2003－DBS－131　(64)
2003／7／17

# Adaptive Time Warping

大桃 論[†]　陳 漢雄[††]
古瀬 一隆[††]　大保 信夫[††]

大規模な時系列データベースに対する高速な類似検索が重要な課題となっている．これまでにユークリッド距離に基づく類似検索技法が多く提案されているが，最近の研究によって，タイムワーピングによって得られた距離が時系列データの類似検索において有効であることが実証されてきた．しかし，タイムワーピングの計算時間は非常に長く，しかも距離公理の三角不等式が成り立たないため，伝統的な索引技法の適用は困難である．この問題を解決するために，時系列を固定長に区切って圧縮する手法が提案されているが，本論文では，時系列を可変長に区切って圧縮する手法を提案する．そして，本論文で提案する手法が有効であることを実験によって実証する．

# Adaptive Time Warping

Satoshi OOMOMO ,[†] Hanxiong CHEN ,[††]
Kazutaka FURUSE [††] and Nobuo OHBO[††]

Recently, time series produced and treated in many fields, and fast similarity search in large time series databases becomes important. For this subject, similarity search techniques based on Euclidean distance are straighforward. However, recent studys have shown that time warping distance is more robust distance in similarity search for time series. Difficulty is that time warping distance is computationally expensive. Traditional indexing techniques is powerless because it violates the triangle inequality. To overcome this problem, compression is considered to be effectvie and some techniques which divide time series into fix length for compressing have been proposed. In this paper we propose an adaptive compressing technique. Prefer to fix length division, our approach divides a time series according to its characters hence obtains higher effect while keeping lower information lost in compression. The effectiveness has been confirmed in out experimental results.

## 1. 序　論

近年，時系列データは科学，医療，経済，工学などの様々な分野で扱われるようになり，それぞれの分野でデータマイニングが行われるようになっている．そして，時系列データに対してデータマイニングを行うには，時系列データの類似度を計算することが必要となる．

現在，類似度として最もよく使用されているのがユークリッド距離である．しかし，ユークリッド距離には時間軸におけるほんの小さな歪みにも影響を受けやすいという欠点がある．その実例を図1に示す．図1のAの2つの時系列は，直観的には類似していると感じられる．しかし，時間軸に歪みがあるので，ユークリッド距離は大きくなってしまい，そのために類似度は低くなってしまう．

このような時間軸における歪みに影響を受けにくい距離を得るために，タイムワーピング（Time Warping: TW）という技法がある．図1のAと同じ時系列に対してTWを適用したものを図1のBに示す．TWでは，一方の時系列の1つの点と他方の時系列の連続する複数の点を対応させることによって，時間軸に伸縮性を持たせて歪みの影響を抑え

† 筑波大学大学院システム情報工学研究科
Graduate School of Systems and Information Engineering, University of Tsukuba
†† 筑波大学電子・情報工学系
Institute of Information Sciences and Electronics, University of Tsukuba

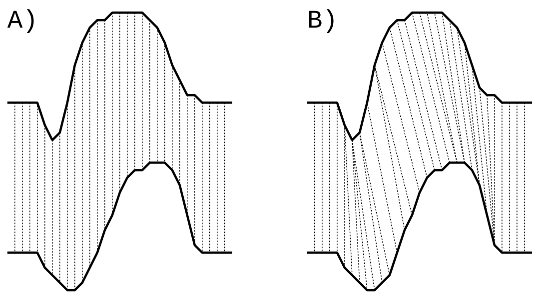


**図 1**　ユークリッド距離とタイムワーピング距離
Fig. 1　Euclidean distance and Time Warping distance

ている．

これまでの研究によって TW の有効性は実証されているが，計算時間が非常に長いという欠点があることも知られている．長さが $n$ の 2 つの時系列が与えられたときに，ユークリッド距離の計算時間のオーダーは $O(n)$ であるが，TW の計算時間のオーダーは $O(n^2)$ となってしまう．よって，対象となるデータベースが大きい場合には，TW を適用することは現実的に不可能となってしまう．そこで，計算時間を短くするための様々な手法が提案されている．その中の 1 つに，時系列を圧縮する手法がある．これは，時系列にそのまま TW を適用して距離を計算するかわりに，圧縮した時系列に TW を適用して距離を計算して，その距離を類似度として使用するというものである．そして，圧縮する手法の 1 つに，時系列を固定長に区切って圧縮する Piecewise Dynamic Time Warping（PDTW）という手法がある．

PDTW を行うことによって，正確性の低下を抑えつつ高速化できることが実証されているが，時系列の変動の度合を考慮に入れずに全ての時系列を固定長で区切って圧縮するよりも，時系列の変動の度合に応じて，変動の多い部分は短く，変動の少ない部分は長く区切って圧縮した方が，圧縮によって生じる誤差を抑えつつ高速化できると考えられる．

そこで，本論文では，時系列を変動の度合に応じて可変長で区切って圧縮して，その圧縮された時系列に対して TW を適用する Adaptive Time Warping（ATW）の手法を提案する．時系列を適切に区切って圧縮することによって，冗長なデータを持つことなく，しかも元の時系列の特徴を残した時系列を得ることができる．冗長なデータを持たないので TW を適用したときの計算時間が短くなり，元の時系列の特徴を残しているので正確性の低下が抑えられる．

本論文のこれからの構成は以下のようになっている．第 2 章では，TW のアルゴリズムと PDTW の概要を述べる．第 3 章では，本論文で提案する ATW について述べる．第 4 章では，ユークリッド距離，TW，PDTW，ATW の性能を比較する実験の結果を述べる．最後に第 5 章で，本研究から得られた結論とこれからの課題を述べる．

## 2. タイムワーピングと固定長圧縮

### 2.1 タイムワーピング

この章では，タイムワーピング（TW）のアルゴリズムを簡単に述べる[1]．

長さが $n, m$ の 2 つの時系列 $X = x_1, x_2, \cdots, x_i, \cdots, x_n$，$Y = y_1, y_2, \cdots, y_j, \cdots, y_m$ に対して TW を適用して距離を求めることを考える．

まずは，2 つの点 $x_i, y_j$ 間の距離 $d(x_i, y_j)$ を $(i, j)$ 要素の値とする $n \times m$ 行列を作成する．ここで，$d(x_i, y_j)$ は以下のように定義する．

$$d(x_i, y_j) = (x_i - y_j)^2 \tag{1}$$

次に，ワーピングパス $W$ を求める．ワーピングパスは，いくつかの条件を満たす行列の要素の順列で表現される．つまり，$W$ の $k$ 番目の要素を $w_k = (i, j)_k$ として，ワーピングパスは以下のように表現される．

$$W = w_1, w_2, \cdots, w_k, \cdots, w_K \tag{2}$$

ここで，ワーピングパスの必要十分条件を示す．

**境界条件**　$w_1 = (1, 1), w_K = (n, m)$ とする．これは，ワーピングパスが行列の 1 つの角から始まり，その対角の角で終わることを意味する．

**連続性**　$w_k = (a, b), w_{k-1} = (a', b')$ とすると，$a - a' \leq 1$ かつ $b - b' \leq 1$ となる．これは，ワーピングパスが隣接している行列の要素だけをたどっていくことを意味する．

**単調性**　$w_k = (a, b), w_{k-1} = (a', b')$ とすると，$a - a' \geq 0$ かつ $b - b' \geq 0$ となる．これは，ワーピングパスが反対方向には進まないことを意味する．

ワーピングパスの例を図 2 に示す．

上記の 3 つの条件を満たすワーピングパス

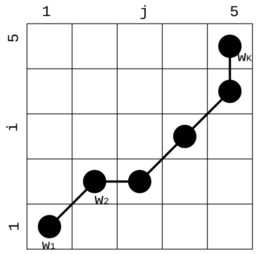


**図 2**　ワーピングパスの例
Fig. 2　An example of warping path

は膨大に存在するが，その中からワーピングコストが最小のパスを見つけて，TW 距離を以下の式から求める．

$$TW(X,Y)=\frac{min\left\{\sqrt{\sum_{k=1}^{K} w_k}\right\}}{K} \quad (3)$$

ワーピングコストが最小のパスを求めるためには，以下の再帰式を利用する．

$$\gamma(i,j)=d(x_i,y_j) + min\left\{\begin{array}{l}\gamma(i-1,j-1)\\ \gamma(i-1,j)\\ \gamma(i,j-1)\end{array}\right\} \quad (4)$$

この再帰式を利用すると，式 (3) は以下のように表される．

$$TW(X,Y)=\frac{\sqrt{\gamma(n,m)}}{K} \quad (5)$$

TW 距離は，動的プログラミングを使用することで，計算時間 $O(nm)$ で求めることができる．

### 2.2 固定長圧縮を利用したタイムワーピング

この章では，時系列を固定長に区切って圧縮した後にタイムワーピングを適用する手法（PDTW）の概要を述べる[2]．

まずは，長さが $n$ の時系列 $X=x_1,\cdots,x_n$ を，長さ $N$ に圧縮する．ここでは，説明を簡単にするために $N$ は $n$ の因数であるとするが，実際には $1\leq N\leq n$ を満たせばよい．圧縮後の時系列を，$\bar{X}=\bar{x}_1,\cdots,\bar{x}_N$ とすると，$\bar{X}$ の $i$ 番目の点 $\bar{x}_i$ の値は以下のように定義される．

$$\bar{x}_i=\frac{N}{n}\sum_{j=\frac{n}{N}(i-1)+1}^{\frac{n}{N}i} x_j \quad (6)$$

この式は，時系列の $n$ 個の点を同じ大きさの $N$ 個のフレームに分けて，フレーム内の $n/N$ 個の点の平均値を圧縮後の時系列の点の値とすることを表している．実際に時系列を圧縮した様子を図 3 に示す．

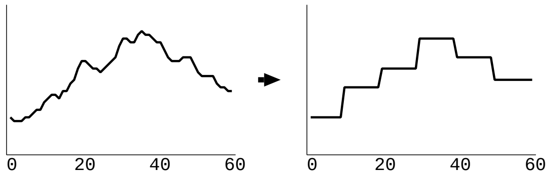


**図 3**　時系列の固定長圧縮
Fig. 3　Fixed lenghth compression of time series

圧縮の程度を指定するパラメータとして，以下のように定義される圧縮率 $c$ を用いる．

$$c=n/N \quad (7)$$

時系列を圧縮した後は，圧縮された時系列に対して第 2.1 章で述べた TW をそのまま適用する．計算時間は，2 つの時系列の長さを $n,m$，圧縮率を $c$ とすると，$O(nm/c^2)$ となる．

## 3. 可変長圧縮を利用したタイムワーピング

第 2.2 章で述べた PDTW は，時系列を固定長に区切って圧縮してから TW を適用しているが，固定長に区切ることが適切ではない場合がある．

図 4 で示すような変動の少ない時系列と変動の多い時系列が，1 つのデータベースの中に存在する場合を考えてみる．変動が少ない時系列にあわせて大きな圧縮率を使用すると，変動が多い時系列は圧縮によって元の時系列の特徴が失われてしまう．反対に，変動が多い時系列にあわせて小さな圧縮率を使用すると，変動が少ない時系列は圧縮したにもかかわらず冗長なデータが多くできてしまう．よって，このデータベースの中の全ての時系列に対して同じ圧縮率を使用することは適切ではない．また，1 つの時系列の中に変動が少ない部分と多い部分が存在する場合もある．そのような時系列に対しては，固定した圧縮率では適切に圧縮することはできない．

このような問題を解決するために，時系列を変動の度合に応じて可変長に区切って圧縮する方法を用いる．時系列の中の変動の少ない部分は，長く区切ることによって高い圧縮率を実現して，冗長なデータを持たないようにする．反対に変動の多い部分は，短く区切ることによって低い圧縮率を実現して，元の

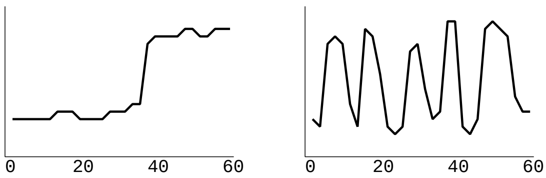


**図 4** 変動の度合が異なる時系列
Fig. 4 Time series with different frequency

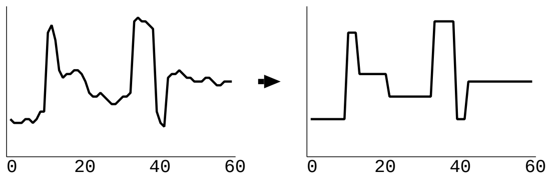


**図 5** 時系列の可変長圧縮
Fig. 5 Variable lenghth compression of time series

時系列の特徴が失われないようにする．実際にどのような方法で時系列を可変長に区切って圧縮するかについては，第 3.1 章で詳しく述べる．

次に，可変長圧縮された時系列に対して TW を適用する方法を考えなければならない．PDTW は固定長圧縮なので，圧縮された時系列に対して TW をそのまま適用することができるが，可変長圧縮の場合には，区切られた長さを考慮に入れて TW を適用しなければならない．実際にどのような方法で TW を適用するかについては，第 3.2 章で詳しく述べる．

本研究では，この 2 つの手法をあわせて ATW とする．

### 3.1 時系列の可変長圧縮

まず初めに，可変長圧縮の流れを示す．

(1) 時系列の区切り箇所を決定して複数の部分時系列に分ける．これからは，この部分時系列のことをフレームと呼ぶ．
(2) 1 つのフレームを 1 つの点に圧縮することによって時系列全体を圧縮する．フレームの値は，そのフレーム内の点の値の平均値とする．

可変長圧縮を行う上で一番重要なことは，適切なフレーム分けを行うことである．適切なフレーム分けを行うとは，圧縮によって生じる誤差を一定以下に抑えつつ，フレームの数をなるべく少なくするということである．フレームの数が圧縮後の時系列の長さになるので，フレームの数が少ないほど，圧縮後の時系列に対して TW を適用するときの計算時間は短くなる．また，ここでいう誤差とは，以下の式から得られる値である．

$$error = \frac{\sum_{i=1}^{n} |f(x_i) - x_i|}{n} \tag{8}$$

$f(x_i)$ は，圧縮したときに $x_i$ が属するフレームの値を表す．適切なフレーム分けを行って時系列を圧縮した例を図 5 に示す．

次に重要なことは，できるだけ短い計算時間でフレーム分けを行うことである．たとえ適切なフレーム分けが行われたとしても，その処理に有する時間が長ければ圧縮する意味がない．

これらのことを踏まえて，計算時間が $O(n)$ で時系列 $X = x_1, \cdots, x_n$ を $\hat{X} = \hat{x}_1, \cdots, \hat{x}_N$ に可変長圧縮を行うアルゴリズムを表 1 に示す．このアルゴリズムに現れる $\tau$ は閾値であり，これについては後で説明する．

**表 1** 可変長圧縮のアルゴリズム
Table 1 Algorithm for variable length compression

1: $s = 1$, $N = 1$
2: find minimum $e(s \le e < n)$ such that
  $max\{x_s, \cdots, x_e\} - min\{x_s, \cdots, x_e\} \le \tau$
  $max\{x_s, \cdots, x_{e+1}\} - min\{x_s, \cdots, x_{e+1}\} > \tau$
  if $e$ dose not exist, go to 5.
3: calculate frame value $\hat{x}_N.v$ and length $\hat{x}_N.l$.
  $\hat{x}_N.v = \frac{\sum_{i=s}^{e} x_i}{e-s+1}$
  $\hat{x}_N.l = e - s + 1$
4: $s = e + 1$, $N = N + 1$, and go to 2.
5: calculate last frame value $\hat{x}_N.v$
  and length $\hat{x}_N.l$.
  $\hat{x}_N.v = \frac{\sum_{i=s}^{n} x_i}{n-s+1}$
  $\hat{x}_N.l = n - s + 1$

このアルゴリズムに従って，$\tau = 4$ として時系列の最初の部分をフレーム分けする様子を図 6 に示す．点 A から点 B までの部分時系列の最大値と最小値の差は 3 であり，$\tau$ 以下の値となっているが，点 B の次の点である点 C まで進むと，最大値と最小値の差は 6 になり，閾値より大きくなる．よって，点 A から点 B までが 1 つのフレームとなり，点 C が次のフレームの開始点になる．これを時系列の最後まで繰り返すと，フレーム分けは終了する．

ここで圧縮によって生じる誤差について考えてみると，式 (8) で表される誤差は以下のようになり，$\tau$ より小さくなることが保証される．

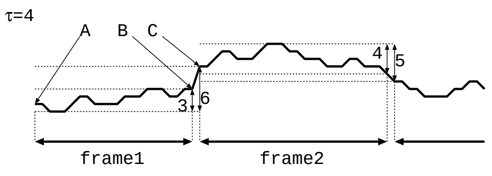


**図 6**　時系列のフレーム分け
Fig. 6　Division of time series into frames

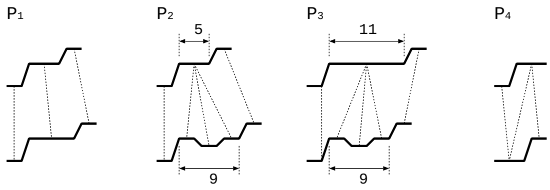


**図 7**　ATW における時系列の対応関係
Fig. 7　Relations of time series in ATW

$$error = \frac{\sum_{i=1}^{n} |f(x_i) - x_i|}{n} < \frac{\sum_{i=1}^{n} \tau}{n} = \tau \qquad (9)$$

次に考えなければならないことは，閾値 $\tau$ をどのように決定するかということである．$\tau$ をユーザが直接指定するのは適切ではない．なぜならば，時系列の取り得る値の範囲をあらかじめ知っていなければ，適切な閾値を指定できないからである．

そこで，圧縮の程度をユーザが直接指定できるパラメータとして，許容振幅率 $\rho$ というものを定義する．この $\rho$ から，閾値 $\tau$ は以下の式で決定される．

$$\tau = \rho \cdot (\ max\{x_1, \cdots, x_n\} - min\{x_1, \cdots, x_n\}) \qquad (10)$$

これによって，時系列の取り得る値の範囲に関係なく，0 から 1 の値で圧縮の程度を指定できるようになる．

### 3.2　可変長圧縮された時系列に対する TW の適用

時系列を可変長圧縮した後は，圧縮された時系列に対して TW を適用して距離を計算することになる．しかし，フレームの長さが一定ではないために，可変長圧縮された時系列に対してそのまま TW を適用することはできない．そこで，フレームの長さを考慮に入れて TW を適用する方法について説明する．

長さが $N, M$ の 2 つの圧縮時系列 $\hat{X} = \hat{x}_1, \hat{x}_2, \cdots, \hat{x}_i, \cdots, \hat{x}_N$，$\hat{Y} = \hat{y}_1, \hat{y}_2, \cdots, \hat{y}_j, \cdots, \hat{y}_M$ に TW を適用して距離を求めることを考える．

まずは，圧縮時系列に対して第 2.1 章で述べた方法をそのまま適用して，ワーピングパスを求める．次に，このワーピングパスから距離を求めるのだが，その方法について述べる前にワーピングパスについて詳しく考えてみる．

第 2.1 章で述べた方法でワーピングパスを

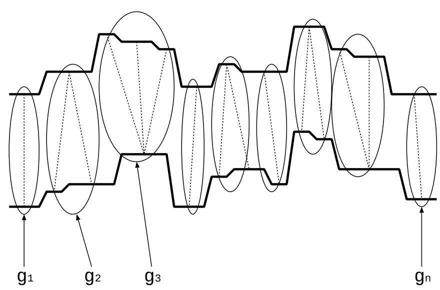


**図 8**　ワーピングパスのグループ分け
Fig. 8　Grouping of warping path

求めると，図 7 の $P_1$ に示すような ‘1’ 対 ‘1’ の関係と，$P_2$ や $P_3$ に示すような ‘1’ 対 ‘多’ の関係は存在するが，$P_4$ に示すような ‘多’ 対 ‘多’ の関係は存在しない．この証明に関しては省略する．

そして，可変長圧縮された時系列におけるワーピングパスでは，‘1’ 対 ‘多’ の関係をさらに 2 つのパターンに分けることができる．図 7 の $P_2$ のような ‘1’ の方のフレームの長さが ‘多’ の方のフレームの長さの和よりも短いパターンと，$P_3$ のような ‘1’ の方が ‘多’ の方よりも長いパターンである．

可変長圧縮された時系列におけるワーピングパスは $P_1, P_2, P_3$ の 3 つのパターンで構成されているので，図 8 に示すようにワーピングパスを $g_1, \cdots, g_n$ にグループ分けして，以下の式に従ってパターン別のグループの値 $g_h.v$ と大きさ $g_h.l$ を求める．

**パターン** $P_1$　グループ $g_h$ における関係が $\{\hat{x}_i\}$ と $\{\hat{y}_j\}$ の ‘1’ 対 ‘1’ であるとすると，

$$g_h.v = d(\hat{x}_i, \hat{y}_j) \cdot max\{\hat{x}_i.l, \hat{y}_j.l\} \qquad (11)$$

$$g_h.l = max\{\hat{x}_i.l, \hat{y}_j.l\} \qquad (12)$$

**パターン** $P_2$　グループ $g_h$ における関係が $\{\hat{x}_i\}$ と $\{\hat{y}_j, \cdots, \hat{y}_k\}$ の ‘1’ 対 ‘多’ であり，フレームの長さが $\hat{x}_i.l \leq \sum_{s=j}^{k} \hat{y}_s.l$ であるとすると，

$$g_h.v = \sum_{s=j}^{k} (d(\hat{x}_i, \hat{y}_s) \cdot \hat{y}_s.l) \qquad (13)$$

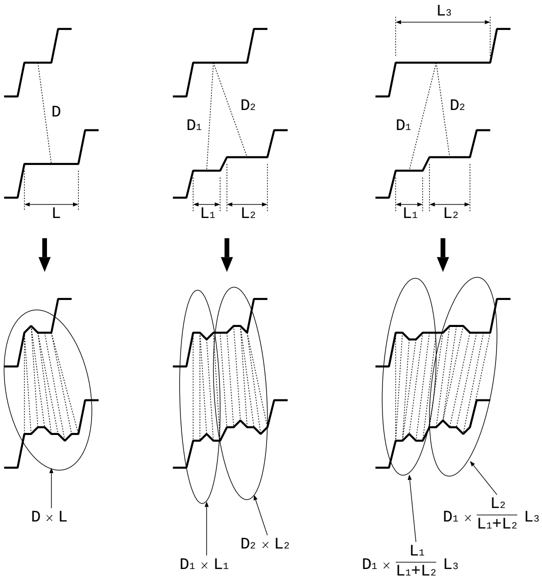


**図 9**　圧縮前の時系列に対する TW の適用
Fig. 9　Application of TW to original time series

$$g_h.l = \sum_{s=j}^{k} \hat{y}_s.l \qquad (14)$$

**パターン** $P_3$　グループ $g_h$ における関係が $\{\hat{x}_i\}$ と $\{\hat{y}_j, \cdots, \hat{y}_k\}$ の‘1’対‘多’であり，フレームの長さが $\hat{x}_i.l > \sum_{s=j}^{k} \hat{y}_s.l$ であるとすると，

$$g_h.v = \frac{\hat{x}_i.l}{\sum_{s=j}^{k} \hat{y}_s.l} \cdot \sum_{s=j}^{k} (d(\hat{x}_i, \hat{y}_s) \cdot \hat{y}_s.l) \qquad (15)$$

$$g_h.l = \hat{x}_i.l \qquad (16)$$

これらの式は，圧縮前の時系列に対して TW を適用すると図 9 のようになるという考えから得られたものである．

パターン毎の計算式から得られた $n$ 個のグループ $g_1, \cdots, g_n$ の値と大きさから，ATW 距離を以下のように定義する．

$$ATW(X, Y) = \frac{\sqrt{\sum_{h=1}^{n} g_h.v}}{\sum_{h=1}^{n} g_h.l} \qquad (17)$$

このようにして得られた ATW 距離と圧縮前の時系列の TW 距離との関係は，以下の式のようになると考えられる．

$$TW(X, Y) \simeq ATW(X, Y) \qquad (18)$$

## 4. 実 験 結 果

この章では，従来からの手法と今回提案した手法の比較実験を行った結果について述べる．比較した手法は，以下の 4 種類である．

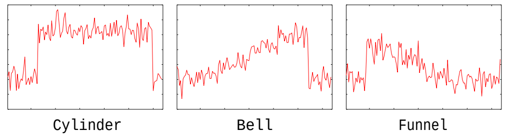


**図 10**　CBF データ
Fig. 10　CBF data

**Euclidean**　ユークリッド距離
**TW**　圧縮を行わずに TW を適用
**PDTW**　固定長で圧縮してから TW を適用
**ATW**　可変長で圧縮してから TW を適用（本論文の提案手法）

時系列の類似検索を行う上で重要なことは，本当に類似している時系列を，正確かつ高速に見つけるということである．そこで，合成データセットと実データセットを使用して分類実験を行い，それぞれの手法の実行時間と正解率を計測した．

合成データセットには，論文[2]において使用されている CBF データセットを用いた．CBF データセットには，Cylinder，Bell，Funnel の 3 種類のクラスが存在して，それぞれのクラスのデータは図 10 のようになる．

実データセットには，Keogh,E.&Folias,T. (2002). The UCR Time Series Data Mining Archive(*http://www.cs.ucr.edu/~eamonn/TSDMA/index.html*) の時系列を使用した．これらの時系列の中から，異なる特徴を持つ 7 種類の時系列をクラスとして選択した．この 7 種類の時系列を平均値が 0，最小値と最大値の差が 1 となるように正規化した後，一定の大きさに分割して，それらを集めてデータセットとした．このデータセットに含まれる 7 種類のクラスのデータを図 11 に示す．

この 2 つのデータセットを用いた分類実験の内容は以下の通りである．まず，それぞれのクラスからランダムに 1 つデータを選んで基準データとする．次に，残りの全てのデータを最も類似度の高い基準データのクラスに分類する．分類が終わったら，全てのデータの中で正しく分類されたデータの割合と要した時間を求める．これを繰り返して平均の結果を求める．

このように実験を行った結果を表 2，3 に示す．PDTW と ATW に関しては，パラメータにいくつかの代表値を選んで実験を行った結果を示した．

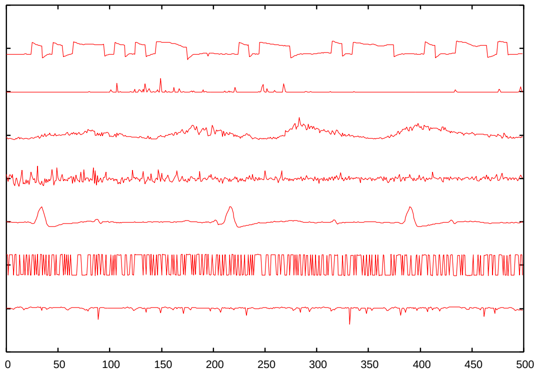


**図 11**　実データ
Fig. 11　Examples of real data

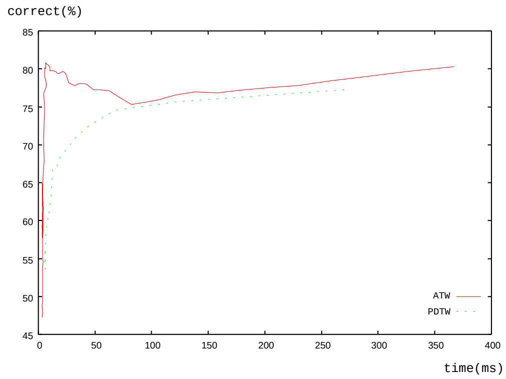


**図 12**　CBF データセットを用いた実験結果
Fig. 12　Result of experiments using CBF dataset

**表 2**　CBF データセットを使用した実験の結果
Table 2　Result of experiments with CBF dataset

| | param | time(ms) | correct(%) |
|---|---|---|---|
| Euclidean | | 3.4 | 64.97 |
| TW | | 1086.3 | 83.30 |
| PDTW | 2 | 270.1 | 77.28 |
| | 4 | 67.8 | 74.56 |
| | 8 | 18.2 | 68.14 |
| ATW | 0.1 | 366.9 | 80.26 |
| | 0.2 | 106.3 | 75.95 |
| | 0.3 | 27.1 | 78.17 |
| | 0.4 | 8.5 | 80.54 |
| | 0.5 | 4.6 | 76.78 |

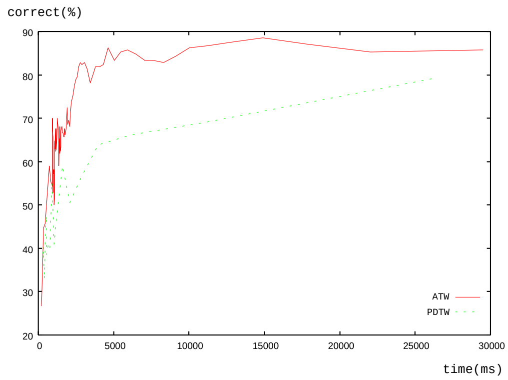


**図 13**　実データセットを用いた実験結果
Fig. 13　Result of experiments using real dataset

**表 3**　実データセットを使用した実験の結果
Table 3　Result of experiments with real dataset

| | param | time(ms) | correct(%) |
|---|---|---|---|
| Euclidean | | 70 | 26.19 |
| TW | | 100020 | 80.95 |
| PDTW | 2 | 26090 | 79.05 |
| | 4 | 6140 | 66.19 |
| | 8 | 1600 | 59.05 |
| ATW | 0.1 | 8320 | 82.86 |
| | 0.2 | 3810 | 81.90 |
| | 0.4 | 1740 | 67.62 |
| | 0.6 | 1200 | 62.86 |

この結果から，ユークリッド距離は計算時間が短く，TW は正解率が高いということがわかる．PDTW と ATW に関しては，それら 2 つの中間的な結果になっていることはわかるが，パラメータを変えることによって実行時間と正解率が大きく変動するので，表 2, 3 からはどちらが優れているかを判断しづらい．そこで，横軸を実行時間，縦軸を正解率としたグラフを図 12，13 に示す．このグラフは，パラメータを変えて得られた結果を線で結んだものである．

このグラフで線が左上にある方が，性能が優れていると判断することができる．よって，今回提案した手法の ATW が，PDTW よりも優れた性能を持っているということができる．

## 5. 結　　論

本研究では，時系列の類似度を定義する上でユークリッド距離よりも優れているタイムワーピング（TW）に対して，可変長圧縮の手法を用いることによって，欠点であった計算の遅さを改善することを試みた．実験によって，今回提案した手法（ATW）は，すでに提案されている固定長圧縮の手法（PDTW）よりも優れた性能を持っていることが実証された．

今後の課題としては，パラメータと実行時間と正解率の関係について考察して，適切なパラメータの自動設定を行う手法の提案ができれば良いと考えている．この他にも，今回提案した手法を応用して，時系列の類似検索においてフィルタリングを行う手法についても考案している．

## 参 考 文 献


1) Joseph B. Kruskall, and Mark Liberman, “The Symmetric Time-Warping Problem: From Continuous to Discrete,” Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison, pp.125–161, Addison-Wesley, 1983.
2) Keogh,E. and Pazzani,M. “Scaling up Dynamic Time Warping for Datamining Applications,” Proc. of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp.285-289, Boston, MA, USA, August.2000