Pattern Matching in Financial Time Series Data

Xian-ping Ge
xge@ics.uci.edu

December 6, 1998

Abstract

We define a novel criterion of pattern similarity in financial time series data. This criterion is robust, and concurs with human intuition. For fast pattern matching, we introduce a hierarchical piecewise linear representation of financial time series data. The piecewise linear approximation greatly reduces the complexity of the raw data, and the hierarchical representation keeps the structure of the time series data.

1 Introduction

1.1 Patterns in the stock times series data

In forecasting stock market, a popular method is to identify prototype patterns in the charts (plots of stock time series data), and forecast the future trend based on these chart patterns. Frequently used patterns include:

```
symmetrical triangles,
ascending triangles,
descending triangles,
Wedges,
flags and pennants,
rectangles,
head and shoulders.
```

Another related problem is to find patterns in the historical data that are similar to the current stock market pattern because many people believe that *history repeats itself.*

A even more interesting application of pattern matching in stock market time series data is to learn the frequently occurring patterns *directly from the data*, and build a model to forecast the stock market. This is exactly what people did to develop technical analysis methods.

1.2 Related work

There has been much work on defining pattern similarity measures for time series data. For example, in [1], the model of similarity of time sequences is introduced to capture the intuitive notion that two sequences should be considered similar if they have enough non-overlapping time-ordered pairs of subsequences that are similar. In [2], probabilistic models for local features (based on a prior distribution on expected deformation from a basic template) and global shapes (based on another prior on the relative locations of individual features) are defined, and this directly leads to an overall distance measure between sequence patterns based on prior knowledge. Unfortunately, these similarity models do not capture well the notion of chart pattern similarity as accepted by the market technicians.

Piecewise linear representation of time series data is a good technique to reduce the complexity of the raw data. Obviously this is an approximation of the original data, so it should try to minimize the approximation error. Frequently used error norms include mean squared error, maximum absolute deviation. Oriented toward different error norms, and other constraints (e.g. whether to allow discontinuous linear segments, or whether to require the end points of the linear segments to be from the original data points, etc), many linear segmentation algorithms has been developed, with a few ([4],[5]) being optimal with regard to its error norm and other constraints. For stock market time series data where we want to do pattern matching, the representation should assist in the analysis of structures in the data, and this problem is not properly addressed by the existing algorithms, although there has been work done on tree representation of waveforms (Relational Tree,[6]).

1.3 Our contribution

We define a novel criterion of pattern similarity in stock market time series data. This criterion is robust, and concurs with human intuition.

We also introduce a hierarchical piecewise linear representation of financial time series data, and an fast segmentation algorithm to convert the raw data to this representation.

Our new similarity model and the hierarchical piecewise linear representation allow flexible matching of chart patterns in the stock market time series data.

1.4 Organization of the paper

In section 2, we discuss what chart patterns are, and how to define similarity between two patterns. In section 3, we give a hierarchical piecewise linear representation of stock time series data for fast pattern matching. The matching algorithm, and some example results, are given in section 4. In section 5, we conclude with a summary and remarks about future directions.

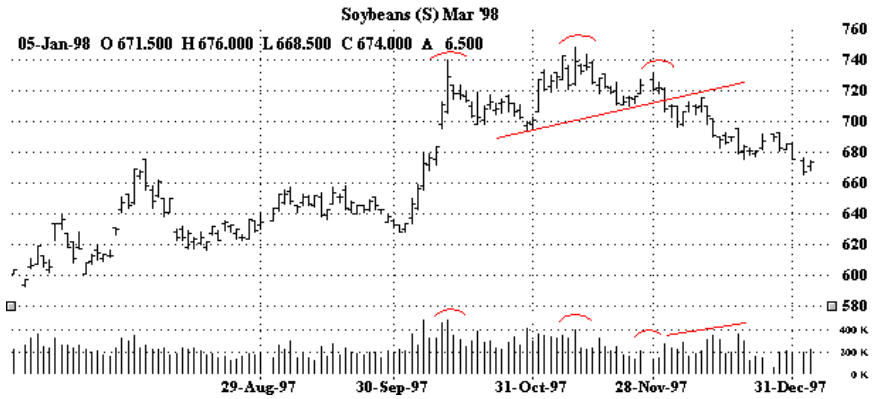

Figure 1: Daily price (and trade volume) of soybean

2 Chart Patterns

2.1 Some Examples

To give a concrete example of what we mean by patterns in time series data, let us look at Fig. 1, the bar chart of Soybean (S) March '98. [1]. For each trading day, a bar is drawn for the open, high, low, and close prices, and the trading volume is given at the bottom. In this figure, a chart pattern called "*head and shoulders*" occurs. A "head and shoulders" pattern often gives the signal of trend reversal. It is illustrated in Fig. 2.

There are many such chart patterns that are of interest to traders using technical analysis. For more examples,

2.2 Properties of chart patterns

There are many different time scales on which to plot stock market time series data. There are daily, weekly, monthly, yearly charts (i.e. each data point correspond to a day, a week, a month, a year, respectively); there are also intro-day charts for every 1 minute, 5 minutes etc. Of course we can construct the daily charts from the 1-minute charts of corresponding days. Figuratively, we "compress" 1-minute charts to get the daily charts. Thus, the patterns we define in the charts should be invariant to the stretching/compressing in the time dimension. In the same way, as different stocks may have different price ranges, it should also be invariant to the "amplitude" rescaling.

[1] Figures 1-4 are taken from `http://www.chartpatterns.com/`

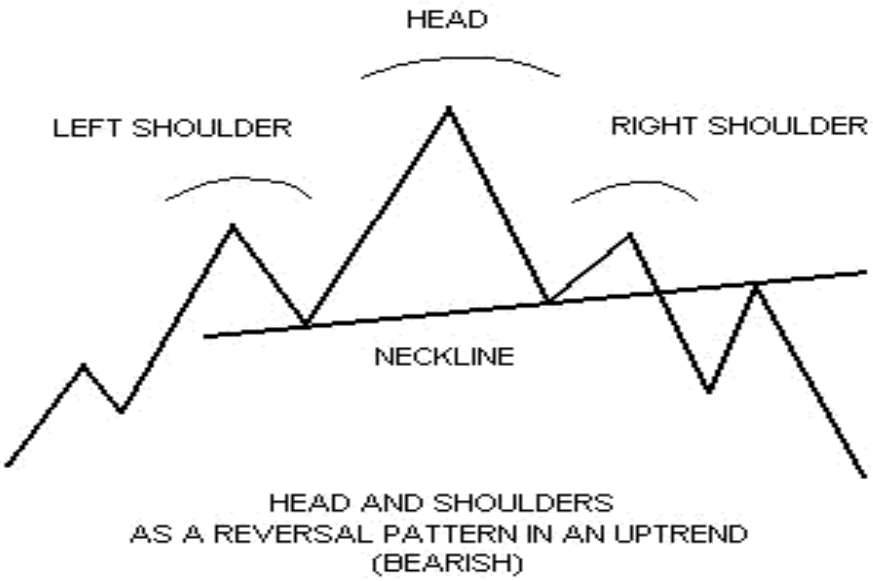

Figure 2: Chart pattern: "Head and shoulders"

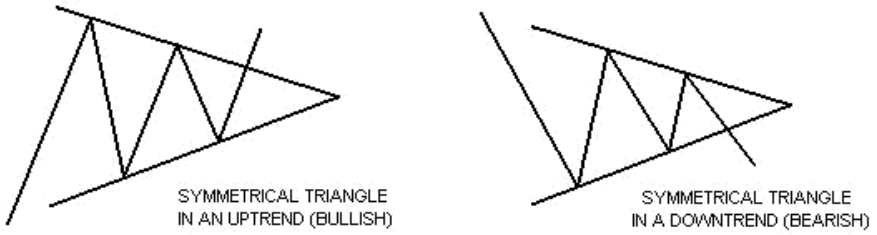

Figure 3: Chart pattern: "Symmetric Triangle"

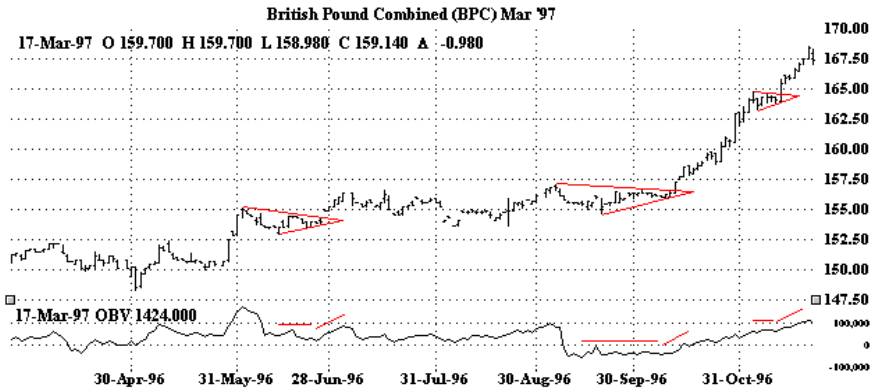

Figure 4: Daily price (and trade volume) of British Pound Combined (BPC) March '97

Related with the varying time scales are the concepts of "long term trends", "short term trends". For example, in the monthly charts, we may find the price of stock going up in a "straight" line; if we "magnify" the chart to the daily charts, we can find out that the "straight" line is actually composed of many ups and downs. In the same way, an upward movement in the daily charts will have many ups and downs in the more detailed 1-minute charts. The moral is this is that there is a hierarchical structure in the time series data. We will give a hierarchical piecewise linear representation in section 3.

An interesting observation can be made from the "*head and shoulders*" in Fig. 2. The shape of the pattern is defined by the peaks and valleys which obeys some rules about their relative positions. For example, the two "*shoulders*" should be lower than the head. From the "*neckline*", we can see that the two valleys at the shoulders and the rightmost peak have an increasing sequence of heights. In addition, the peaks and valleys correspond to the highs and lows of the stock prices (or indices), and they have a very important significance in technical analysis.

2.3 Definition of chart pattern similarity

In accordance with our preceding observations of chart patterns, here we give a novel definition of chart pattern similarity.

Suppose p and q are the piecewise linear representation of two chart patterns. For simplicity, suppose p and q have the same number of line segments.

The piecewise linear representations are in the form of

$$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$$

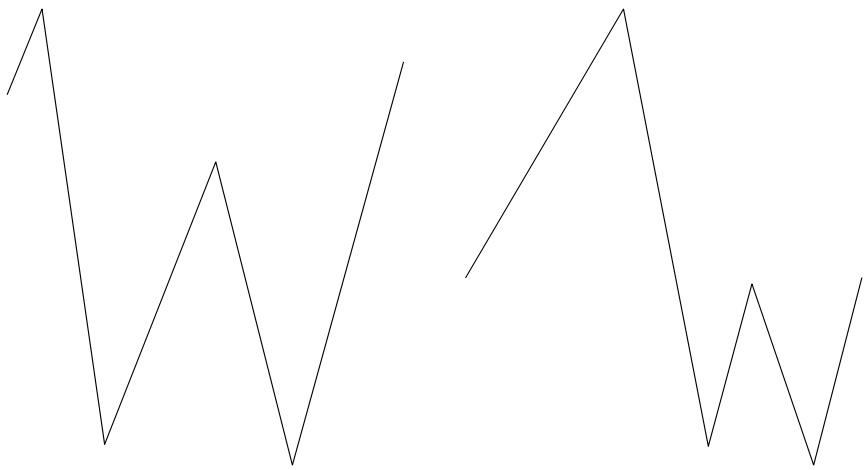

Figure 5: An example of matched patterns

where $x_1 < x_2 < \ldots < x_n$ are the time components. Connecting adjacent data points we will have the piecewise linear representation.

With the data points numbered according to time (x_i), if we sort the y_i's, we will have something like

$$y_{i_1}, y_{i_2}, \ldots, y_{i_n},$$

where $i_1, i_2, \ldots, i_n$ is a permutation of $1, 2, \ldots, n$.

Now we say that the permutation $i_1, i_2, \ldots, i_n$ represents the relative positions of the end points of the line segments, and we use it to decide if two patterns are similar by looking at if they have the same permutation of the y_i's.

Because we stipulate only the orders the x_i's and y_i's, and we have said nothing about their actual values, this can solves easily the problem time scale stretching/compressing and amplitude rescaling. Additional flexibilities will emerge from our hierarchical representation which allows us to merge small segments into one large segment. And this enables us to compare two patterns of different number of (basic) segments.

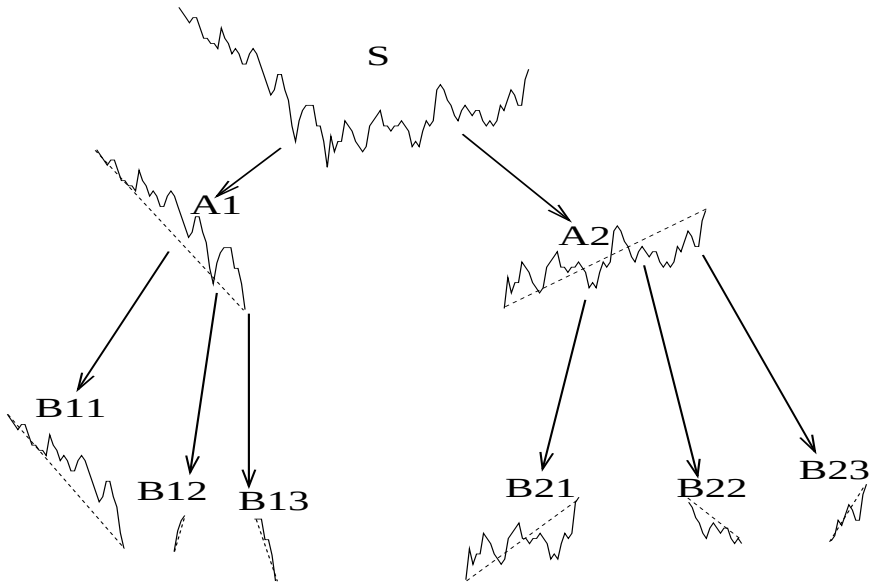

Figure 6: Hierarchical piecewise linear representation of time series data

3 Hierarchical piecewise linear representation of time series data

In Fig. 6 we give an example of hierarchical piecewise linear representation of time series data S. S is broken into two "linear" segments, the upward A_1 and the downward A_2. In principle, we can use A_1 and A_2 to approximate S; the approximation error may be large, but A_1 and A_2 give us a general idea of the shape of S. To give more detailed description, we can further break A_1 and A_2; here A_1 is broken into B_{11}, B_{12}, B_{13}, and A_2 is broken into B_{21}, B_{22}, B_{23}. And this process is repeated until our desired "precision" is achieved, i.e. the line connecting the end points of segment is very close the original data points. This closeness can be measured by the maximum distance from the data points to the line.

Why should we want to use this hierarchical representation? The answer is that it concisely encodes all the patterns in the data. Take for example the time series data in Fig. 6, we can say that $A_1 - A_2$,$B_1 2 - B_1 3 - A_2$,$A_1 - B_2 1 - B_2 2$, ... are all patterns that can be potentially interesting. This can be seen in Fig. 7.

Please note that for the segments A_1 and A_2, the maximum and minimum both coincide with the end points. This is because we require a segment as a whole to be either upward or downward; Otherwise, we should break this

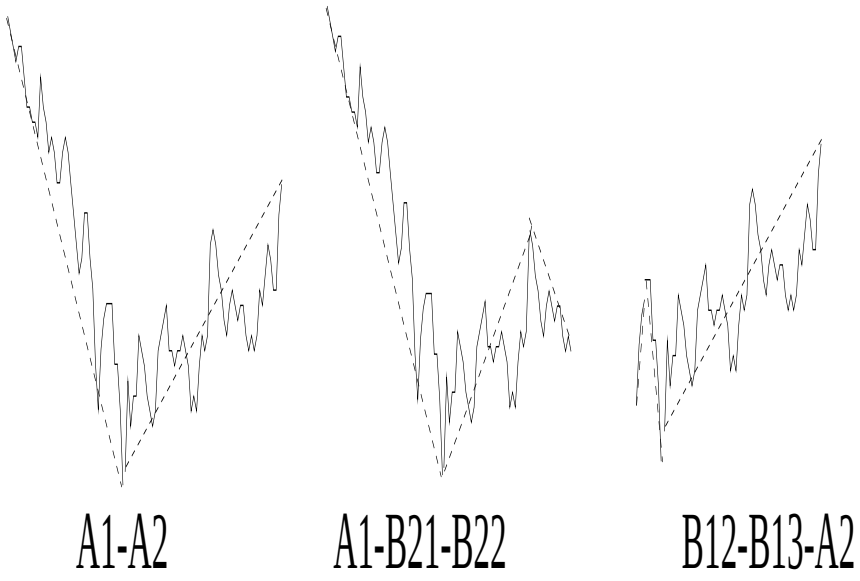

Figure 7: Different patterns in Fig. 6

segment at the maximum or minimum points into a number of smaller segments, and *replace* this segment with these smaller segments. Please note the word *replace* here; this is different from the case where we break A_1 into B_{11}, B_{12}, B_{13}, and put B_{11}, B_{12}, B_{13} directly under A_1 in the hierarchy. But here when we break a segment at the maximum or minimum point, we say that this segment is *illegal*, it can not stay in our hierarchy. In other words, it has to be *replaced* with some smaller segments such that each of these smaller segments meet our requirement of maximum and minimum occurring at end points.

```
Procedure BuildTree(S).
  Input: segment S.

  Let S be represented as x[1..n], y[1..n].

  if (MAX(y[1..n]) == y(1) OR MAX(y[1..n]) == y(n))
     AND (MIN(y[1..n]) == y(1) OR MIN(y[1..n]) == y(n))
  then /* this is a valid segment; */
     Create a node in the hierarchy for this segment;

     Draw a line between (x[1],y[1]) and (x(n), y(n));
     maxd = maximum distance of (x[i],y[i]) to the line;
```

```
   if (maxd < tolerance)
   then
      /* this segment is good enough; no further work */
   else
      Let (x[j],y[j]) be the point with maximum distance to the line.
      Break the segment at (x[j],y[j]) into S1, S2;
      PARENT(S1) = S;
      PARENT(S2) = S;
      BuildTree(S1);
      BuildTree(S2);
   end
else
   Break the segment at the maximum and/or minimum point(s)
        into smaller segments S1, S2, ..., Sm;

   /* Replace this segment with these smaller segments */
   for i=1 to m
        PARENT(Si)=PARENT(S);
   end
   delete(S);

   for i=1 to m
        BuildTree(Si);
   end
end
```

4 The Pattern Matching Algorithm

In the preceding sections, we introduce a pattern similarity model, and a hierarchical piecewise linear representation of time series data. In this section, we give an algorithm to match a pattern in a time series data.

The central idea of the algorithm is like this:

> Let P be represented by its component edges $E_1, E_2, ..., E_m$ in its piecewise linear representation. Suppose we have already matched the first k edges of the pattern with the interval $[x_{i_1}, x_{i_2}]$, and now we want to extend the matching by one edge further. We try each segment in the hierarchy that starts at x_{i_2}; if this new segment matches the $(k+1)$st edge of the pattern, then continue, else try next segment in the hierarchy that starts at x_{i_2}. And if there is no such segment left, then backtrack, i.e. discard the current matching with the kth edge of the pattern, and try another alternative.

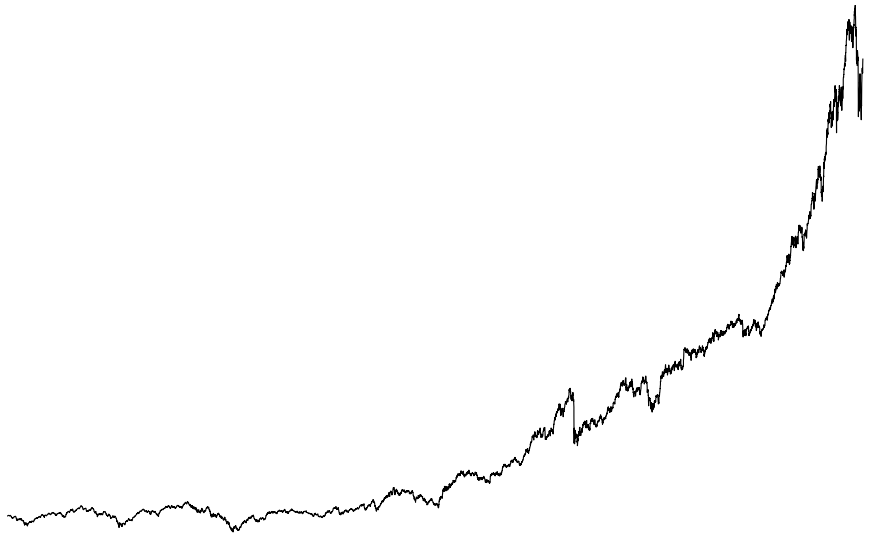

Figure 8: New York Stock Exchange (NYSE) Daily Composite Index Closes

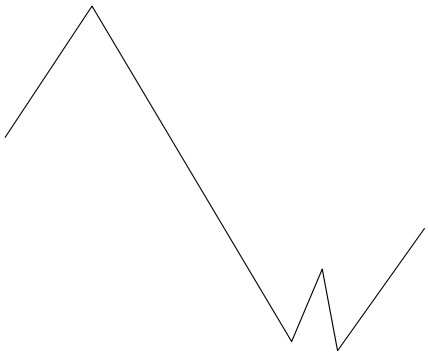

Figure 9: A pattern that we want to match in Fig. 8.

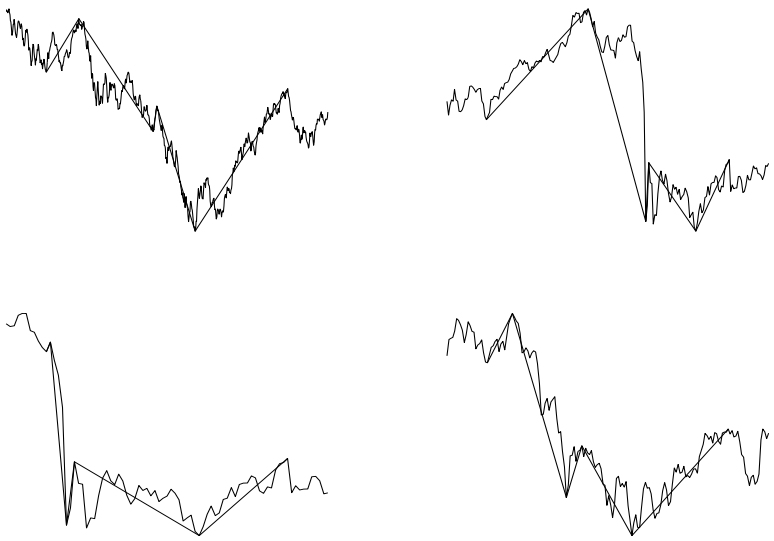

Figure 10: Matching Results

Let us look at an example. Fig. 7 is the chart of New York Stock Exchange (NYSE) daily composite index closes. We want to find patterns that are similar to that of Fig. 8. The matching results are given in Fig. 9.

5 Conclusion and Future Work

In this paper, we give a novel algorithm for matching chart patterns in stock market time series data. This matching method is very flexible because (a) the match is defined in terms of the relative positions (i.e. ordering) of the peaks and valleys in the patterns, and (b) we can easily go up and down the hierarchy for suitable segments to use in matching. The results of such a matching can be thought of as a "pattern family". An interesting problem could be to define the distance between two different pattern families. This may lead to some elegant "language" for describing "*head and shoulders*", "*ascending triangles*", and other popular chart patterns. An even more interesting direction would be to find all the patterns in, say, all the historical trading data of New York Stock Exchange, and cluster the patterns. This may give us the frequently occurring patterns, and we can now investigate how these patterns may help in forecasting stock market. For example, we can compute the probability that the price will go up given that a certain pattern has occurred.

References

1 Rakesh Agrawal et al, 'Fast Similarity Search in the presence of Noise, Scaling and Translation in Time-Series Databases,' *Proceedings of the 21st VLDB Conference*, Zurich, Switzerland 1995.

2 Eamonn Keogh and Padhraic Smyth, 'A Probabilistic Approach to Fast pattern matching in Time Series Databases,' *KDD '97.*

3 D. J. Berndt and J. Clifford. 'Using dynamic time warping to find patterns in time series,' *KDD-94: AAAI Workshop on Knowledge Discovery in Databases*, pages 359-370, Seattle, Washington, July 1994.

4 Hiroshi Imai and Masao Iri, 'An optimal Algorithm for Approximating a Piecewise Linear Function,' *Journal of Information Processing* Vol.9, No.3, 1986, pp.59-62

5 Y.Zhu, L.D.Seneviratne, 'Optimal polygonal approximation of digitized curves,' *IEE proceedings. Vision, image, and signal processing.* Vol: 144, no. 1, Feb 1997, pp. 8-14

6 R. W. Erich and J. P. Foth, 'Representation of random waveform by relational trees,' *IEEE Trans. Comput.*, vol. C-25, pp. 725-736, July 1976.