local features are important. The use of wavelet transform instead of Fourier transform was actually suggested in Ref. 17.

In this article we describe the wavelet-based feature extraction method, suggest some data mining applications for the extracted feature vectors, and finally experiment with real financial time series.

2. FEATURE EXTRACTION

Our approach to time series data mining is to first extract a set of features from each time series and then base further analysis on the feature vectors. The feature extraction is based on the discrete wavelet transform. With some postprocessing, the resulting feature vectors have many useful properties. First, the features provide a good coverage of the natural features of the time series both at small scale, such as sharp peaks and turns, and at large scale, such as wider mountains and valleys. Second, the features are insensitive to global changes in the vertical position, scaling, and trend of the time series. Third, the features are nearly independent identically distributed variables, and thus have equal "weights." In this section we describe how the feature vectors are constructed.

Discrete wavelet transform The discrete wavelet transform (DWT) is a linear transformation of a sequence into wavelet coefficients. In this subsection we describe the DWT we are using. For more information on DWT and wavelets, see e.g. Refs. 18,19,16.

The simplest DWT, the Haar transform, for a sequence of length 8, is done by multiplying the sequence vector with the following matrix:

$$H = \begin{bmatrix} 2^{-3/2} & 2^{-3/2} & 2^{-3/2} & 2^{-3/2} & 2^{-3/2} & 2^{-3/2} & 2^{-3/2} & 2^{-3/2} \\ -2^{-3/2} & -2^{-3/2} & -2^{-3/2} & -2^{-3/2} & 2^{-3/2} & 2^{-3/2} & 2^{-3/2} & 2^{-3/2} \\ -2^{-1} & -2^{-1} & 2^{-1} & 2^{-1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -2^{-1} & -2^{-1} & 2^{-1} & 2^{-1} \\ -2^{-1/2} & 2^{-1/2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2^{-1/2} & 2^{-1/2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -2^{-1/2} & 2^{-1/2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2^{-1/2} & 2^{-1/2} \end{bmatrix}$$

The rows of this matrix are the wavelets that extract features from the sequence. The features are also called *wavelet coefficients* since the original sequence is recovered by multiplying the feature vector with $H^T = H^{-1}$. In other words, the DWT gives a decomposition of the sequence as a linear combination of the wavelets.

The matrix H can be decomposed into the following product of three 8×8-matrices.

$$H = \begin{bmatrix} \begin{matrix} 2^{-1/2} & 2^{-1/2} \\ -2^{-1/2} & 2^{-1/2} \end{matrix} & 0 \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} \begin{matrix} 2^{-1/2} & 2^{-1/2} & 0 & 0 \\ 0 & 0 & 2^{-1/2} & 2^{-1/2} \\ -2^{-1/2} & 2^{-1/2} & 0 & 0 \\ 0 & 0 & -2^{-1/2} & 2^{-1/2} \end{matrix} & 0 \\ 0 & 1 \end{bmatrix}$$

$$\times \begin{bmatrix} 2^{-1/2} & 2^{-1/2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2^{-1/2} & 2^{-1/2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2^{-1/2} & 2^{-1/2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2^{-1/2} & 2^{-1/2} \\ -2^{-1/2} & 2^{-1/2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2^{-1/2} & 2^{-1/2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -2^{-1/2} & 2^{-1/2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2^{-1/2} & 2^{-1/2} \end{bmatrix}$$

These matrices consist of the pairs $\begin{bmatrix} -2^{-1/2} & 2^{-1/2} \end{bmatrix}$ and $\begin{bmatrix} 2^{-1/2} & 2^{-1/2} \end{bmatrix}$ called the *mother wavelet* and the *father wavelet*, respectively. It should be easy to see how the construction generalizes for lengths of any power of two. For the length $n = 2^k$, the decomposition consists of k $n \times n$-matrices. The total number of elements other than zeros and diagonal ones is $4n - 4$. Thus, the transformation can be computed in linear time.

The 2^{k-1} wavelets at the bottom of the Haar matrix extract features of very short duration (the fine details) from the sequence. The extracted features are called the *level* $k-1$ wavelet coefficients. The next 2^{k-2} wavelets extract features of twice as long duration, the *level* $k-2$ coefficients, and so on for levels $k-3, \ldots, 0$. The topmost wavelet extracts the vertical position of the sequence; it is called the *transposition coefficient*. All the other coefficients are transposition invariant.

The Haar coefficients are sensitive to the general trend: tilting the sequence affects all coefficients. It would be useful if the fine detail features were not affected by such large scale changes. Wavelets having this property are called second order wavelets. While the Haar coefficients reflect the local trend, the second order wavelet coefficients reflect a deviation from the local trend.

The Daubechies-4 wavelet is a second order wavelet. The transformation matrix is constructed from the mother wavelet $\begin{bmatrix} 0.129 & 0.224 & -0.837 & 0.483 \end{bmatrix}$ and the father wavelet $\begin{bmatrix} 0.483 & 0.837 & 0.224 & -0.129 \end{bmatrix}$ similar to the Haar transform. The transformation matrix for sequences of length 8 is the following.

$$D_4 = \begin{bmatrix} 2^{-3/2} & 2^{-3/2} & 2^{-3/2} & 2^{-3/2} & 2^{-3/2} & 2^{-3/2} & 2^{-3/2} & 2^{-3/2} \\ 0.065 & -0.241 & -0.371 & -0.548 & -0.065 & 0.241 & 0.371 & 0.548 \\ -0.354 & -0.729 & 0.046 & 0.512 & 0.171 & 0.046 & 0.137 & 0.171 \\ 0.171 & 0.046 & 0.137 & 0.171 & -0.354 & -0.729 & 0.046 & 0.512 \\ -0.837 & 0.483 & 0 & 0 & 0 & 0 & 0.129 & 0.224 \\ 0.129 & 0.224 & -0.837 & 0.483 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.129 & 0.224 & -0.837 & 0.483 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.129 & 0.224 & -0.837 & 0.483 \end{bmatrix}$$

Post-processing Next, we describe how to modify the Daubechies-4 wavelet transformation of a time series to obtain a feature vector with the following properties: 1) transposition invariance, 2) trend invariance, 3) scaling invariance, 4) statistical independence of features, 5) similar distribution of the features, and 6) minimal loss of information.

The first two properties are achieved by omitting some of the wavelet coefficients. The coefficients other than the transposition coefficient are transposition invariant and most are also trend invariant. The exception are the wavelets that cross the sequence boundary, for example the fifth row in D_4. The affected coefficients are the first and last coefficient at each level. These as well as the transposition coefficient are omitted from the feature vector. The resulting *cleaned feature vector* has $n - 2k$ coefficients.

The Daubechies-4 (as well as Haar) transform is an orthogonal transformation making the coefficients statistically independent. In typical time series, the coefficients are nearly normally distributed with zero mean. To achieve similar variance, the coefficients may have to be scaled. Suitable scaling for economic time series is to multiply level i coefficients by 2^i, because the amplitude of the deviation represented by a feature is typically proportional to the duration of the deviation. Other kinds of time series may require other kind of scaling. For example, white noise needs no corrective scaling.

As a final step, the feature vector is normalized, i.e., scaled to Euclidean norm 1. This achieves scaling invariance and makes further computations easier.

To achieve the above properties, the wavelet used must be an orthogonal second order wavelet. There are other such wavelets, for example higher order Daubechies wavelets. However, they have wider mother and father wavelets and, consequently, more boundary coefficients to be omitted. Thus, Daubechies-4 provides minimal loss of information among orthogonal second order wavelets.

3. DATA MINING WITH FEATURE VECTORS

In this section, we suggest some applications of the feature vectors in data mining. In the following section, these ideas are then experimentally evaluated on two financial databases.

Similarity measures The basic similarity measure we use for time series is the *cosine similarity* of their feature vectors, i.e, the cosine of the angle between the vectors. For two normalized feature vectors $\mathbf{x}$ and $\mathbf{y}$, the cosine similarity is computed as the absolute value of their inner product, i.e.,

$$sim(\mathbf{x}, \mathbf{y}) = |\mathbf{x} \cdot \mathbf{y}| = |x_1 y_1 + x_2 y_2 + \cdots|.$$

A shortcoming of the cosine similarity is that it is difficult to interpret the significance of a particular value, especially since the significance depends on the number of dimensions of the vectors. For example, in two dimensions, vectors with an angle of 45° are not considered similar, whereas already in 16 dimensions it is exceptional to find two vectors so close to each other. To overcome this problem, we propose a probability-based measure derived from the cosine similarity. The *inverse angle probability* of two vectors $\mathbf{x}$ and $\mathbf{y}$ is defined as

$$iap(\mathbf{x}, \mathbf{y}) = -ln(\Pr\{sim(\mathbf{x}, \hat{\mathbf{x}}) \leq sim(\mathbf{x}, \mathbf{y})\}),$$

Table 1. *Iap* values.

angle	dimension 2	16	128	1024
10°	2.20	27.8	225	1791
45°	0.69	6.50	46.3	358
70°	0.25	1.72	9.53	66.3
80°	0.12	0.68	3.01	17.6
85°	0.057	0.30	1.12	5.25
88°	0.022	0.11	0.36	1.33
89°	0.011	0.054	0.17	0.55

Table 2. Threshold angles for significant similarity.

significance	*iap*	dimension 2	16	128	1024
95%	3.0	4.5°	59.1°	80.0°	86.5°
99%	4.6	0.9°	50.1°	76.8°	85.4°
99.9%	6.9	0.09°	40.5°	73.2°	84.1°
99.99%	9.2	0.009°	33.2°	70.2°	83.0°
99.999%	11.5	0.0009°	27.4°	67.6°	82.1°

where $\hat{\mathbf{x}}$ is a random vector generated by choosing each value independently from the same normal distribution with zero mean. Table 1 illustrates the behavior of *iap* for various dimensions and angles.

Since *iap* is based on probability, it has meaningful interpretations. For example, an *iap* value higher than 3 can be considered to indicate similarity at the 95% significance level ($3 \approx -\ln 0.05$). Table 2 gives the threshold angles for some significance levels and dimensions. This interpretation assumes that the random vector $\hat{\mathbf{x}}$ accurately models the feature vector of a random time series. This is, of course, never precisely true, but with the economic time series that we have experimented with it is remarkably close so. This can also be seen in Figure 2 which includes a time series "Random1" that was generated from a random feature vector by inverse wavelet transform. The random time series looks very much like a real time series.

Clustering There are a large variety of methods for analyzing and visualizing multidimensional data sets. Some of the methods use only a distance or similarity measure on the data points. Examples include many hierarchical clustering methods.[20] Other methods use the whole feature vector, for example nearest centroid methods.[20] The latter category includes also the self-organizing map (SOM)[6] which resembles the nearest centroid clustering methods; the difference is that the SOM clusters have a predefined two-dimensional neighborhood topology.

Explicit selection of features Often there is a need to focus the attention on a particular period of time or a particular level of detail in the time series. Since the features are local in both time and frequency, focusing is possible by simply selecting or applying weights to the relevant features. New feature extraction process is not needed.

This property can be utilized in data mining algorithms on top of the proposed feature representation. Different aspects of similarity can be analyzed by selecting a subset of the features at a time. For instance, by considering the features from one level of the wavelet transformation at a time, one obtains similarities of time series at different time granularities. In a similar way, by selecting features from one time period at a time, one can examine how the similarity changes as a function of time.

Feature compression The time and space complexities of the proposed feature extraction and comparison methods are linear in the total size of the time series. Although this already compares favorably with other time series comparison techniques, it is tempting to improve on this. An efficient and well-justified way to compress feature vectors is the principal component analysis (PCA).[20] PCA applies a linear orthogonal transformation for a set of feature vectors such that the resulting vectors have the largest variance in the first dimension, the second largest variance in the second dimension, and so on. Choosing m first dimensions should produce a more informative set of features than any set of m original features. Other dimensionality reducing methods include Sammon mapping,[21] FastMap[22] and projection pursuit.[23]

4. EXPERIMENTS

We present experimental results with two real-world financial data sets, both publicly available on the Web. Our purpose is to demonstrate the use and behavior of the proposed feature extraction and similarity measurement methods. The routines have been implemented with MATLAB using the WaveLab library of routines for wavelets.[24]

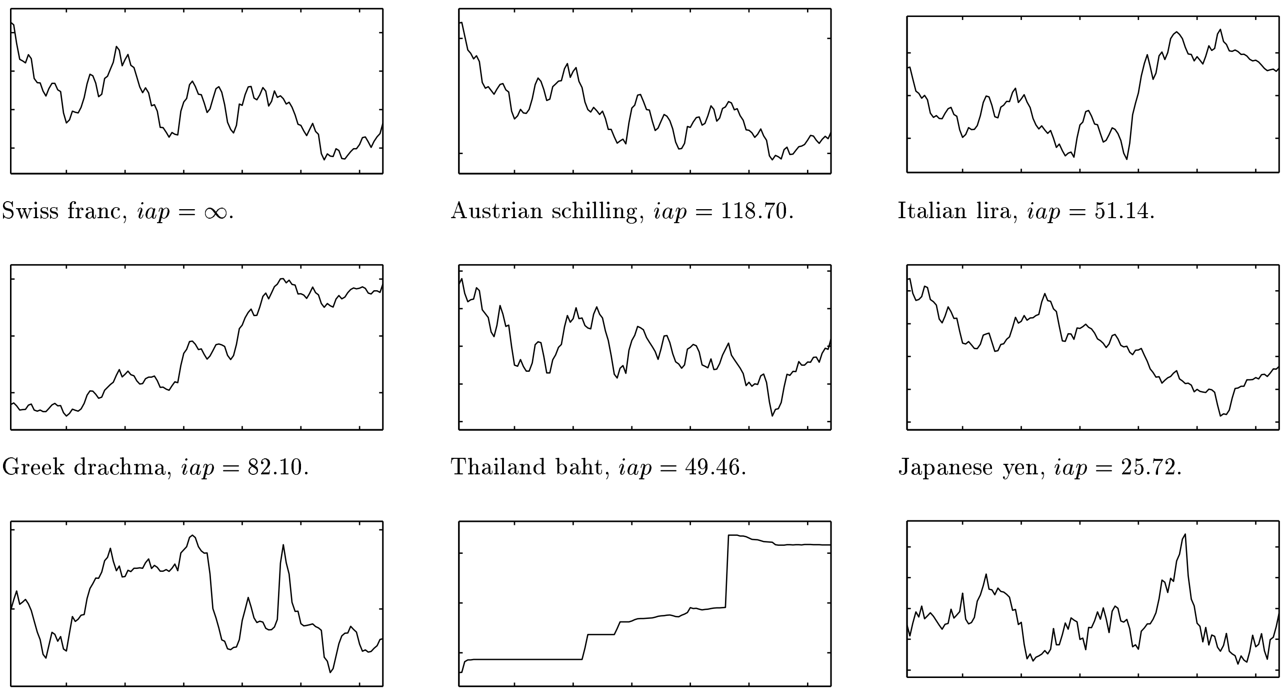
Table 3. The *iap* similarity between Swiss franc and other currencies. The columns show the iap similarity using all 114 features, the 39 features of the period Sep/91 – Apr/94, the 30 level 5 features, 20 PCA-compressed features, 12 PCA-compressed features, and the 126 discrete cosine transform (DCT) features. See text for details.

	Set of features used					
Currency	All	9/91–4/94	Level 5	PCA 20	PCA 12	DCT
Switzerland (Franc)	∞	∞	∞	∞	∞	∞
Austria (Schilling)	118.70	44.51	34.18	23.24	23.80	135.87
Netherlands (Guilder)	118.64	44.82	33.71	23.20	23.69	135.55
Germany (Mark)	117.88	44.01	33.73	23.05	23.78	134.24
Belgium (Franc)	115.11	40.26	31.57	21.41	23.74	131.49
France (Franc)	113.60	40.98	32.29	21.19	23.36	121.73
Denmark (Krone)	99.68	31.42	30.59	18.70	20.77	115.55
Norway (Krone)	86.49	35.47	29.79	16.83	14.63	90.34
Greece (Drachma)	82.10	34.71	31.76	15.31	13.81	89.34
Portugal (Escudo)	75.17	21.45	23.83	13.96	15.54	83.72
Spain (Peseta)	59.59	19.68	23.77	11.58	13.45	64.45
Italy (Lira)	51.14	18.66	14.91	9.71	8.75	45.11
Thailand (Baht)	49.46	24.49	23.18	9.69	7.33	55.39
Finland (Markka)	45.93	11.73	14.76	8.94	7.43	51.96
Sweden (Krona)	45.85	14.67	18.00	8.99	7.52	50.04
Singapore (Dollar)	35.60	16.14	14.62	7.04	8.12	45.01
Japan (Yen)	25.72	9.00	14.23	5.35	3.91	31.66
South Africa (Rand)	21.96	9.38	6.96	4.56	3.44	12.98
Malaysia (Ringgit)	5.55	1.34	3.74	1.42	1.11	7.16
India (Rupee)	3.77	2.00	0.15	1.03	0.82	2.44
Sri Lanka (Rupee)	3.62	0.33	2.78	1.00	0.72	2.51
Taiwan (Dollar)	1.74	0.42	2.26	0.54	0.38	2.67
Random 2	1.60	2.26	0.48	0.51	0.39	0.35
South Korea (Won)	1.05	1.05	2.48	0.36	0.27	1.42
Canada (Dollar)	0.57	0.32	1.42	0.20	0.19	0.12
Random 1	0.48	0.25	0.08	0.18	0.17	1.28
Hong Kong (Dollar)	0.43	0.08	0.13	0.15	0.14	0.71
China (Yuan)	0.35	0.15	0.17	0.13	0.13	0.05

Data sets The first data set consists of the monthly exchange rates of 26 different currencies against the U.S. dollar in the period from May 1986 to December 1996. The data was obtained from the FRED database of the Federal Reserve Bank of St. Louis[25] (*http://www.stls.frb.org/fred/*). For experimental purposes, we added two random time series to this data set (a value for each feature was drawn independently from the standard normal distribution). The currencies are listed in the first column of Table 3 and some of the time series are shown in Figure 2.

The other data set consists of the daily closing values of 164 Canadian stocks and indices on the period from March 31, 1995 to April 12, 1996. The original data (available at *ftp://sunsite.unc.edu/pub/archives/misc.invest/historical-data/stocks/canada/*) covers varying periods and has many missing values. For the experiments we selected the 164 stocks with at most 10 missing values and no consecutive missing values on the selected period. Missing data was filled in by linear interpolation.

Similarity The column "All" of Table 3 shows the overall *iap* similarity of each currency to the Swiss franc. Overall, European currencies are similar to Swiss franc, irrespective of possibly different overall trends. Central European currencies are the most similar ones, South and North European slightly less similar. This result is natural and can be verified by a closer study of the behavior of the European currencies. The time series in Figure 2 show that seemingly different sequences, such as Swiss franc and Greek drachma, can actually be very similar if the major



Swiss franc, $iap = \infty$. Austrian schilling, $iap = 118.70$. Italian lira, $iap = 51.14$.

Greek drachma, $iap = 82.10$. Thailand baht, $iap = 49.46$. Japanese yen, $iap = 25.72$.

Malaysian ringgit, $iap = 5.55$. Chinese yuan, $iap = 0.35$. Random1, $iap = 0.48$.

Figure 2. The exchange rates of some currencies against the U.S. dollar. The similarity of each currency to the Swiss franc is shown below the time series.

trend is ignored. Indeed, a closer look reveals that the two time series share almost all the finer details.

Clustering For a better overview of the similarities between various currencies, we ran clustering experiments using a self-organizing map[6] with the SOM_PAK package.[26] Figure 3 presents the results; the proximity of nodes reflects the similarity of the corresponding clusters. The clustering classifies the currencies quite naturally to European and Asian currencies. The European currencies are separated by large devaluations that occurred slightly after the middle of the observation interval. The European currencies that did not devaluate are in their own cluster at the upper right-hand corner.

For further experiments, we selected from each cluster those two currencies that are closest to the Swiss franc. These currencies will be collectively called by the name of the first country in the cluster.

Comparison based on selected features It is possible to base the comparison of time series on a given time period of the sequences. Figure 4 shows graphically how the similarity of the selected currencies to Swiss franc has changed during the observed period. The changes are not radical, but the period that stands out most is between September 91 and April 94. During this time devaluations of many European currencies took place. The column "9/91–4/94" of Table 3 gives the similarities to the Swiss franc during that period for all currencies.

Another useful possibility is to select only features of some particular granularities for the similarity measurement. Figure 5 shows how the similarity of the selected currencies changes as a function of the level of the features. As a rule, the values of iap decrease due to the decreasing number of features; less features means smaller statistical significance. Many currencies cross between levels 6 and 5, showing how certain currencies are more similar to Swiss franc in "details", whereas others are more similar in a larger time scale. Column "Level 5" of Table 3 lists the similarities in the second finest granularity, the level 5 features.

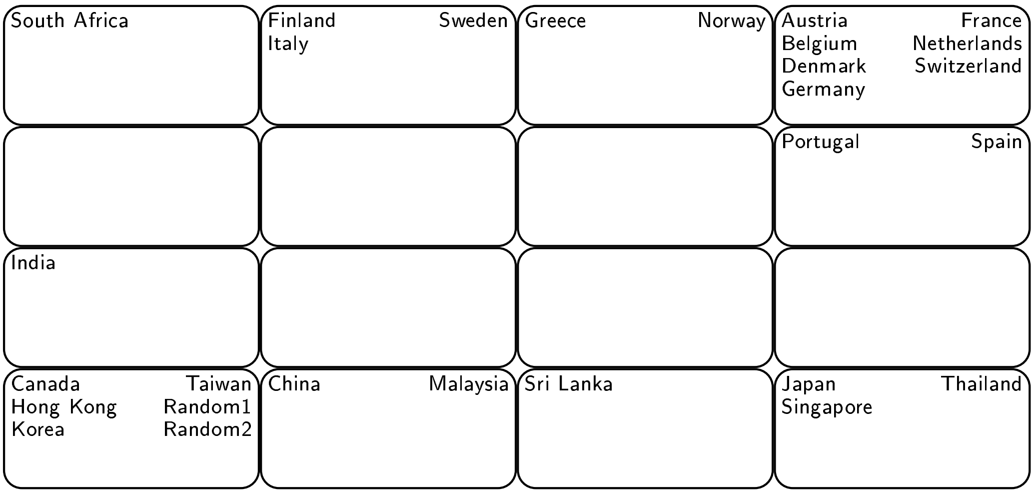


Figure 3. A clustering of the currency database with a self-organizing map.

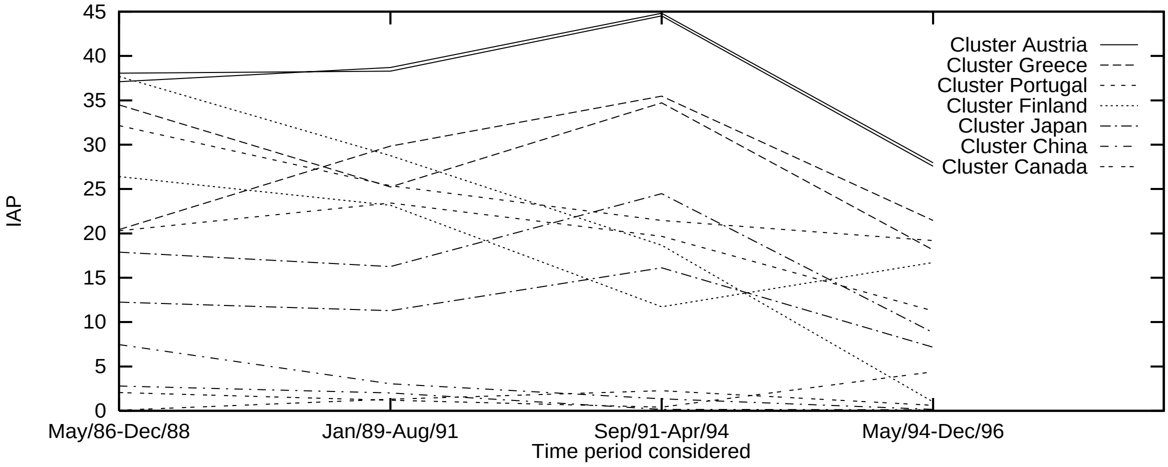


Figure 4. Similarities of currencies to Swiss franc, as a function of time.

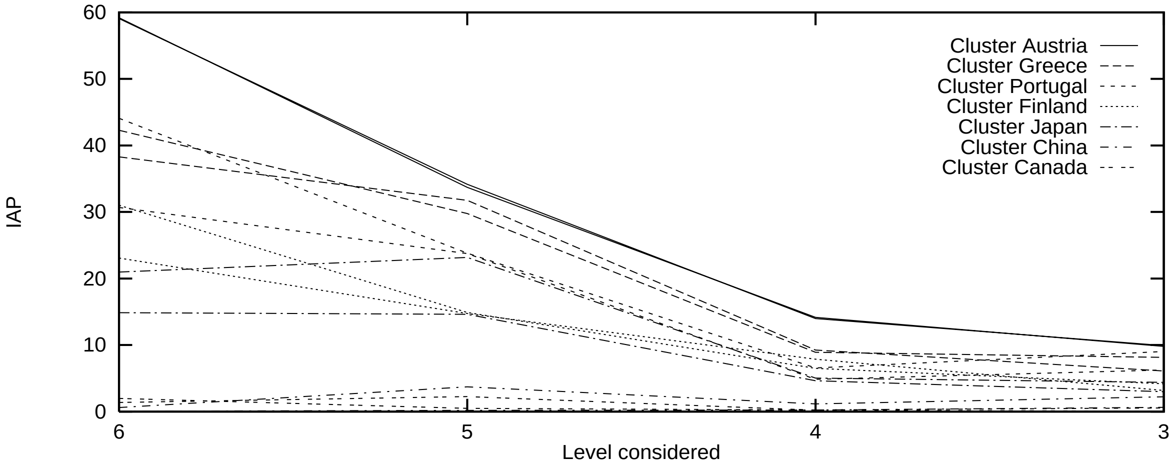


Figure 5. Similarities of currencies to Swiss franc, as a function of the level of features considered. The 62 features on level 6 correspond to periods of 2–4 months; the 6 features on level 3 to 16–32 months.

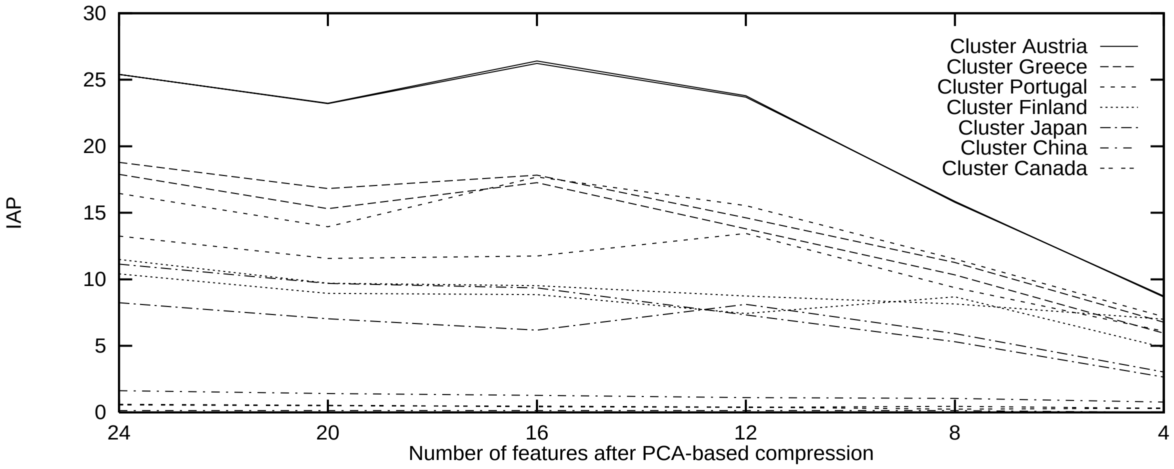


Figure 6. Similarities of currencies to Swiss franc, as a function of number of features used. 24 features correspond to a compression rate of 21 %, 4 features to 3.4 %. The *iap* values at different compression rates are not directly comparable.

Compression We experimentally evaluated the effect of PCA-based compression on the currency data. Columns "PCA 20" and "PCA 12" in Table 3 show the similarities after compressing the original 114 features to 20 or 12 PCA features, respectively. Figure 6 shows how the similarities are affected as a function of the compression. The iap values are lower because PCA selects the dimensions particularly to minimize similarities in the data set. The relative similarities are, however, reasonably well-preserved.

Comparison to DCT We evaluated empirically the effect of using discrete wavelet transform in the feature extraction, in comparison to more traditional feature extraction methods. The discrete cosine transform (DCT) is very similar to the discrete Fourier transform (DFT): both break the sequence into a linear combination of different frequencies. Where DFT uses both sines and cosines, DCT uses only cosines. We computed DCT-based similarities for the currencies the same way we did with DWT. We removed the two lowest frequency coefficients that specify the vertical position and most of the trend. The coefficients were then scaled to have the same variance using the assumption that original variance is proportional to the wave length. Using the resulting feature vectors, we computed the *iap* similarities of all currencies to Swiss franc. The results are given in column "DCT" of Table 3.

As expected, the measures are similar, whether based on DWT or DCT. However, DWT has important advantages over DCT in data mining. First, DWT can be computed in linear time while DCT (as well as DFT) requires $O(n \log n)$ time. Second, with DCT it is not possible to focus on a particular time period without a new transformation. Third, the interpretation of DWT features is easy since they represent natural characteristics such as individual peaks and valleys, rather than different frequencies over the whole time span.

Stock market data For reasons of brevity, we here only mention some of the results of similar experiments on the stock market data. In clustering we would expect to find some information about which companies operate in the same line of business. As it turned out, the group of companies in gold and silver industry formed a strong cluster with one other company — which was into diamonds. Forestry companies also show a distinctive cluster, but other lines of business are not as clearly separated in the clusters.

A query of stocks similar to IBM yielded a curious result. Out of the 5 closest time series to IBM, 4 were indices. This could be interpreted as indicating that either IBM is the most "typical" company in the database or that its weight in the indices is large.

Running time We present performance results only for the larger stock market data set. When storing time series in a database, feature extraction needs to be performed only once as a preprocessing step. In our performance test, the feature vectors were computed for all the 164 time series of length 256 in a total time of 14.4 seconds, i.e., time per sequence was 0.09 seconds.

In a k nearest query test, we compared a query sequence against all the sequences and sorted the result by similarity. This was repeated for each of the 164 sequences as a query sequence. The total time was 29 seconds giving 0.18 seconds for a single k nearest query.

The time complexity of feature extraction is linear in the length of the time series; the complexity of a k nearest neighbor query is linear in the total size of the time series database. These complexities can be reduced significantly by the use of compression. All the routines in the tests were implemented with the MATLAB script language. Implementation with a compiled language should improve the times significantly. The tests were run on a 90 MHz Pentium with Linux operating system.

5. CONCLUDING REMARKS

We have described a method for extracting features from time series using wavelets, and suggested ways of using the features in data mining of time series databases. We also defined the inverse angle probability or *iap* measure of similarity of two time series, based on wavelet-extracted features. Experiments with financial time series produced promising results. The feature-based similarity measure seems to behave sensibly and is able to discover significant similarities in seemingly different time series.

As examples of the utility of the feature extraction method we presented experimental results where the similarities of different currencies to Swiss franc were given as a function of time or frequency. Such information is much more useful in the discovery of interesting relationships or differences between time series than just giving one similarity measure.

Finally, experiments with principal component analysis indicate that compression can reduce the number of features to a fraction of the original without essential qualitative changes in the behavior. This experimental result is encouraging for the application of these methods in very large time series databases.

There are a number of interesting possibilities for the use and processing of the feature vectors. One of the approaches we are planning to try is to replace PCA with the the projection pursuit (PP) methods[23] in compression and dimensionality reduction. A related task is to generalize the *iap* similarity measure for the cases where the features have different variances, for example the PCA and PP feature vectors and feature vectors with user-defined weighting.

Another research direction is to determine what kind of pre- and post-processing methods are useful for different kinds of time series. The feature vectors are insensitive to changes in the base level, scale, and trend of the time series, but can be quite sensitive to some other kinds of phenomena, for example seasonal variation and outliers. There may be need for preprocessing methods to remove such properties before the wavelet transformation. Other kinds of time series may require changes in the post-processing phase, for example different scaling factors for the levels of coefficients. The scaling factors are related to the autocorrelation in time series (see, e.g., Ref. 27).

REFERENCES

1. D. Rafiei and A. Mendelzon, "Similarity-based queries for time series data," in *Proceedings of ACM SIGMOD Conference on Management of Data (SIGMOD'97)*, pp. 13–25, ACM Press, 1997.
2. G. Das, D. Gunopulos, and H. Mannila, "Finding similar time series," in *Proceedings of the 1st European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD'97)*, J. Komorowski and J. Zytkow, eds., vol. 1263 of *LNAI*, pp. 88–100, Springer, 1997.
3. E. Keogh and P. Smyth, "A probabilistic approach to fast pattern matching in time series databases," in *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD'97)*, D. Heckerman, H. Mannila, D. Pregibon, and R. Uthurusamy, eds., pp. 24–30, AAAI Press, 1997.
4. F. Korn, H. V. Jagadish, and C. Faloutsos, "Efficiently supporting ad hoc queries in large datasets of time sequences," in *Proceedings of ACM SIGMOD Conference on Management of Data (SIGMOD'97)*, pp. 289–300, ACM Press, 1997.
5. B.-K. Yi, H. V. Jagadish, and C. Faloutsos, "Efficient retrieval of similar time sequences under time warping," in *Proceedings of the 14th International Conference on Data Engineering (ICDE'98)*, pp. 201–208, IEEE Computer Society Press, Feb. 1998.

6. T. Kohonen, "The self-organizing map," *Proceedings of the IEEE* **78**, pp. 1464–1480, 1990.
7. R. Agrawal, K.-I. Lin, H. S. Sawhney, and K. Shim, "Fast similarity search in the presence of noise, scaling, and translation in time-series databases," in *Proceedings of the 21st International Conference on Very Large Data Bases (VLDB'95)*, U. Dayal, P. M. D. Gray, and S. Nishio, eds., pp. 490–501, Morgan Kaufmann, 1995.
8. C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, "Fast subsequence matching in time-series databases," in *Proceedings of ACM SIGMOD Conference on Management of Data (SIGMOD'94)*, pp. 419–429, ACM Press, 1994.
9. C.-S. Li, P. S. Yu, and V. Castelli, "Hierarchyscan: A hierarchical similarity search algorithm for databases of long sequences," in *Proceedings of the 12th International Conference on Data Engineering (ICDE'96)*, pp. 546–555, IEEE Computer Society Press, 1996.
10. D. Q. Goldin and P. C. Kanellakis, "On similarity queries for time-series data: Constraint specification and implementation," in *Proceedings of the 1st International Conference on Principles and Practice of Constraint Programming (CP'95)*, vol. 976 of *Lecture Notes in Computer Science*, Springer, 1995.
11. H. Shatkay and S. B. Zdonik, "Approximate queries and representations for large data sequences," in *Proceedings of the 12th International Conference on Data Engineering (ICDE'96)*, pp. 536–545, IEEE Computer Society Press, 1996.
12. C. Faloutsos, H. V. Jagadish, A. Mendelzon, and T. Milo, "A signature technique for similarity-based queries," tech. rep., AT&T Bell Laboratories, 1996.
13. H. Shatkay, "Approximate queries and representations for large data sequences," Tech. Rep. CS-95-03, Department of Computer Science, Brown University, Mar. 1995.
14. R. Agrawal, G. Psaila, E. L. Wimmers, and M. Zait, "Querying shapes of histories," in *Proceedings of the 21st International Conference on Very Large Data Bases (VLDB'95)*, U. Dayal, P. M. D. Gray, and S. Nishio, eds., pp. 502–514, Morgan Kaufmann, 1995.
15. D. J. Berndt and J. Clifford, "Finding patterns in time series: A dynamic programming approach," in *Advances in Knowledge Discovery and Data Mining*, U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., pp. 229 – 248, AAAI Press, Menlo Park, CA, 1996.
16. R. T. Ogden, *Essential Wavelets for Statistical Applications and Data Analysis*, Birkhauser, Boston, 1997.
17. R. Agrawal, C. Faloutsos, and A. N. Swami, "Efficient similarity search in sequence databases," in *Proceedings of the 4th International Conference of Foundations of Data Organization and Algorithms (FODO'93)*, D. Lomet, ed., vol. 730 of *Lecture Notes in Computer Science*, pp. 69–84, Springer, 1993.
18. C. K. Chui, *An Introduction to Wavelets*, vol. 1 of *Wavelet Analysis and its Applications*, Academic Press, 1992.
19. A. Graps, "An introduction to wavelets," *IEEE Computational Science and Engineering* **2**(2), pp. 50–61, 1995. Also available in HTML format at http://www.amara.com/IEEEwave/IEEEwavelet.html.
20. M. R. Anderberg, *Cluster Analysis for Applications*, Academic Press, New York, 1973.
21. J. Sammon, "A nonlinear mapping for data stucture analysis," *IEEE Computer* **C-18**, pp. 401–409, 1969.
22. C. Faloutsos and K.-I. Lin, "FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets," in *Proceedings of ACM SIGMOD Conference on Management of Data (SIGMOD'95)*, M. J. Carey and D. A. Schneider, eds., pp. 163–174, ACM Press, 1995.
23. J. H. Friedman, "Exploratory projection pursuit," *Journal of the American Statistical Association* **82**(397), pp. 249–266, 1987.
24. J. Buckheit, S. Chen, D. Donoho, I. Johnstone, and J. Scargle, "WaveLab .701 [http://playfair.stanford.edu/~wavelab/]," Mar. 1996.
25. "FRED database — federal reserve economic data, Federal Reserve Bank of St. Louis, [http://www.stls.frb.org/fred/]," Sept. 1997.
26. T. Kohonen, J. Hynninen, J. Kangas, and J. Laaksonen, "SOM_PAK: The Self-Organizing Map program package," Report A31, Helsinki University of Technology, Laboratory of Computer and Information Science, Jan. 1996.
27. C. Chatfield, *The Analysis of Time Series, An Introduction*, Chapman & Hall, London, UK, 1989.