

An Introduction to Similarity Analysis Using SAS[®]

Michael Leonard, Jennifer Sloan, Taiyeong Lee, Bruce Elsheimer
SAS Institute Inc., Cary, NC

ABSTRACT

Web sites and transactional databases collect large amounts of time-stamped data related to an organization's suppliers and/or customers over time. Mining these time-stamped data can help business leaders make better decisions by enabling them to listen to their suppliers or customers via their transactions collected over time. A business can have many suppliers and/or customers and might have a set of transactions associated with each one. However, each set of transactions might be quite large, making it difficult to perform many traditional data mining tasks. This paper proposes techniques for large-scale similarity analysis that uses similarity measures combined with automatic time series analysis and decomposition. After similarity analysis, traditional data mining techniques can then be applied to the similarity analysis results along with other profile data. This paper demonstrates how to use these techniques with SAS/ETS[®] software.

INTRODUCTION

Given two ordered numeric sequences (input and target), such as two time series, a similarity measure is a metric that measures the distance between the input sequence and the target sequence while taking into account the ordering. Similarity measures can be computed between the input sequence and the target sequence in addition to similarity measures that "slide" the target sequence with respect to the input sequence. The "slides" can occur by observation index (sliding-sequence similarity measures) or by seasonal index (seasonal-sliding-sequence similarity measures).

In computing the similarity measure between two time series, you need tasks for preparing the time series data, transforming the time series, normalizing sequences, scaling sequences, and computing metrics or measures.

For preparing time series data, you can use smoothing (simple, trend, seasonal, or all three), decomposition/adjustment (seasonal, detrending, causal, and event adjustments), and other methods. For transforming the input and target time series, you can use functional transformations (log, square root, logistics, and Box-Cox), differencing transformations (simple and seasonal), and other transformation functions. For normalizing the input and target sequences, you can use standard and absolute normalization and other normalization functions. For scaling the input sequence to the target sequence, you can use standard and absolute scaling and other scaling functions. For computing metrics, you can use fixed and dynamic time-warped metrics for squared and absolute deviations in addition to other metrics.

Similarity measures can be used to compare a single target sequence to many other input sequences. This situation arises in *time series search and retrieval* or ranking. For example, given a single target sequence, you can find input sequences that are "similar" or "close" to the target. Search and retrieval and analogies are important for new product

forecasting and analogous time series forecasting. These techniques are useful when a large number of historical time series are available.

Similarity measures can be used to compare a single input sequence to several other representative target sequences. This situation arises in *time series classification*. For example, given a single input sequence, you can classify the input sequence by finding the “most similar” or “closest” target sequence. This analysis can be repeated to classify large numbers of input sequences.

Similarity measures can be computed between several sequences to form a similarity matrix. For example, given K time sequences, a $(K \times K)$ symmetric matrix can be constructed whose ij th element contains the similarity measure between the i th and j th sequence. Clustering techniques can then be applied to the similarity matrix. This situation arises in *time series clustering*.

Sliding similarity measures (observational or seasonal index) can be used to compare a single target sequence to subsequences of many other input sequences on a sliding basis. This situation arises in historical time series analogies. For example, given a single target series, you can find the times in the history of the input sequence that are similar while preserving the ordering or seasonal indices.

SCOPE

This paper focuses on mining (discrete) time-stamped data generated by business and economic activity. The techniques described in this paper are less applicable to voice, image, multimedia, medical, and scientific data or continuous-time data.

OVERVIEW OF SIMILARITY ANALYSIS

This section provides a theoretical background on similarity measures. It is intended to provide the analyst with motivation and orientation.

This section covers the steps involved in computing similarity measures, starting with the raw data set. A time series that contains both an input and a target sequence is required to perform this analysis. Next, the distance metrics are computed, and a matrix is built using these metrics. Constraints on both the input and target sequences are considered before selecting the path to traverse the matrix from the beginning of the matrix $(1, 1)$ to the end of the matrix (N_x, N_y) . The shortest path having the least cost is chosen, and statistics are calculated for this path. This iterative process is described in detail in subsections that follow.

Suppose that there are two time sequences, *input* and *target*, that you want to compare. Let X represent the *input* sequence, and let Y represent the *target sequence*. Since there might be many input and target sequence values, it is desirable to construct a *scalar measure* for this comparison; in this case, we will use a *similarity measure*. There are

many similarity measures that you can use to compare an input sequence to a target sequence. Let $Sim(X, Y)$ represent such a similarity measure.

TIME SEQUENCES

Time sequences are observed data ordered with respect to time. Let x_i represent the *input sequence value* at input sequence index $i = 1, \dots, N_x$, where N_x represents the input sequence length. Let $X = \{x_i\}_{i=1}^{N_x}$ represent the *input sequence vector*. Let y_j represent the *target sequence value* at target sequence index $j = 1, \dots, N_y$, where N_y represents the target sequence length. Let $Y = \{y_j\}_{j=1}^{N_y}$ represent the *target sequence vector*. In this paper, the input and target sequences represent a univariate (scalar) sequence, but many of the concepts apply to multivariate (vector) sequences.

Simple Example

This section introduces a simple time sequence example that is used throughout the rest of this paper. Let $X = \{3, 5, 3, 3, 3, 6, 3, 8\}$ be an input sequence vector with $N_x = 8$. Let $Y = \{2, 4, 6, 7, 3, 8, 9, 3, 10, 11\}$ be a target sequence vector with $N_y = 10$. The input and target sequence are plotted in Figures 1 and 2.

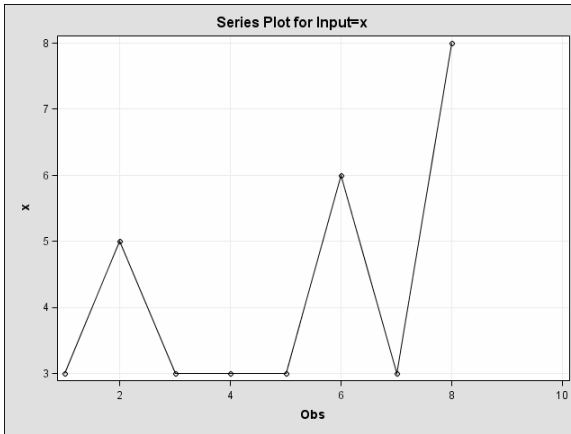


Figure 1. Input Sequence Plot

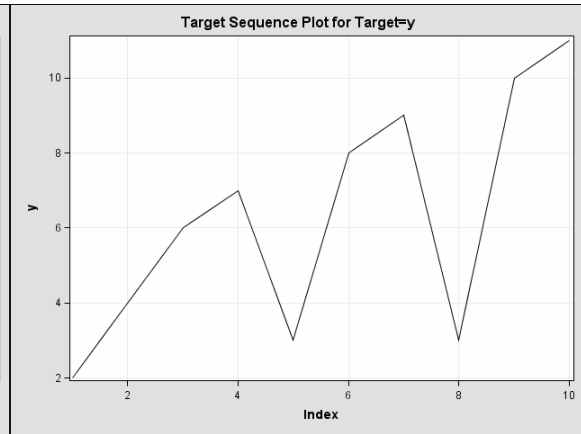


Figure 2. Target Sequence Plot

Notice that the two time sequence plots are somewhat similar but differ in length and timing. Both sequences have two phases of upward movements which are followed by downward movements and a final phase of upward movement. A formal approach of similarity analysis is illustrated by using this simple example.

DISTANCE METRICS

There are many distance metrics that you can use to compare the input and target time sequence values. Let $d_{i,j} = d(x_i, y_j)$ represent such a *distance metric* between an input sequence value, x_i , and a target sequence value, y_j . The following examples illustrate two of the many possible distance metrics:

$$\begin{aligned} \text{Squared deviation} \quad d(x_i, y_j) &= (x_i - y_j)^2 \\ \text{Absolute deviation} \quad d(x_i, y_j) &= |x_i - y_j| \end{aligned}$$

The choice of the distance metric varies with the problem under consideration. Many statistical analysts like to think in terms of squared deviations. Many business analysts like to think in terms of absolute deviations. Other metrics might satisfy other people's needs.

DISTANCE MATRIX

There are $N_x \times N_y$ possible distance metrics that can be computed between the input and target sequence values. Let D represent the $N_x \times N_y$ matrix of all possible distance measures between the input and target sequence values, where $d_{i,j}$ represents the (i,j) th element of the *distance matrix*, where j is the row index and i is the column index of the matrix.

Since the distance matrix D contains many elements, similarity measures must summarize this matrix into a single number: $Sim(X, Y) = Norm(D)$. For example, you could summarize the matrix in one of the following ways:

- $Norm(D) = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} d_{i,j}$ simply sums all of the elements
- $Norm(D) = \frac{\sum_{i=1}^{N_x} \sum_{j=1}^{N_y} d_{i,j}}{(N_x N_y)}$ averages all of the elements
- $Norm(D) = \min_{i,j} \{d_{i,j}\}$ uses the minimum

However, these similarity measures would ignore the ordering of the time sequence values. For example, the distances $d_{1,1}$ and d_{1,N_y} are treated with equal importance in these similarity measures even though they compare sequence values that might be far apart with respect to their timing.

DISTANCE MATRIX PATHS

For another approach, similarity measures can compare the various *paths* through the distance matrix, starting with element $(1,1)$ and ending with element (N_x, N_y) , and determine the path that minimizes the sum or average of the distances along the path or the *cost* of traversing the path.

A more complete description of paths and costs is given later. Suffice it to say that paths are a way to step through the distance matrix from element $(1,1)$ to element (N_x, N_y) without stepping on the same element twice. Diagonal path movements represent direct maps. Vertical path movements represent compression of the target sequence. Horizontal path movements represent expansion of the target sequence.

PENALIZED DISTANCE MATRICES

Similarity measures that exploit the ordering of the time sequences can penalize distances that are far apart with respect to their timing. Modifying the distance matrix, D , so that distances that are far apart with respect to their timing are set to missing or infinite values is one way to penalize those distances. Another way is to use a weighting scheme, but that method is beyond the scope of this paper. You can penalize distances by using *warping limits*, which limit the expansion and/or compression of the target sequence.

Input sequence indices, i , that are much *larger* than the target indices, j , usually represent *expansion* of the target sequence. Input sequence indices, i , that are much *smaller* than the target indices, j , typically represent *compression* of the target sequence.

With *warping limits*, the distance metric can be penalized as follows:

- $d_{i,j} = d(x_i, y_j)$ (i, j) between the warping limits
- $d_{i,j} = \infty$ (i, j) outside the warping limits

A more complete description of expansion and compression is given later. Suffice it to say that limiting the warping will provide such a penalty.

PENALIZED DISTANCE MATRICES AND PATHS

Yet another approach is to combine paths with warping limits. These similarity measures minimize the cost of traversing the path while penalizing for timing.

Distance Matrix Example

Continuing the preceding simple example, there are 8×10 possible distance metrics that can be computed between the input and target sequence values. The distance matrix is illustrated in Figure 3. Given an expansion limit of 3 and a compression limit of 2, the distance matrix with limits is illustrated in Figure 4. A possible distance matrix path is illustrated in Figure 5. A possible distance matrix path with limits is illustrated in Figure 6.

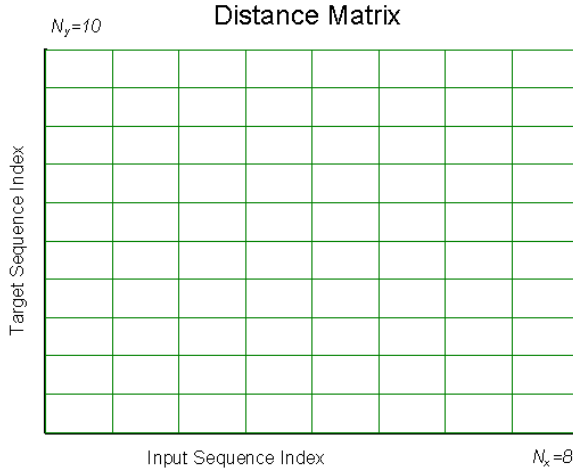


Figure 3. Distance Matrix

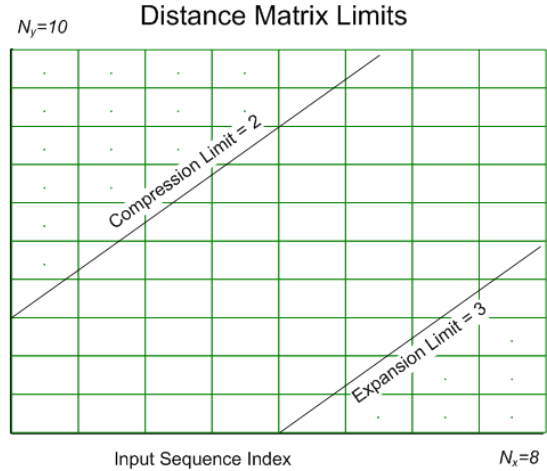


Figure 4. Distance Matrix Limits

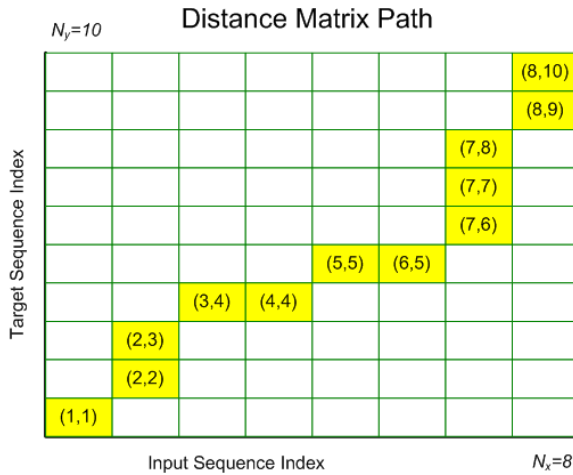


Figure 5. Distance Matrix Path

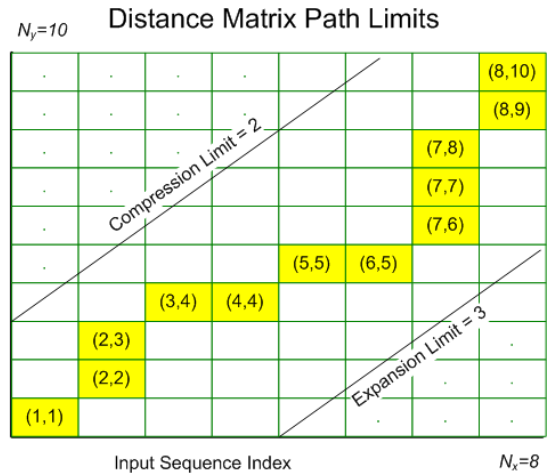


Figure 6. Distance Matrix Path Limits

In Figure 5, the *diagonal* movement from element (1,1) to element (2,2) represents a *direct* mapping between the target and input sequence; the *vertical* movement from element (2,2) to element (2,3) represents a *compression* mapping of the target sequence with respect to input sequence; the *horizontal* movement from element (3,4) to element (4,4) represents an *expansion* mapping of the target sequence with respect to input sequence.

A *path* is a sequence of direct (diagonal), compression (vertical), and expansion (horizontal) mappings from the lower-left corner, element (1,1), to the upper-right corner, element (N_x, N_y) , of the distance matrix. When penalties are applied, the path might be restricted to certain elements of the distance matrix.

WARPING LIMITS

Expansion of the target sequence is the same as compression of the input sequence, and vice versa. Likewise, compression of the target sequence is the same as expansion of the input sequence, and vice versa. In the discussion that follows, the warping limits are expressed *relative to the target sequence*.

EXPANSION LIMITS

Sometimes it is desirable to limit the amount of expansion of the target sequence with respect to the input sequence. The *expansion limit*, l_e , is used to restrict the amount of expansion. Expansion *must* occur when the input sequence length is *greater* than the target sequence length, $N_x > N_y$. Additionally, the input sequence length bounds the maximum allowable expansion. The *valid* expansion limits are described by the following interval:

$$\max(0, (N_x - N_y)) \leq l_e < N_x$$

where $l_e = 0$ represents *no* expansion, $l_e = \max(0, (N_x - N_y))$ represents the minimum *required* expansion, and $l_e = (N_x - 1)$ represents maximum *allowable* expansion. There might be other expansion limits that are not linear (piecewise linear or nonlinear limits).

The expansion limit can be represented by the following linear equation:

$$(j-1) = \left(\frac{N_y - 1}{N_x - 1} \right) (i-1 - l_e)$$

where $(N_y - 1)/(N_x - 1)$ and l_e represent the slope and x -intercept of the expansion limit line, respectively.

The preceding linear restriction can be represented by the following inequalities:

$$\begin{aligned} (j-1) &\geq \left(\frac{N_y - 1}{N_x - 1} \right) (i-1 - l_e) \\ (i-1 - l_e)(N_y - 1) &\leq (j-1)(N_x - 1) \\ \left(\frac{N_y - 1}{N_x - 1} \right) &\leq (j-1)/(i-1 - l_e) \end{aligned}$$

If $j = 1$, then $1 \leq i \leq (l_e + 1)$; if $l_e = 0$, then $i = 1$; and if $l_e = N_x - 1$, then $1 \leq i \leq N_x$.

If $i = N_x$ then $j > \left(\frac{N_y - 1}{N_x - 1} \right) (N_x - 1 - l_e)$; if $l_e = 0$, then $j = N_y$; and if $l_e = (N_x - 1)$, then $1 \leq j \leq N_y$.

COMPRESSION LIMITS

Sometimes it is desirable to limit the amount of compression of the target sequence with respect to the input sequence. The *compression limit*, l_c , is used to restrict the amount of compression. Compression *must* occur when the input sequence length is *less* than the target sequence length, $N_x < N_y$. Additionally, the target sequence length bounds the maximum allowable compression. The *valid* compression limits are described by the following interval:

$$\max(0, (N_y - N_x)) \leq l_c < N_y$$

where $l_c = 0$ represents *no* compression, $l_c = \max(0, (N_y - N_x))$ represents the minimum *required* compression, and $l_c = (N_y - 1)$ represents the maximum *allowable* compression. There might be other compression limits that are not linear (piecewise linear or nonlinear limits).

The compression limit can be represented by the following linear equation:

$$(j - 1 - l_c) = ((N_y - 1)/(N_x - 1))(i - 1)$$

where $((N_y - 1)/(N_x - 1))$ and l_c represent the slope and y-intercept of the compression limit line, respectively.

The preceding linear restriction can be represented by the following inequalities:

$$\begin{aligned} (j - 1 - l_c) &\leq ((N_y - 1)/(N_x - 1))(i - 1) \\ (j - 1 - l_c)(N_x - 1) &\leq (i - 1)(N_y - 1) \\ (j - 1 - l_c)/(i - 1) &\leq ((N_y - 1)/(N_x - 1)) \end{aligned}$$

If $i = 1$, then $1 \leq j \leq (l_c + 1)$; if $l_c = 0$, then $j = 1$; and if $l_c = N_y - 1$, then $1 \leq j \leq N_y$.

If $j = N_y$, then $i > ((N_y - 1)/(N_x - 1))(N_y - 1 - l_c)$; if $l_c = 0$, then $i = N_x$; and if $l_c = (N_y - 1)$, then $1 \leq i \leq N_x$.

WARPING LIMIT INEQUALITIES

The preceding compression and expansion limit inequalities can be combined to form the *warping limits*:

- Expansion limit: $(i - 1 - l_e)(N_y - 1) \leq (j - 1)(N_x - 1)$
- Compression limit: $(j - 1 - l_c)(N_x - 1) \leq (i - 1)(N_y - 1)$

When $N_x = N_y$, these warping limits simplify to $-l_c \leq (i - j) \leq l_e$. When $N_x = 1$, these warping limits simplify to $i = 1$ and $1 \leq j \leq N_y$. When $N_y = 1$, these warping limits simplify to $j = 1$ and $1 \leq i \leq N_x$.

With these warping limits, the distance metric is penalized as follows:

- $d_{i,j} = d(x_i, y_j)$ (i, j) between the warping limits
- $d_{i,j} = \infty$ (i, j) outside the warping limits

Warping Limits Example

Continuing the preceding simple example, a valid expansion limit must be contained in the interval, $0 \leq l_e < 8$, and a valid compression limit must be contained in the interval, $2 \leq l_c < 10$. The valid warping limits are illustrated in Figure 7. Notice that in the upper left corner of the matrix, there is a maximum compression limit of $N_y - 1$ and in the lower right corner of the matrix, there is a maximum expansion limit of $N_x - 1$. Specified warping limits that are used in examples that follow are illustrated in Figure 8.

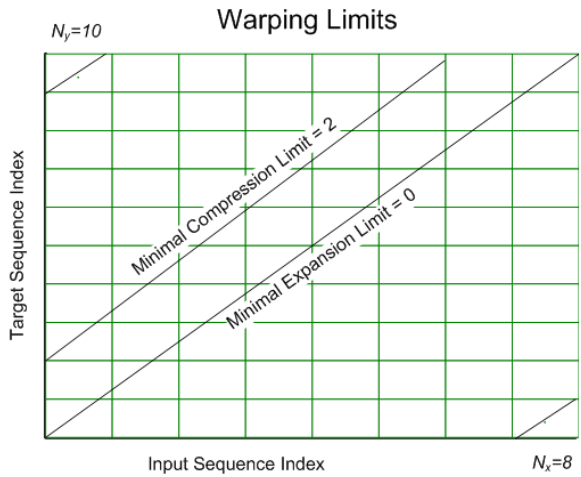


Figure 7. Warping Limits

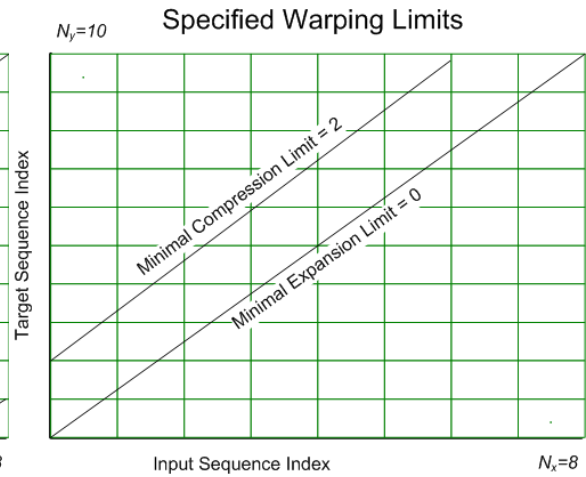


Figure 8. Specified Warping Limits

PATHS

A *path* steps through the elements of the distance matrix, starting with element (1,1) and ending with element (N_x, N_y) . The steps along the path preserve the ordering of the time sequences by not revisiting the same matrix elements and by not moving backward with respect to time. There are many possible paths through the distance matrix.

Let Π represent the set of all possible paths through the distance matrix, D . Let $\pi \in \Pi$ represent one such path.

In computing a similarity measure, the goal is to find a path that *minimizes the cost* of traversing the path. A more complete description of costs is given later. Suffice it to say that the goal is finding a path that has smaller distances between the input and target sequence values.

PATH FUNCTION

The *path function* assigns an input sequence index and a target sequence index for a given path index associated with path $\pi \in \Pi$. Given a path index, $p = 1, \dots, P$, the path function, $(i_p, j_p) = Path_\pi(p)$, returns the input sequence index, $i_p \in \{1, \dots, N_x\}$, and target sequence time index, $j_p \in \{1, \dots, N_y\}$, associated with the path index. The path length through the distance matrix is P , where $\max(N_x, N_y) \leq P < N_x + N_y$. For notational convenience, let $Path(p)$ be any possible valid path function.

To preserve the timing of the two sequences, a valid path function has the following properties:

If $(i_p, j_p) = Path(p)$ and $(i_{p+1}, j_{p+1}) = Path(p+1)$, then $(i_{p+1} - i_p) \in \{0, 1\}$, $(j_{p+1} - j_p) \in \{0, 1\}$, and one of the following occurs:

- A direct map occurs when
there is a *diagonal* movement $(i_{p+1} - i_p) = (j_{p+1} - j_p) = 1$
 $Path(p+1) - Path(p) = (1, 1)$
- A compression map occurs when
there is a *vertical* movement $(i_{p+1} - i_p) = 0$ and $(j_{p+1} - j_p) = 1$
 $Path(p+1) - Path(p) = (0, 1)$
- An expansion map occurs when
there is a *horizontal* movement $(i_{p+1} - i_p) = 1$ and $(j_{p+1} - j_p) = 0$
 $Path(p+1) - Path(p) = (1, 0)$
- An invalid map occurs when
there is *no* movement $(i_{p+1} - i_p) = 0$ and $(j_{p+1} - j_p) = 0$
 $Path(p+1) - Path(p) = (0, 0)$

Compression and expansion are expressed relative to the target sequence in the preceding properties. An invalid path function has no movements or $(i_{p+1} - i_p) \notin \{0, 1\}$ and/or $(j_{p+1} - j_p) \notin \{0, 1\}$. These represent backward movements or jumps with respect to time. Once again, a valid path must preserve the timing.

Path Example

Continuing the preceding simple example, the path length is $P = 12$.

The *path plot* is illustrated in Figure 9. The horizontal axis represents the input sequence index, i , and the vertical axis represents the target sequence index, j . The dots represent

the path coordinates $(i_p, j_p) = Path(p)$ for $p = 1, \dots, P$. The upper parallel line represents the compression limit, l_c , and the lower parallel line represents the expansion limit, l_e . This plot visualizes the path through the distance matrix. Vertical movements indicate compression and horizontal movements represent expansion of the target sequence with respect to the input sequence. This plot is useful for visualizing the amount of expansion and compression along the path.

The *time warp plot* is illustrated in Figure 10. The horizontal axis represents the (input and target) sequence index, $k = 1, \dots, N$, where $N = \max(N_x, N_y)$. The upper line plot represents the target sequence, y_j for $j = 1, \dots, N_y$. The lower line plot represents the input sequence, x_i for $i = 1, \dots, N_x$. The lines that connect the input and target sequence values represent the *mapping* between the input and target sequence indices along the path: $(i_p, j_p) = Path(p)$ for $p = 1, \dots, P$. This plot visualizes the *warping* of the time index with respect to the input and target sequence values. Expansion of a single *target* sequence value occurs when it is mapped to more than one input sequence value. Expansion of a single *input* sequence value occurs when it is mapped to more than one target sequence value. This plot is useful for visualizing the mapping between the input and target sequence values along the path.

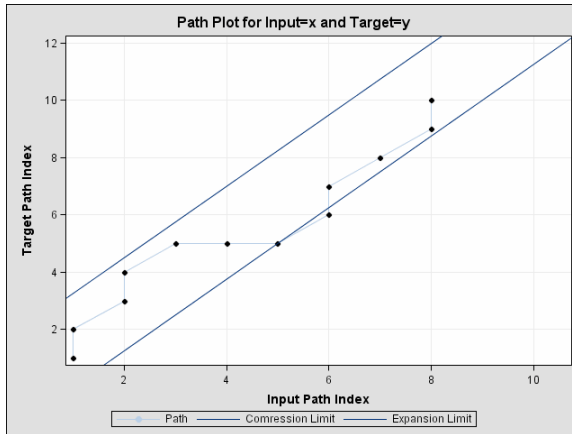


Figure 9. Path Plot

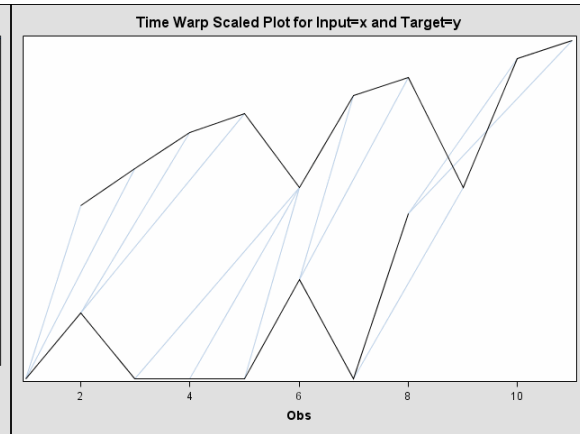


Figure 10. Time Warp Plot

PATH SEQUENCES

Using the path function, *path sequences* can be formed based on the path index. Let $\tilde{x}_p = x_{i_p}$ represent the *time warped input sequence*, and let $\tilde{y}_p = y_{j_p}$ represent the *time warped target sequence* for $p = 1, \dots, P$, where the symbol ‘ \sim ’ indicates warping. In other words, both the input sequence and the target sequence are *expanded* with respect to the path index. Since these two path sequences are equal in length, they can be directly compared based on the path index, p . The ability to compare the path sequences allows for the computation of path distances, which are described later.

Path Sequences Example

Continuing the preceding simple example, the path length is $P = 12$.

The *path sequence plot* is illustrated in Figure 11. The horizontal axis represents the path index, p . The dashed line represents the time warped input sequence: $\tilde{x}_p = x_{i_p}$ for $p = 1, \dots, P$. The solid line represents the time warped target sequence: $\tilde{y}_p = y_{j_p}$ for $p = 1, \dots, P$. This plot visualizes the *mapping* between the input and target sequence values with respect to the *path index*. Figure 12 represents the same plot with the input and target sequence values scaled and evenly separated for visual convenience.

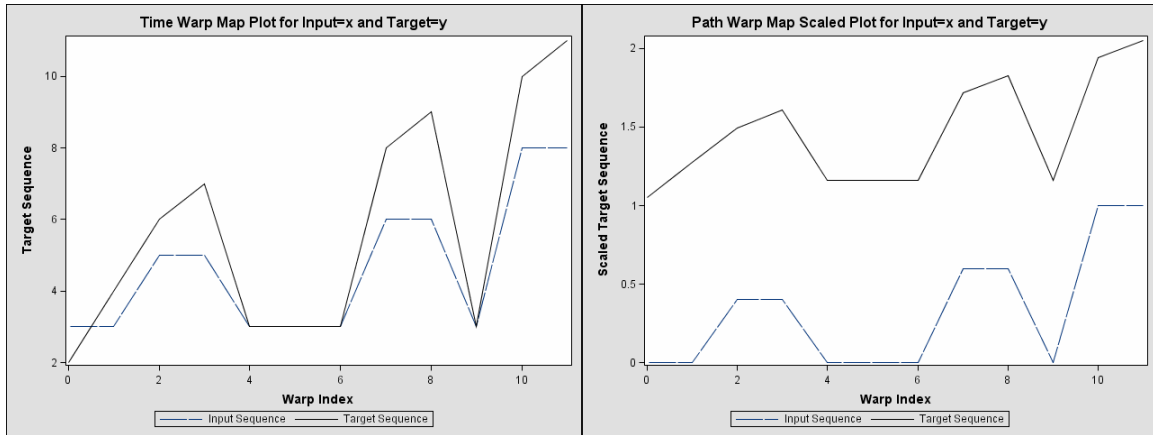


Figure 11. Path Sequence Plot

Figure 12. Path Scaled Sequence Plot

The plots in Figure 11 and 12 are useful for comparing the path sequences or input and target sequence *after* time warping. Both path sequences have two phases of upward movements which are followed by downward movements and a final phase of upward movement. Notice that the two time sequence plots are somewhat similar, but unlike the plots in Figures 1 and 2 they now have the same length and (warped) timing.

PATH STATISTICS

Given a path function, $(i_p, j_p) = Path(p)$, *path statistics* can be computed that describe the direct, compression, and expansion maps along the path. The path statistics describe the amount of compression and expansion of the target sequence relative to the input sequence. The following examples illustrate some of the many possible path statistics.

Direct map statistics are computed from the path associated with direct mappings, as follows:

Direct number	$P^D = \left\ \{Path(p+1) - Path(p) = (1,1)\}_{p=1}^{P-1} \right\ $
Direct path percent	P^D / P
Direct input percent	P^D / N_x
Direct target percent	P^D / N_y

Direct minimum

$$P_{\min}^D = \min_m \{m : Path(p) - (0,0) = \dots = Path(p+m) - (m,m), \quad 1 \leq p \leq P\}$$

or the minimum number of contiguous diagonal movements

Direct maximum

$$P_{\max}^D = \max_m \{m : Path(p) - (0,0) = \dots = Path(p+m) - (m,m), \quad 1 \leq p \leq P\}$$

or the maximum number of contiguous diagonal movements

Compression path statistics are computed from the path associated with compression maps, as follows:

Compression number $P^C = \left\| \{Path(p+1) - Path(p) = (0,1)\}_{p=1}^{P-1} \right\|$

Compression path percent P^C / P

Compression input percent P^C / N_x

Compression target percent P^C / N_y

Compression minimum

$$P_{\min}^C = \min_m \{m : Path(p) - (0,0) = \dots = Path(p+m) - (0,m), \quad 1 \leq p \leq P\}$$

or the minimum number of contiguous vertical movements

Compression maximum

$$P_{\max}^C = \max_m \{m : Path(p) - (0,0) = \dots = Path(p+m) - (0,m), \quad 1 \leq p \leq P\}$$

or the maximum number of contiguous vertical movements

Expansion path statistics are computed from the path associated with expansion maps, as follows:

Expansion number $P^E = \left\| \{Path(p+1) - Path(p) = (1,0)\}_{p=1}^{P-1} \right\|$

Expansion path percent P^E / P

Expansion input percent P^E / N_x

Expansion target percent P^E / N_y

Expansion minimum

$$P_{\min}^E = \min_m \{m : Path(p) - (0,0) = \dots = Path(p+m) - (m,0), \quad 1 \leq p \leq P\}$$

or the minimum number of contiguous horizontal movements

Expansion maximum

$$P_{\max}^E = \max_m \{m : Path(p) - (0,0) = \dots = Path(p+m) - (m,0), \quad 1 \leq p \leq P\}$$

or the maximum number of contiguous horizontal movements

From the preceding path statistics, you can evaluate the amount of warping required to traverse a given path. Even though the similarity measure associated with two sequences might be small, excessive amounts of (contiguous) warping might indicate that the two sequences are dissimilar. Additionally, these path statistics can help detect timing anomalies in an input sequence when compared to a target sequence.

A more complete description of similarity measures is given later. Suffice it to say that the similarity measure might not be the only consideration.

Path Statistics Example

Continuing the preceding simple example, the path length is $P = 12$. The path statistics are illustrated in Table 1.

Path Statistics								
Path	Number	Path Percent	Input Percent	Target Percent	Maximum	Path Maximum Percent	Input Maximum Percent	Target Maximum Percent
Missing Map	0	0.000%	0.000%	0.000%	0	0.000%	0.000%	0.000%
Direct Maps	6	50.00%	75.00%	60.00%	2	16.67%	25.00%	20.00%
Compression	4	33.33%	50.00%	40.00%	1	8.333%	12.50%	10.00%
Expansion	2	16.67%	25.00%	20.00%	2	16.67%	25.00%	20.00%
Warps	6	50.00%	75.00%	60.00%	2	16.67%	25.00%	20.00%

Table 1. Path Statistics

Since this simple example deals with short sequences, these path statistics are not very informative. But these path statistics do show that direct maps are larger than expansion and compression maps.

PATH DISTANCES

Given a path function, $(i_p, j_p) = Path(p)$, and a distance metric, $d_{i,j} = d(x_i, y_j)$, path distances can be computed between the values of the input sequences and target sequence along a given path. Let $\tilde{x}_p = x_{i_p}$ and $\tilde{y}_p = y_{j_p}$ be the time warped input sequence and target sequence values, respectively, along the given path for a path index, p . Let path distances along the path be represented by $d_p = d_{i_p, j_p} = d(x_{i_p}, y_{j_p}) = d(\tilde{x}_p, \tilde{y}_p)$. Dividing a path distance by its corresponding target sequence value results in the *path relative*

distance, $r_p = d_p / \tilde{y}_p$. Let $\vec{d} = \{d_p\}_{p=1}^P$ and $\vec{r} = \{r_p\}_{p=1}^P$ represent the path distance and relative distance vectors, respectively.

From the preceding path distances, you can evaluate the amount of (relative) deviation encountered while traversing a given path. Even though the similarity measure associated with two sequences might be small, excessive amounts of deviation along the path could indicate that the two sequences are dissimilar or could indicate path outliers or *distance anomalies*.

Path Distances Example

Continuing the preceding simple example, the path length is $P = 12$.

The *path distance plot* is illustrated in Figure 13, which plots $\vec{d} = \{d_p\}_{p=1}^P$. The *path relative distance plot* is illustrated in Figure 14, which plots $\vec{r} = \{r_p\}_{p=1}^P$. The horizontal axis represents the path index, p . The vertical needles represent the (relative) distances. The horizontal reference lines indicate one and two standard deviations.

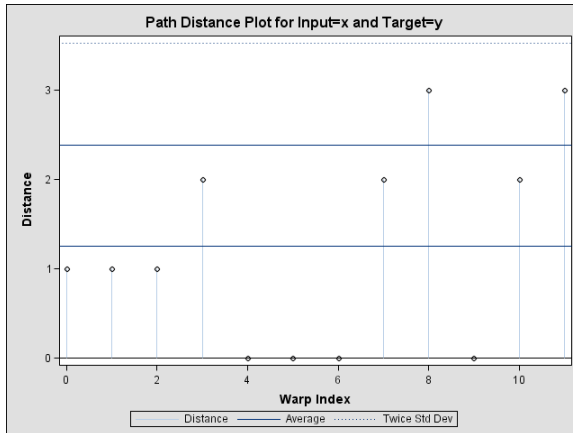


Figure 13. Path Distance Plot

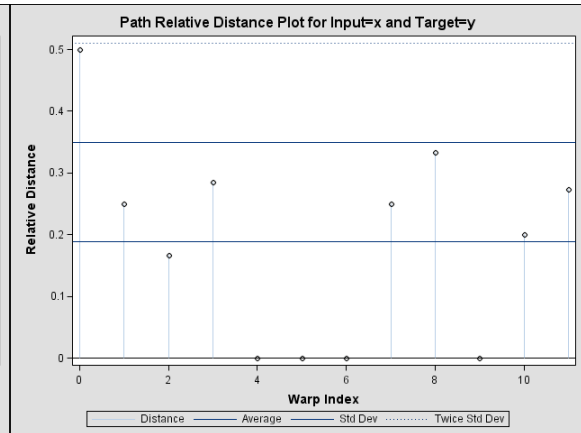


Figure 14. Path Relative Distance Plot

The *path distance histogram* is illustrated in Figure 15. The *path relative distance histogram* is illustrated in Figure 16. The bars represent the histogram, and the solid line represents a normal distribution with the same mean and variance.

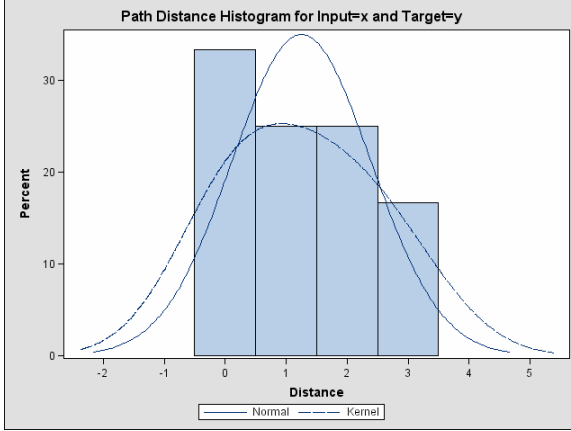


Figure 15. Path Distance Histogram

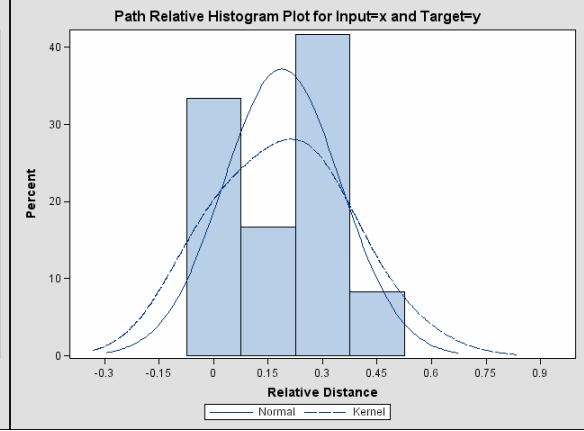


Figure 16. Path Relative Distance Histogram

Although the distribution plots are uninformative in this simple example, distribution analysis of the distances is useful for finding path distances (and sequence values) that are much larger than usual (path outliers).

COST STATISTICS

Given a path distance vector, $\vec{d} = \{d_p\}_{p=1}^P$, and/or a path relative distance vector,

$\vec{r} = \{r_p\}_{p=1}^P$, *cost statistics* can be computed that describe the cost of traversing a given valid path. The following examples illustrate some of the many possible cost statistics:

Number	P	P
Total	$\sum_{p=1}^P d_p$	$\sum_{p=1}^P r_p$
Average	$\frac{1}{P} \sum_{p=1}^P d_p$	$\frac{1}{P} \sum_{p=1}^P r_p$
Standard deviation	$std(\{d_p\}_{p=1}^P)$	$std(\{r_p\}_{p=1}^P)$
Minimum	$\min(\{d_p\}_{p=1}^P)$	$\min(\{r_p\}_{p=1}^P)$
Maximum	$\max(\{d_p\}_{p=1}^P)$	$\max(\{r_p\}_{p=1}^P)$

The following cost statistics provide a *penalty* for long paths through the distance matrix:

Input length average	$\frac{1}{N_x} \sum_{p=1}^P d_p$	$\frac{1}{N_x} \sum_{p=1}^P r_p$
Target length average	$\frac{1}{N_y} \sum_{p=1}^P d_p$	$\frac{1}{N_y} \sum_{p=1}^P r_p$

$$\begin{array}{l} \text{Minimum path average} \quad \frac{1}{\max(N_x, N_y)} \sum_{p=1}^P d_p \quad \frac{1}{\max(N_x, N_y)} \sum_{p=1}^P r_p \\ \text{Maximum path average} \quad \frac{1}{(N_x + N_y - 1)} \sum_{p=1}^P d_p \quad \frac{1}{(N_x + N_y - 1)} \sum_{p=1}^P r_p \end{array}$$

If you divide the sum of the distances by something other than the path length, the cost statistic associated with different pairs of sequences can be penalized for large path lengths.

From the preceding cost statistics, you can evaluate the costs required to traverse a given path through the distance matrix. Even though the similarity measure associated with two sequences might be small, excessive cost statistics might indicate that the two sequences are dissimilar.

Let $Cost(X, Y)$ represent a single cost statistic associated with a warping path between the target sequence and the input sequence. When this cost statistic is related to a specific path, $\pi \in \Pi$, it is denoted as follows: $Cost_{\pi}(X, Y)$.

Cost Statistics Example

Continuing the preceding simple example, the path length is $P = 12$. The cost statistics are illustrated in Table 2.

Cost Statistics								
Cost	Number	Total	Average	Standard Deviation	Min	Max	Minimum Path Mean	Maximum Path Mean
Absolute	12	15.00000	1.250000	1.138180	0	3.000000	1.875000	0.8823529
Relative	12	2.25844	0.188203	0.160922	0	0.500000	0.282305	0.1328495

Table 2. Cost Statistics

Since this simple example deals with short sequences, these cost statistics are not very informative. But these cost statistics do show that the cost of traversing the path varies a lot when the mean and standard deviation are considered.

OPTIMAL PATHS

When judging the similarity between the input and target sequences, there are many possible paths from which a measure can be computed. Since one of the cost statistics can provide a single statistic, the path that minimizes a specified cost statistic can provide such a measure. Let Π represent the set of all possible paths through the distance matrix, with $\pi \in \Pi$ representing a single path and $Cost_{\pi}(X, Y)$ representing the cost of traversing the path:

$$Sim(X, Y) = \min_{\pi \in \Pi} (Cost_{\pi}(X, Y))$$

There are many algorithms that can be used to perform the preceding minimization efficiently. One such algorithm is dynamic time warping, which uses an elegantly simple dynamic programming algorithm that has an order of complexity of $O(N_x N_y)$. Other algorithms such as least common subsequence might be useful.

PROBLEMS WITH DISTANCE AND/OR PROXIMITY MEASURES AS SIMILARITY MEASURES

Using a given distance metric, one possible approach is to compute similarity measures that strictly observe the sequence indices—that is, $Sim(X, Y) = \sum_k d(x_k, y_k)$.

The problems with this similarity measure are evident when you consider that the sequences might differ in length and timing. For example, if the input sequence length is smaller than the target sequence length, should the comparison be anchored to the start, end, or intermediate sequence indices of the target sequence? Suppose that the input and target sequences are similar but shifted with respect to time. Strictly observing the timing might result in a poor comparison.

SIMILARITY VECTORS

Similarity measures can be used to compare a single target sequence to K input sequences to form a *similarity vector*. This situation usually arises in *time series search/retrieval*. For example, given a single target sequence, you can search for similar input sequences by finding the “most similar” or “closest” input sequences. For example, given K input sequences, a $(K \times 1)$ similarity vector can be constructed whose i th element contains the similarity measure between the single target sequence and the i th input sequence. This analysis can be repeated to search for a large number of target sequences.

More formally, given a single target sequence, Y , and K input sequences, $X^{(i)}$, where $i = 1, \dots, K$, a similarity vector can be constructed as follows: $\vec{m} = \{m_i\}_{i=1}^K$, where $m_i = Sim(X^{(i)}, Y)$, $m_i = SlideSim(X^{(i)}, Y)$, or $m_i = SeasonSlideSim(X^{(i)}, Y)$. Searches and retrievals can be performed based on ranking the elements of the similarity vector, \vec{m} .

Similarity measures can be used to compare a single input sequence to K representative target sequences to form a *similarity vector*. This situation usually arises in *time series classification*. For example, given a single input sequence, you can classify the input sequence by finding the “most similar” or “closest” target sequences. For example, given K target sequences, a $(1 \times K)$ similarity vector can be constructed whose j th element contains the similarity measure between the single input sequence and the j th target sequence. This analysis can be repeated to classify large numbers of input sequences.

More formally, given a single input sequence, X , and K target sequences, $Y^{(j)}$, where $j = 1, \dots, K$, a similarity vector can be constructed as follows: $\bar{m} = \{m_j\}_{j=1}^K$, where $m_j = Sim(X, Y^{(j)})$, $m_j = SlideSim(X, Y^{(j)})$, or $m_j = SeasonSlideSim(X, Y^{(j)})$.

Classifications can be computed based on ranking the elements of the similarity vector, \bar{m} .

SIMILARITY MATRICES

Similarity measures can be used to compare several time sequences to form a *similarity matrix*. This situation usually arises in *time series clustering*. For example, given K time sequences, a $(K \times K)$ symmetric matrix can be constructed whose ij th element contains the similarity measure between the i th and j th sequence.

More formally, given K time sequences, $Q^{(j)}$, where $j = 1, \dots, K$, a similarity matrix can be constructed as follows: $M = \{m_{i,j}\}_{i,j=1}^K$, where $m_{i,j} = Sim(Q^{(i)}, Q^{(j)})$, $m_{i,j} = SlideSim(Q^{(i)}, Q^{(j)})$, or $m_{i,j} = SeasonSlideSim(Q^{(i)}, Q^{(j)})$. Clusters can be computed based on the rows and columns of the similarity matrix, M .

SAS IMPLEMENTATION

The similarity analysis described above can be performed with SAS software. This section describes how the SIMILARITY procedure analyzes time-stamped and time series data. The SIMILARITY procedure can be extended by using user-defined subroutines.

For details about the PROC SIMILARITY syntax, see the *SAS/ETS User's Guide*.

PROC SIMILARITY EXAMPLE

The simple example illustrated above can be implemented using PROC SIMILARITY as follows:

```
data test;
input i y x;
datalines;
1 2 3
2 4 5
3 6 3
4 7 3
5 3 3
6 8 6
7 9 3
8 3 8
9 10 .
10 11 .
;
run;

ods listing close;
ods graphics on;
ods html;

* no path limits;
proc similarity data=test out=_null_ outpath=path
print=all plot=all;
input x;
target y / measure=absdev slide=none
;
run;

* with path limits
proc similarity data=test out=_null_ outpath=pathlimit
print=all plot=all;
input x;
target y / measure=absdev slide=none
compress=(localabs=2)
expand=(localabs=2)
;
run;

ods listing;
ods graphics off;
ods html close;
```

CONCLUSION

Mining transactional and time series data can help business leaders make better decisions by enabling them to listen to their suppliers or customers via their transactions collected over time. A single set of transactions or a single time series can contain large amounts of information. A business can have many suppliers and/or customers. The volume of information associated with each of the suppliers and customers makes data mining tasks difficult. The approach outlined in this paper relies on similarity analysis that uses similarity measures combined with automatic time series analysis and decomposition. This similarity analysis can help make traditional data mining tasks such as searching, classification, clustering, and decision trees more tractable.

RECOMMENDED READING

An introductory discussion of time series analysis, decomposition, and forecasting can be found in Makridakis, Wheelwright, and Hyndman (1997), Brockwell and Davis (1996), and Chatfield (2000). A more detailed discussion of time series analysis and forecasting can be found in Box, Jenkins, and Reinsel (1994), Hamilton (1994), Fuller (1995), and Harvey (1994). An introductory discussion of data mining can be found in Barry and Linoff (1997). A more detailed discussion of data mining can be found in Han and Kamber (2001). A detailed discussion of data preparation for data mining can be found in Pyle (1999). A detailed discussion of time warping can be found in Sankoff and Kruskal (2001).

REFERENCES

- Barry, M. J. and Linoff, G. S. (1997), *Data Mining Techniques: For Marketing, Sales, and Customer Support*, New York: John Wiley & Sons.
- Box, G. E. P, Jenkins, G. M., and Reinsel, G. C. (1994), *Time Series Analysis: Forecasting and Control*, Englewood Cliffs, NJ: Prentice Hall.
- Brockwell, P. J. and Davis, R. A. (1996), *Introduction to Time Series and Forecasting*, New York: Springer-Verlag.
- Chatfield, C. (2000), *Time Series Models*, Boca Raton, FL: Chapman & Hall/CRC.
- Fuller, W. A. (1995), *Introduction to Statistical Time Series*, New York: John Wiley & Sons.
- Hamilton, J. D. (1994), *Time Series Analysis*, Princeton, NJ: Princeton University Press.
- Han, J. and Kamber, M. (2001), *Data Mining: Concepts and Techniques*, San Francisco: Morgan Kaufmann Publishers.
- Harvey, A. C. (1994), *Time Series Models*, Cambridge, MA: MIT Press.

Leonard, M. J. (2002), "Large Scale Automatic Forecasting: Millions of Forecasts," *International Symposium of Forecasting*.

Leonard, M. J. (2004), "Large Scale Automatic Forecasting with Inputs and Calendar Events," *International Symposium of Forecasting*.

Leonard, M. J. and Wolfe, B. L. (2005), "Mining Transactional and Time Series Data," *Proceedings of the Thirtieth Annual SAS Users Group International Conference*, Cary, NC: SAS Institute Inc.

Makridakis, S. G., Wheelwright, S. C., and Hyndman, R. J. (1997), *Forecasting: Methods and Applications*, New York: John Wiley & Sons.

Pyle, D. (1999), *Data Preparation for Data Mining*, San Francisco: Morgan Kaufman Publishers.

Sankoff, D. and Kruskal, J. B. (2001), *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, Stanford, CA: CSLI Publications.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author(s):

Michael Leonard
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
Work Phone: (919) 531-6967
Email: Michael.Leonard@sas.com

Jennifer Sloan
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
Work Phone: (919) 531-1572
Email: Jennifer.Sloan@sas.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.