$$\mathbf{Y} = \mathbf{y}\hat{\mathbf{V}} \tag{2.137}$$

Similarly, the reconstruction of $\tilde{\mathbf{y}}$, the approximation of a time series $\mathbf{y}$, from its SVD coefficients $\mathbf{Y}$ is given as follows.

$$\tilde{\mathbf{y}} = \mathbf{Y}\hat{\mathbf{V}}^T \tag{2.138}$$

The orthonormal basis vectors $\{\mathbf{V}_i\}$ of SVD allow us to minimize the approximation errors for a collection of time series globally. In DFT and DWT, the orthonormal basis vectors are data independent, that is, the basis vectors chosen are not derived from the data. By contrast, SVD finds the optimal basis vectors given the time series.

Here is an experiment on a collection of 300 random walk time series that demonstrates the above properties of SVD. Figure 2.17(a) shows two of the random walk time series and their SVD approximations. Each of the time series has the length $n = 250$. They are approximated by 8 SVD coefficients. We can see that with only 8 SVD coefficients, we capture the raw shape of the time series very well. The basis vectors for the SVD are also shown in fig. 2.17(b). We can see that the basis vectors are similar to the basis vectors of DFT, though they are not exactly trigonometric functions.

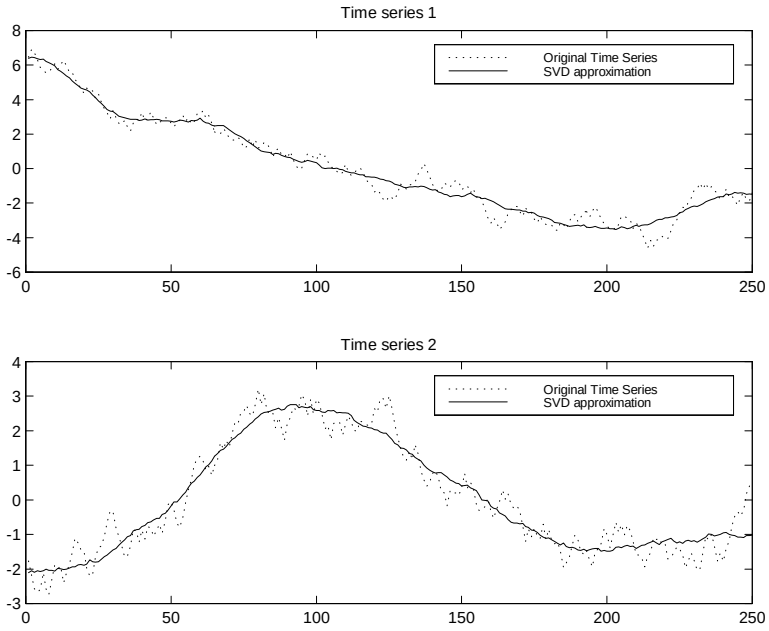
To demonstrate the adaptivity of SVD, we add some components to each of time series in the collection. A short burst is superimposed to each time series either around the time point of 100 or 200. Two of the resulting time series samples are shown in fig. 2.18(a). We compute the SVD of the new collection of time series again and show their SVD approximations. The SVD approximations follow the new burst very closely. The reason is that the basis vectors computed from the new data can incorporate the burst adaptively. In fig. 2.18(b), the basis SVD vectors for the new collection of time series show the new burst components clearly.

For a collection of time series, the SVD approximations are the closest approximations overall in terms of Euclidean distance comparing to any orthogonal-based transform such as DFT and DWT.

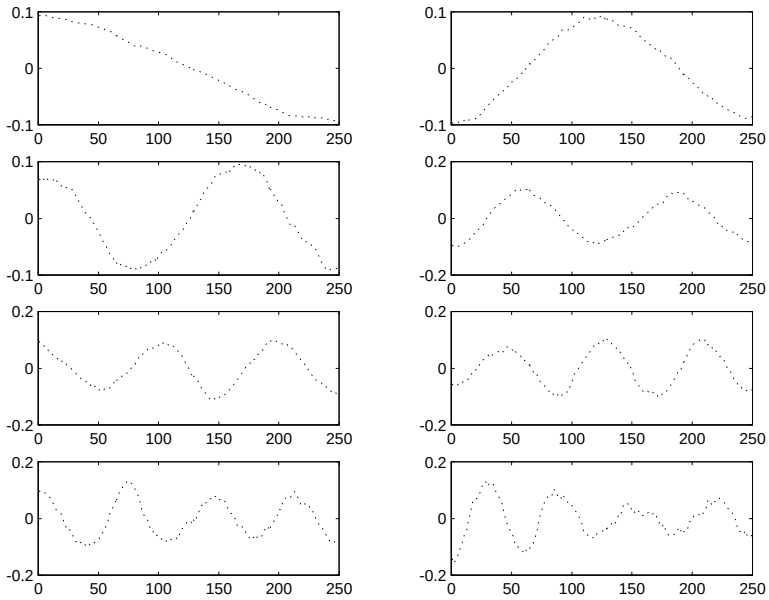
2.4 Sketches

The data reduction techniques we have discussed so far are all based on orthogonal transformations. If we think of a time series as a point in some high dimensional coordinate space, orthogonal transformation is just the rotation of the axes of the coordinate space. If a collection of time series have some principal components and the transformed axes are along these principal components, we can keep only those axes corresponding to the principal components to reduce the dimensionality of the time series data.

But what if the data do not have any clear principal components? Consider a collection of time series of white noise for example. Clearly, we cannot use

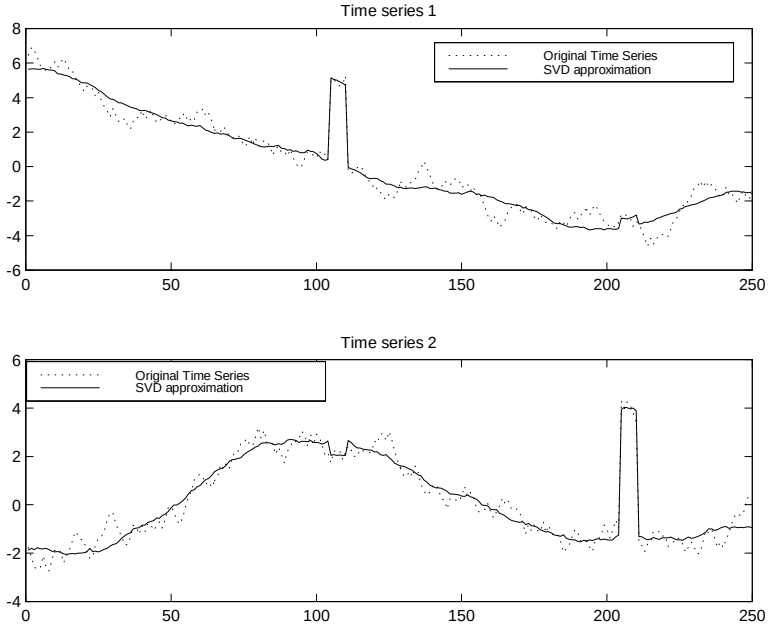


(a) The SVD approximations

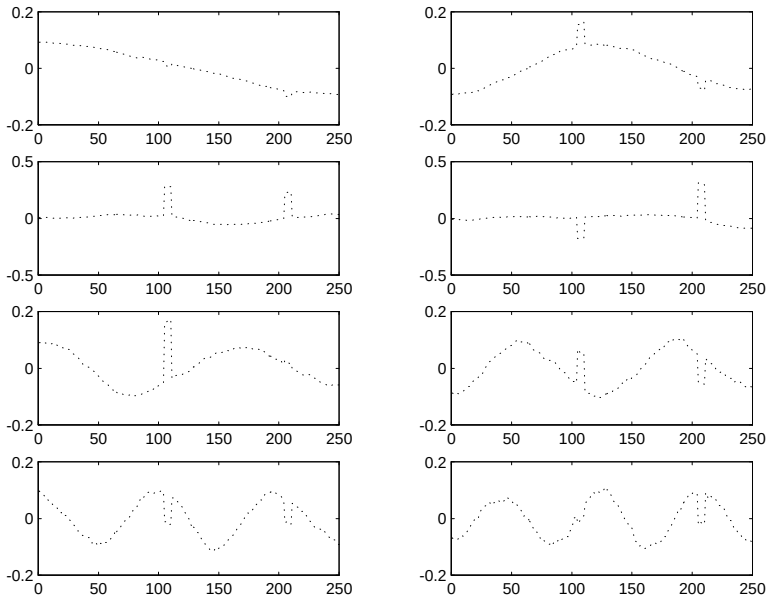


(b) The SVD basis vectors

Fig. 2.17. SVD for a collection of random walk time series



(a) The SVD approximations



(b) The SVD basis vectors

Fig. 2.18. SVD for a collection of random walk time series with bursts

orthogonal transformations such as DFT, DWT or SVD. We need a new kind of Data Reduction technique: *random projection*.

Random projection will project a high dimensional point corresponding to a time series to a lower dimensional space randomly based on some distribution. If we choose the distribution carefully, we can have some probabilistic guarantee on the approximation of the distance between any two higher dimensional points to their corresponding distance in the lower dimensional space.

In random project, we try to approximate the distance between each pair of time series given a collection of time series, instead of getting some approximation of time series as for DFT, DWT and SVD.

Unlike data reduction based on orthogonal transformations, random projection can approximate different types of distances. We will start with the Euclidean Distance.

2.4.1 Euclidean Distance

Random projection is based on the construction of the sketches for a time series.

Definition 2.46 (sketches). *Given a time series* $\mathbf{x}^n = (x(1), x(2), ..., x(n))$ *and a collection of* k *vectors* $\mathbf{r}_i^n = (r_i(1), r_i(2), ..., r_i(n)), i = 1, 2, ..., k$, *where all elements* $r_i(j), i = 1, 2, ..., k, j = 1, 2, ..., n$, *are random variables from a distribution* $\mathcal{D}$, *the* $\mathcal{D}$*-sketches of* $\mathbf{x}$ *are defined as* $\mathbf{s}(\mathbf{x}) = (s(1), s(2), ..., s(k))$, *where*

$$s(i) = \langle \mathbf{x}, \mathbf{r}_i \rangle, \qquad i = 1, 2, ..., k \tag{2.139}$$

i.e., $s(i)$ *is the inner product between* $\mathbf{x}$ *and* $\mathbf{r}_i$.

The sketches can be written as

$$\mathbf{s}(\mathbf{x}) = \mathbf{x}\mathbf{R}, \tag{2.140}$$

where

$$\mathbf{R}_{n\times k} = (\mathbf{r}_1^T, \mathbf{r}_2^T, \ldots, \mathbf{r}_k^T) = \begin{pmatrix} r_1(1) & r_2(1) & \ldots & r_k(1) \\ r_1(2) & r_2(2) & \ldots & r_k(2) \\ \vdots & \vdots & \ddots & \vdots \\ r_1(n) & r_2(n) & \ldots & r_k(n) \end{pmatrix}. \tag{2.141}$$

The collection of random vectors $\mathbf{R}$ is called the *sketch pool*. Obviously, the time complexity to compute the sketch for each time series is $O(nk)$.

If the distribution $\mathcal{D}$ is a Gaussian distribution, we can compute the Gaussian sketches of a time series. The most important property of sketches is stated by the Johnson-Lindenstrauss lemma[52] for Gaussian sketches.

Lemma 2.47. *Given a collection C of m time series with length n, for any two time series $\mathbf{x}, \mathbf{y} \in C$, if $\epsilon < 1/2$ and $k = \frac{9 \log m}{\epsilon^2}$, then*

$$(1 - \epsilon) \leq \frac{||\mathbf{s}(\mathbf{x}) - \mathbf{s}(\mathbf{y})||^2}{||\mathbf{x} - \mathbf{y}||^2} \leq (1 + \epsilon) \tag{2.142}$$

holds with probability $1/2$, where $\mathbf{s}(\mathbf{x})$ is the Gaussian sketches of $\mathbf{x}$.

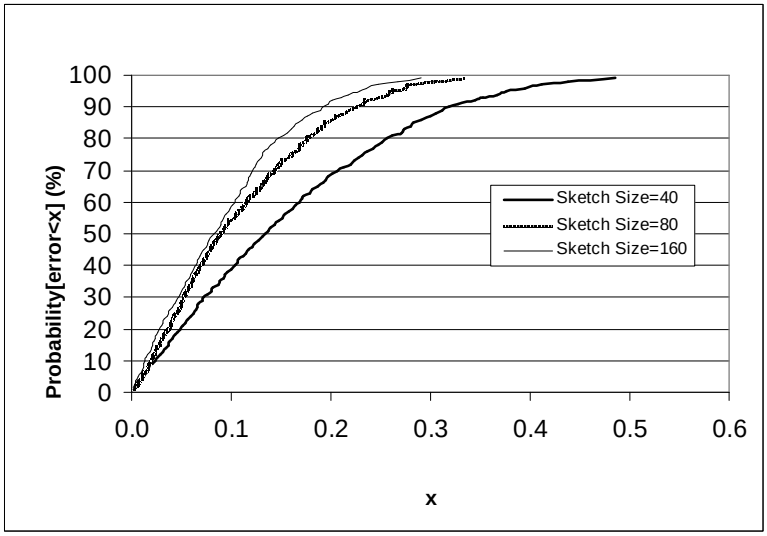
In other words, Johnson-Lindenstrauss lemma says that a collection of m points in a n-dimensional space can be mapped to their k-dimensional sketch space. The Euclidean distances between any pair of points in the transformed sketch space approximate their true distance in the n-dimensional space with probability $1/2$. There are also other flavors of random projection based on Johnson-Lindenstrauss lemma that give similar probabilistic approximation of Euclidean distance, such as [7].

From the lemma, we can see that if we increase the size of the sketches k, the approximation error ϵ will be smaller. Also we can boost the probability of success using standard randomization methods. If a sketch approximation is within ϵ, we call it a success. We keep s different sketches and repeat the approximate distance computation s times, the probability that the median of the approximate distances is within precision ϵ is the same as the probability that the number of success is larger than $s/2$. From the Chernoff bound, if we compute $O(\log 1/\delta)$ repetition of sketches and take the median of the distance between the sketches, the probability of success can be boosted to $1 - \delta$.

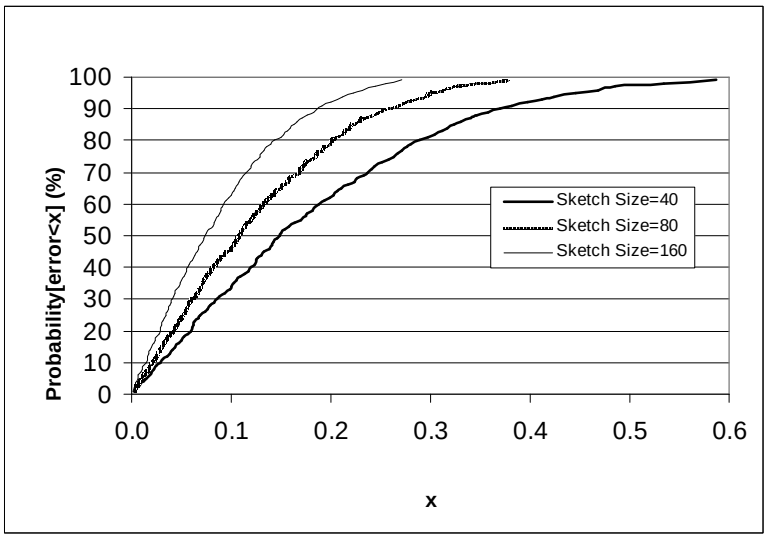
Using sketches to approximate the Euclidean distances between time series, we do not require the time series to have principal components. We perform the follow experiments to verify this.

The time series under consideration is a collection of $10,000$ stock price time series. The time unit of the time series is one second. Each time series in the collection has a size $3,600$, corresponding to one hour's data. We compute the sketches of these time series with sketch size $k = 40$ and therefore reduce the dimensionality of the time series from $3,600$ to 40. We randomly pick $1,000$ pairs of time series in the collection and compute their Euclidean distances. The approximations of the distances using sketches are also computed. Let the approximation error be the ratio of the absolute difference between the approximate distance and the true distance to the true distance. The distributions of approximation errors using sketches are shown in fig. 2.19(a). For example, from the figure we can see that 90% of the approximation errors are within 0.32. We also repeat the experiment for larger sketch sizes $k = 80$ and $k = 160$. Larger sketch sizes give better approximations. For example, with $k = 160$, 90% of the approximation errors are within 0.22, while with $k = 80$, 90% of the approximation errors are within 0.19.

We know that the stock price data can be modeled by a random walk and they have a few large principal components. However, the price return time series is close to a white noise time series. The price return time series is defined as the time series derived from the price time series by computing



(a) price time series



(b) price return time series

Fig. 2.19. The approximation of distances between time series using sketches; 1 hour of stock data

the point-by-point price differences. That is, given a price time series $\mathbf{x}^n = (x(1), x(2), ..., x(n))$, its price return time series is $\mathbf{dx}^n = (x(2) - x(1), x(3) - x(2), ..., x(n) - x(n-1))$. There is no any principal component in the price return time series. Will the distance approximations using sketches work for these price return time series? We repeat the previous experiment on the price return time series. The results are shown in fig 2.19(b). We can see that the qualities of approximations using sketches are very close for both data sets.

One interesting observation is that the sketch size k depends only on the number of time series m, the approximation bound ϵ and the probability guarantee bound δ. The sketch size k does not depend on the length of the time series n. This makes random projections ideal for data reduction for a relatively small collection of time series with very long length. We repeat the previous experiment on a collection of stock price time series with longer lengths. The size of the collection of time series is the same as before, but the length of each time series in the collection is doubled, corresponding to two hours' data. From fig. 2.20, we can see that the same quality of approximation using sketches is achieved even though the time series are longer.

The size of the sketches can be large if the approximation requirement is high (small ϵ and δ). We can use the SVD to further reduce the dimensions of the sketch space. This works especially well if the time series data have some principal components after the random projection.

2.4.2 L_p Distance

In addition to Euclidean distance, sketches can also be used for approximation of L_p-distance. Although Euclidean distance is used most often for time series, other distance measures between time series can provide interesting results too.

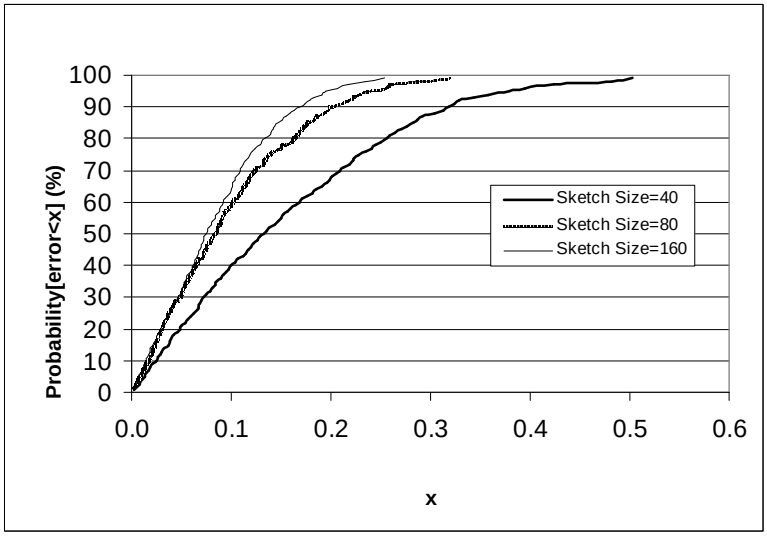
The approximation of L_p-distance is based on the concept of *stable distribution*[77]. A stable distribution $\mathcal{D}(p)$ is a statistical distribution with parameter $p \in (0, 2]$. An important property of stable distribution is as follows.

Definition 2.48 (stable distribution). *A distribution $\mathcal{D}(p)$ is stable if for any n real number $a_1, a_2, ..., a_n$ and n i.i.d. (independent and identically distributed) random variables $X_1, X_2, ..., X_n$ from $\mathcal{D}(p)$,*

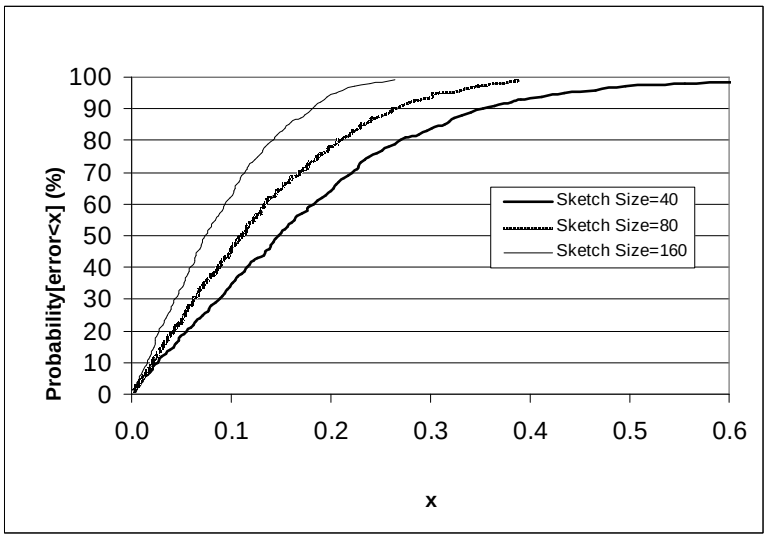
$$\sum_{i=1}^{n} a_i X_i \sim (\sum_{i=1}^{n} |a_i|^p)^{1/p} X, \tag{2.143}$$

i.e., $\sum_{i=1}^{n} a_i X_i$ has the same distribution as $(\sum_{i=1}^{n} |a_i|^p)^{1/p} X$, where X is drawn from $\mathcal{D}(p)$.

$\mathcal{D}(2)$ is a Gaussian distribution and $\mathcal{D}(1)$ is Cauchy distribution. Indyk[46] shows that one can construct $\mathcal{D}(p)$ sketches to approximate L_p distance.



(a) price time series



(b) price return time series

Fig. 2.20. The approximation of distances between time series using sketches; 2 hours of stock data

Lemma 2.49. *Given a collection C of m time series with length n, for any two time series $\mathbf{x}, \mathbf{y} \in C$, if $k = c\frac{\log(1/\delta)}{\epsilon^2}$ for some constant c, let $\mathbf{s}(\mathbf{x})$ and $\mathbf{s}(\mathbf{y})$be the $\mathcal{D}(p)$ sketches of $\mathbf{x}$ and $\mathbf{y}$ with size k, then*

$$(1-\epsilon) \leq \frac{B(p)median(|\mathbf{s}(\mathbf{x}) - \mathbf{s}(\mathbf{y})|)}{||\mathbf{x} - \mathbf{y}||_p} \leq (1+\epsilon) \tag{2.144}$$

holds with probability $1-\delta$, where $B(p)$ is some scaling factor.

In the above lemma, $|\mathbf{s}(\mathbf{x}) - \mathbf{s}(\mathbf{y})|$ is a vector with size k. The median of $|\mathbf{s}(\mathbf{x}) - \mathbf{s}(\mathbf{y})|$ is the median of the k values in the vector. It turns out that the scaling factor is 1 for both $p = 1$ (Cauchy distribution) and $p = 2$ (Gaussian distribution).

It is also possible to approximate Hamming distance L_0 between pairs of time series using stable distribution. The reader can refer to the recent result in [24].

In the time series data mining research, sketch-based approaches were used to identify representative trends [47, 25], to compute approximate wavelet coefficients[38],etc. Sketches have also many applications in streaming data management, including multidimensional histograms [90], data cleaning [28], and complex query processing [31, 27].

2.5 Comparison of Data Reduction Techniques

Having discussed the four different data reduction techniques, we can now compare them. This will help the data analysts choose the right data reduction technique. The comparison is summarized in table 2.4.

First we discuss the time complexity in computing the data reduction for each time series with length n.

- Using the Fast Fourier Transform, computing the first k DFT coefficients will take time $\min\big(O(n\log n), O(kn)\big)$.
- The time complexity for a DWT computation is lower, $O(n)$.
- The time complexity of SVD depends on the size of the collection of time series under consideration. For a collection of $m, m >> n$ time series, SVD takes time $O(m+n^3)$. The SVD for each time series requires $O(\frac{m}{n} + n^2)$ time. This is the slowest among all the data reduction techniques we discuss.
- The time complexity for the random projection computation is $O(nk)$, where k is the size of the sketches.

DFT, DWT and SVD are all based on orthogonal transforms. From the coefficients of the data reduction, we can reconstruct the approximation of the time series. By comparison, random projection is not based on any orthogonal transform. We cannot reconstruct the approximation of the time series. Pattern matching does not have to be information preserving.

In terms of distance approximation, DFT, DWT and SVD can be used for the approximation of only Euclidean (L_2) distance with one exception. Piecewise Aggregate Approximation (PAA), a transform closely related to the Discrete Haar Wavelet Transform, can handle any distance metric $L_p, p \neq 2$,.

Next we discuss the basis vectors using in these data reduction technique. For the DFT, the basis vectors are fixed to be vectors based on trigonometric functions. One particular benefit of using DWT is that one can choose from a vast number of wavelet families of basis vectors. SVD is desirable in many cases because the basis vectors are data dependent. These vectors are computed from the data to achieve optimality in reduce approximation error. But this also implies that we need to store the basis vectors in addition to the SVD coefficients if we want to reconstruct the time series. The basis vectors of the random projection are chosen, well, randomly.

To approximate a time series by a few coefficients, the DFT, DWT and SVD all require the existence of some principal components in the time series data. Random projection, by contrast, does not make any assumption about the data. It can work even for white noise. This makes random projection very desirable for time series data having no obvious trends such as price differencs in stock market data.

A particular drawback of DFT as a data reduction method is that the basis vectors of DFT do not have compact support. This makes it very hard for DFT to approximate time series having short term bursts or jumps. Most of the DWT basis vectors have compact support. Therefore, DWT can approximate a time series with jumps, but we need to choose a subset of coefficients that are not necessarily the first few DWT coefficients. SVD deals with the problem of discontinuity in the time series data more gracefully. If a short term bursts or jumps are observed at the same location of most time series, it will be reflected by the basis vectors of SVD at that location.

To conclude this chapter, in fig. 2.21 we present a decision tree to help you choose the right data reduction technique given the characteristics of your time series data.

2.6 Questions

1. a) Write a program in your favorite language (c, matlab, k, etc.) to implement the computation of Discrete Fourier Transform.
   b) Write a function that generates a random walk time series of length n.
   c) Compute the Discrete Fourier Transform of the random walk time series you generate.
2. Prove theorem 2.12(b).
3. a) Download the Fast Fourier Transform program from http://www.fftw.org/ and perform the FFT on the same time series you generate. Does it give you the same results as your codes?

Table 2.4. Comparison of data reduction techniques

Data Reduction Technique	DFT	DWT	SVD	Random Projection
Time Complexity	$n \log n$	n	$\frac{m}{n} + n^2$	nk
Based on Orthogonal Transform	Yes	Yes	Yes	No
Approximation of Time Series	Yes	Yes	Yes	No
L_p Distance	$p = 2$	$p = 2$	$p = 2$	$p = [0, 2]$
Basis Vectors	fixed one choice	fixed many choices	adaptive optimal	random
Require Existence of Principal Components	Yes	Yes	Yes	No
Compact Support	No	Yes	Yes	Not Relevant

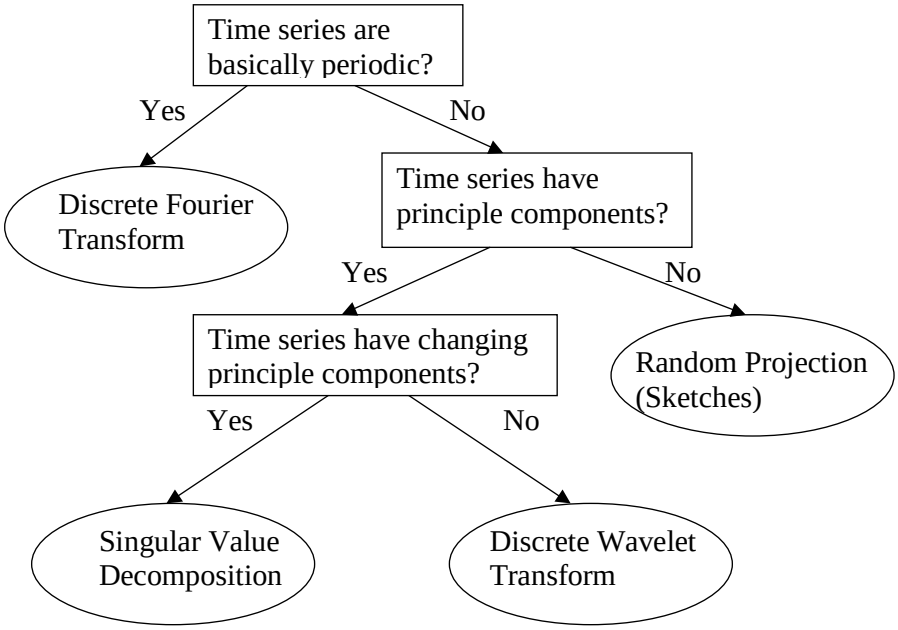


Fig. 2.21. A decision tree for choosing the best data reduction technique