



時系列の類似性検索における上界関数による効率化

瀧, 敬士
中村, 哲也
上原, 邦昭

(Citation)

電子情報通信学会論文誌. D, 情報・システム, J91-D(12):2926-2938

(Issue Date)

2008-12-01

(Resource Type)

journal article

(Version)

Version of Record

(Rights)

Copyright (c) 2008 IEICE

(URL)

<https://hdl.handle.net/20.500.14094/90000828>



時系列の類似性検索における上界関数による効率化

瀧 敬士^{†a)} 中村 哲也[†] 上原 邦昭^{†b)}

Efficient Similarity Retrieval with Upper Bound for Time Series

Keishi TAKI^{†a)}, Tetsuya NAKAMURA[†], and Kuniaki UEHARA^{†b)}

あらまし 近年、時系列の分類に関する研究が盛んに行われている。その一つとして多次元時系列の類似性検索に有効な手法 Angular Metrics for Shape Similarity (AMSS) がある。AMSS は時系列をベクトル系列として扱い、個々のベクトルの角度を比較して類似度を計測し、動的計画法を用いてベクトル系列間全体の類似度を計算する手法である。また、角度の比較にはコサイン類似度を用いているため、ノイズのような部分は類似度に影響を与えないようにできるという特徴がある。しかしながら、計算コストが高いという問題点を抱えている。本論文では、AMSS を用いた検索のフィルタリングとして上界関数を導入して、計算コストを削減する手法を提案する。ここで上界関数とは、本来の関数より必ず大きな値となる近似値を求める関数である。具体的には、ベクトル系列を合成ベクトル系列に変換して、近似値を求めている。上界関数を用いるため、類似しない時系列を false dismissal なしに検索対象から取り除くことができるという特徴がある。最終的に、上界関数の正当性を示すために、数学的帰納法による証明について述べる。

キーワード データマイニング、時系列、類似度、次元圧縮

1. ま え が き

時系列データマイニングの分野では、分類やクラスタリング、インデクシングのために、時系列間の類似度を計測する手法が提案されている [3], [8]。類似度計測手法として、Dynamic Time Warping (DTW) が知られている [2]。DTW は時系列の長さを動的に整合し、時系列間の類似度を求めている。しかし、時系列間を完全にマッチングする性質から、ノイズに弱いという問題がある。このため、最長共通部分列 (LCSS) に基づいて類似度を計測する手法が提案されている [3]。しかし、LCSS の類似度は、時系列全体に対する類似部分時系列の割合で表されるのみで、どのくらい部分時系列が似ているかという情報はもっていない。また、部分時系列の類似性を判断するためのしきい値は一意に決定できない。更に、DTW と LCSS は、時系列の座標値を用いて類似性を計測しているため、類似する時系列間でも、適切に類似度を求められないという問

題がある。例えば、同じ動きを行う軌跡でも、動く方向が異なっていたり、観測する初期位置が異なる場合には、類似度を正確に測定できない。

上記の問題を解決するため、Angular Metrics for Shape Similarity (AMSS) という手法が提案されている [1]。AMSS は時系列をベクトル系列^(注1)として扱い、ベクトル間の角度のみを比較して類似度を計算する手法である。また、データ整合の問題を解決するために、動的計画法を用いて各ベクトル間の類似度を足し合わせて、ベクトル系列間の類似度を計算している。更に、角度の比較にはコサイン類似度を用いており、ベクトル間の角度が $\pi/2$ 以上の場合には、類似度を 0 としている。このため、ノイズと通常のデータ間のようにベクトル間の角度が大きく異なった場合、類似度は 0 となる。したがって、部分時系列の方向が極度に異なる際には、仮に部分時系列が動的計画法によりマッチした場合でも、類似度の総和は変化せず、ノイズの影響を受けないことになる。

一般に、計算コストは時系列に内在する問題である。すなわち、大規模時系列の場合、すべてのデータ

[†] 神戸大学大学院工学研究科, 神戸市

Graduate School of Engineering, Kobe University, 1-1 Rokkodai-cho, Nada-ku, Kobe-shi, 657-8501 Japan

a) E-mail: taki@ai.cs.scitec.kobe-u.ac.jp

b) E-mail: uehara@kobe-u.ac.jp

(注1): 時系列が多変量時系列の場合も時系列のことをベクトル系列と呼ぶが、本論文では時系列の隣り合う 2 点の座標値を結ぶことにより作成されるベクトルによって構成される系列のことをベクトル系列と呼ぶ。

点の組合せに対して類似度を計算するため、膨大な計算コストがかかるという問題がある。このような問題を解決するために、多くの研究者が高速化手法を提案している [4], [14]。例えば、時系列をシンボル列で表現する Symbolic Aggregate approXimation (SAX) では、次元圧縮を導入して [9]、長さ n の時系列を任意長 w のシンボル列に圧縮して、計算コストの削減を図る手法が提案されている ($w \ll n$)。また、時系列のデータ数は計算コストに大きく関係するので、検索候補を削減するため、R-tree などのインデックスを使用する手法も提案されている [8]。

しかし、これらの手法では、もとのデータの詳細が失われたり、false dismissal が生じてしまう可能性がある。false dismissal とは、本来、検索されるべき解まで検索候補から外してしまうことである。例えば、SAX は時系列をシンボル列で表すため、もとのデータの詳細が失われる。また、検索の高速化のために検索候補数を削る場合、関数によっては、本来、検索結果として返されるべき解まで捨てられてしまう可能性がある。このため、DTW には下界関数、LCSS には上界関数が提案されている [10], [11], [13]。これらの関数は、もとの距離や類似度より必ず小さく、または必ず大きくなる関数を用いて、もとのデータの詳細さを失わず、false dismissal を生じずに、類似性のない時系列のみをフィルタリングする目的で導入されている。

本論文では、AMSS に対して上界関数を導入し、計算コストの軽減について検討する。AMSS での上界関数は、ベクトル系列を合成ベクトル系列へと次元削減して、類似度を求めるものである。合成ベクトル系列の要素数は、もとのベクトル系列の要素数よりも明らかに少なくなるため、計算コストを下げるができる。また、合成ベクトル系列による類似度関数は上界関数となるために、必ずもとのベクトル系列の情報を保有している。最後に、本手法の正当性を評価する実験を行い、類似性計測における効果を示す。

2. AMSS

まず AMSS による時系列間の類似度計算の概要を示す。ここで Q と C を、あるサンプリングレートで得られた、時間と実数値で構成される、それぞれ長さ L_Q と L_C の時系列と仮定する。

$$Q = q_1, q_2, \dots, q_{L_Q} (\in R^k) \quad (1)$$

$$C = c_1, c_2, \dots, c_{L_C} (\in R^k) \quad (2)$$

座標系の影響をなくすため、 Q と C をベクトル系列 $\hat{Q}$ と $\hat{C}$ に変換する。

$$\begin{aligned} \hat{Q} &= (q_2 - q_1), (q_3 - q_2), \dots, (q_{L_Q} - q_{L_Q-1}) \\ &= \vec{q_1}, \vec{q_2}, \dots, \vec{q_N} \quad (N = L_Q - 1) \end{aligned} \quad (3)$$

$$\begin{aligned} \hat{C} &= (c_2 - c_1), (c_3 - c_2), \dots, (c_{L_C} - c_{L_C-1}) \\ &= \vec{c_1}, \vec{c_2}, \dots, \vec{c_M} \quad (M = L_C - 1) \end{aligned} \quad (4)$$

また、 $\vec{q_i}$ と $\vec{c_j}$ の間の角を θ とすると、 $\vec{q_i}$, $\vec{c_j}$ 間の類似度は以下のように計算される。

$$\begin{aligned} Sim(i, j) &= \cos \theta \\ &= \frac{\vec{q_i} \cdot \vec{c_j}}{|\vec{q_i}| |\vec{c_j}|} \end{aligned} \quad (5)$$

もし $\theta > \pi/2$ であれば、 $Sim(i, j)$ は無条件で 0 となる。逆に、 $\theta \leq \pi/2$ ならば、式 (5) により類似度が計算される。また、 $\hat{Q}$ と $\hat{C}$ 間の全体の類似度は以下の再帰関数を用いて計算される。

$$\begin{aligned} S(i, j) &= \text{Max}[S(i-1, j-1) + 2Sim(i, j), \\ &S(i-2, j-1) + Sim(i-1, j) + 2Sim(i, j), \\ &S(i-1, j-2) + Sim(i, j-1) + 2Sim(i, j)] \end{aligned} \quad (6)$$

$$S(i, 1) = S(1, j) = -\infty, S(1, 1) = Sim(1, 1)$$

ここで、 $S(i, j)$ は $\hat{Q}$ の i 番目と $\hat{C}$ の j 番目までの時系列でとり得る最大の類似度となる。最終的に、 $\hat{Q}$ と $\hat{C}$ の類似度は以下の式で得られる。

$$AMSS(\hat{Q}, \hat{C}) = S(N, M) \quad (7)$$

3. 上界関数

上界関数 $UB_{AMSS}(Q, C)$ とは、本来の類似度 $AMSS(Q, C)$ を、より少ない計算コストで近似的に求める関数である。このため、二つの時系列 Q, C の類似性を求める場合、以下の式を満たす必要がある。

$$UB_{AMSS}(Q, C) \geq AMSS(Q, C) \quad (8)$$

ここで、 Q との類似度が ϵ 以上の時系列をデータベースから検索する場合を考える。データベース中の時系列 C に対して、本来の類似度を計算する前に、上界関数を計算する。もし $UB_{AMSS} < \epsilon$ であれば、式 (8) から以下の式が成り立つ。

$$AMSS(Q, C) \leq UB_{AMSS}(Q, C) < \epsilon \quad (9)$$

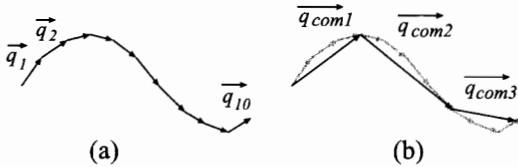


図 1 ベクトル合成の例
Fig. 1 Examples of vector compositions.

よって、 $AMSS(Q, C)$ を計算しなくとも、類似度は ϵ より小さいことが分かる。つまり、 $UB_{AMSS} \geq \epsilon$ の場合にのみ、 $AMSS(Q, C)$ を計算すればよく、候補となる時系列数を減らすことができる。

上界関数の性能は、以下の 2 点に依存している。一つは計算コストである。最低でも、本来の類似度の計算コストよりも小さくなければ、フィルタリングの意味をなさない。本手法の上界関数では、時系列を合成して、もとの時系列より少ない時系列で近似的に表し、計算コストの削減を図っている。

二つ目は緊密性の問題である。緊密性とは、上界関数による類似度と本来の類似度との差である。例えば、 $UB_{AMSS} = \infty$ とすれば、式 (8) は必ず成り立つ。しかし、 ϵ がどんなに大きな値でも $UB_{AMSS} \geq \epsilon$ となってしまうので、結局 $AMSS(Q, C)$ を計算することになる。逆に、緊密性が小さければ、本来の類似度を用いなくてすむので、検索時間を短くすることができる。

3.1 上界関数の定義

図 1 の時系列は、横軸が時間、縦軸が座標値を表している。図 1 (a) の 10 個からなるベクトル系列が図 1 (b) のような 3 個からなる合成ベクトル系列に合成されるものとする。ここで、ベクトル系列の類似性を求める場合、対象となるベクトル数は 3 個に減少するので、計算コストは $\frac{3}{10}$ に削減できる。このように、もとのベクトル系列を近似し、合成ベクトル系列間の類似性を求めれば、類似性検索の効率を上げることができる。

しかしながら、類似度の近似方法が問題となる。例えば、図 2 (a) の $\vec{q}_1$ と $\vec{c}_1$ の類似度と $\vec{q}_{com1}$ と $\vec{c}_1$ の類似度を比較すると、 $Sim(\vec{q}_1, \vec{c}_1) > Sim(\vec{q}_{com1}, \vec{c}_1)$ となってしまう。つまり、合成ベクトルとの類似度よりも、もとの部分ベクトル系列との類似度の方が高くなり、不等式 (8) を満たさない場合が生じ得る。これは、もとの部分ベクトル系列では $\angle \vec{q}_1$ から $\angle \vec{q}_3$ までの角度をすべて考慮しているのに対し、合成ベクトル系列では、単一の $\angle \vec{q}_{com1}$ という角度しか考慮しない

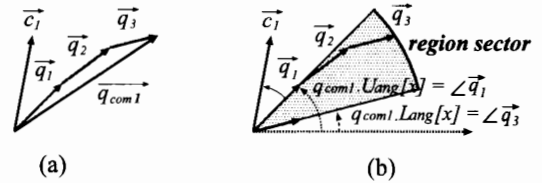


図 2 類似度の比較
Fig. 2 Comparison of similarity.

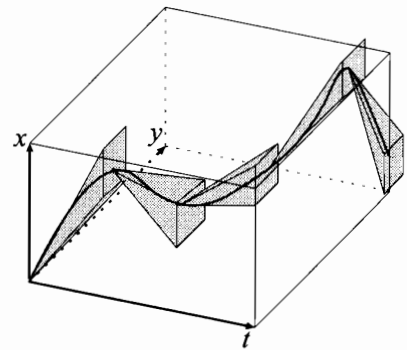


図 3 Region sector 系列
Fig. 3 Region sector sequence.

ためである。

そこで、もとの部分ベクトル系列の角度を考慮するため、合成ベクトルの各次元成分を図 2 (b) のような、もとの部分ベクトル系列の最小角度 ($q_{com1}.Uang[n]$) から最大角度 ($q_{com1}.Uang[n]$) までの角度の範囲により生成される領域 (region sector) で表すことにする。なお、 n は n 次元成分であることを表している。例えば、図 2 の縦軸を x 軸とすると、 $\vec{q}_{com1}$ は $q_{com1}.Uang[x] = \angle \vec{q}_1$ 、 $q_{com1}.Llang[x] = \angle \vec{q}_3$ とする region sector で表されることになる。言い換えると、region sector はもとの部分ベクトル系列を包含しているので、他のどんなベクトルと類似度を比較した場合でも、region sector からそのベクトルまでの角度は、もとの部分ベクトル系列までの角度より大きくなることはない。すなわち、類似度は常に region sector 間の方が大きくなるので、region sector を合成ベクトルとみなせば、上界関数を定義できる。また、時系列が二次元の場合、図 3 のような region sector 間の類似度を求めることが、合成ベクトル間の類似度を求めることになる。このように、region sector は二次元では扇型となるが、三次元では角錐となる。

また、region sector は領域で表されるため、図 4 のように region sector 間が離れている場合 (separa-

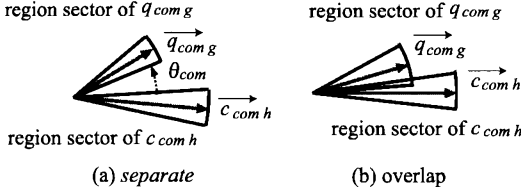


図 4 Region sector 間の類似度を決定する各状態
Fig. 4 Each state to decide similarity between region sector.

tion), region sector 間が重なっている場合 (overlap) の二つの状態がある。重なっている場合とは、合成ベクトルがほとんど類似していることを表すので、類似度を 1 とする。一方、離れている場合は region sector を用いて類似度を計算すればよい。そこで、合成ベクトル系列 $\hat{Q}_{com}$ の g 番目の合成ベクトル $\overrightarrow{q_{com}g}$ と合成ベクトル系列 $\hat{C}_{com}$ の h 番目の合成ベクトル $\overrightarrow{c_{com}h}$ の region sector 間の角を θ_{com} とすると (図 4(a) 参照), $\overrightarrow{q_{com}g}$ と $\overrightarrow{c_{com}h}$ の region sector 間の類似度は以下の式で計算される。

$$Sim_{com}(g, h) = \begin{cases} \cos\theta_{com} & \text{separation} \\ 1.0 & \text{overlap} \end{cases} \quad (10)$$

θ_{com} とは以下の式のように、各座標平面の region sector 間の角度の最小値で表される。

$$\theta_{com} = \min\{\theta_{com}[x], \theta_{com}[y], \dots\} \quad (11)$$

ここで、 x 座標平面での region sector 間の角度 $\theta_{com}[x]$ は以下の式で求められる。

$$\theta_{com}[x] = \begin{cases} q_{com} \cdot L_{ang}[x] - c_{com} \cdot U_{ang}[x] & \text{if } q_{com} \cdot L_{ang}[x] > c_{com} \cdot U_{ang}[x] \\ c_{com} \cdot L_{ang}[x] - q_{com} \cdot U_{ang}[x] & \text{else if } c_{com} \cdot L_{ang}[x] > q_{com} \cdot U_{ang}[x] \end{cases} \quad (12)$$

また、他の平面についても同様にして求めることができる。

そして、もとの部分ベクトルの類似度の定義と同様に、 θ_{com} が $\pi/2$ 以上離れていれば類似度を 0 とする。また、各 region sector のいずれかが図 4(b) のように重なる場合には類似度を 1 とする。

すべての region sector 間の類似度を式 (10) により計算した後、region sector 系列間、つまり、合成ベクトル系列 $\hat{Q}_{com}$ と $\hat{C}_{com}$ 間の類似度を求めるための再帰関数を式 (13) のように定める。

$$S_{com}(g, h) = \max \begin{cases} S_{com}(g-1, h-1) \\ + (2cn_d + cn_{sub_d}) \cdot Sim_{com}(g, h) \\ S_{com}(g-1, h) \\ + (2cn_h + cn_{sub_h}) \cdot Sim_{com}(g, h) \\ S_{com}(g, h-1) \\ + (2cn_v + cn_{sub_v}) \cdot Sim_{com}(g, h) \end{cases} \quad (13)$$

$$cn_d = \text{Min}[\overrightarrow{q_{com}g}.num, \overrightarrow{c_{com}h}.num]$$

$$cn_h = \text{Min}[\overrightarrow{q_{com}g}.num, \overrightarrow{c_{com}h}.num - 1]$$

$$cn_v = \text{Min}[\overrightarrow{q_{com}g}.num - 1, \overrightarrow{c_{com}h}.num]$$

$$cn_{sub_d} = \text{Min}[cn_d, (\overrightarrow{q_{com}g}.num - \overrightarrow{c_{com}h}.num)]$$

$$cn_{sub_h} = \text{Min}[cn_h, (\overrightarrow{q_{com}g}.num - \overrightarrow{c_{com}h}.num)]$$

$$cn_{sub_v} = \text{Min}[cn_v, (\overrightarrow{q_{com}g}.num - \overrightarrow{c_{com}h}.num)]$$

$$UB_{AMSS}(\hat{Q}, \hat{C}) = AMSS_{com}(\hat{Q}_{com}, \hat{C}_{com}) = S_{com}(N', M') \quad (14)$$

ここで $S_{com}(g, h)$ は $\hat{Q}_{com}$ の g 番目と $\hat{C}_{com}$ の h 番目までの合成ベクトル系列間の最大の類似度になる。なお、 $\overrightarrow{q_{com}g}.num$, $\overrightarrow{c_{com}h}.num$ はそれぞれ $\hat{Q}_{com}$ の g 番目と $\hat{C}_{com}$ の h 番目の合成ベクトルを構成するもとのベクトル数を表している。いかなる二つの $\hat{Q}$ と $\hat{C}$ においても、以下の不等式が成り立つ。

$$UB_{AMSS}(\hat{Q}, \hat{C}) \geq AMSS(\hat{Q}, \hat{C}) \quad (15)$$

したがって、以上の手法を上界関数として、似ていない時系列を検索対象から外すことにより、効率的な検索を実現できる。なお、式 (15) については付録で証明する。

4. データ圧縮のためのベクトル合成

時系列中の部分ベクトル列を一つのベクトルに合成する「ベクトル合成」手法について説明する。時系列には、少なくとも滑らかなに変化するデータ、振動の多いデータの 2 種類が存在する。本論文では、これら二つの観点から合成手法を検討している。それぞれ “variational angles” と “variational area” と呼ぶ。

variational angles とは、上界関数の緊密性を高めるために、合成ベクトルにより作られる region sector の中心角に制約をかける条件である。図 2(b) の region sector では中心角 $\overrightarrow{q_{com}g}.U_{ang}[x] - \overrightarrow{q_{com}g}.L_{ang}[x]$ が

しきい値より小さい値をとるように制約をかけている。variational angles は比較的単調な軌跡をしているデータに有効である。逆に、ベクトルの角度が急激に変化している波形では、多くのベクトルを合成することができないという問題がある。

variational area は、ベクトルを合成した場合に生じる、もとのベクトルと合成したベクトル間の領域を情報損の量として考え、その領域の面積を制約条件として用いるものである。variational area は、variational angles ほど角度の変化には敏感にならず、ノイズや激しい変動をする軌道に対しても効果的にベクトル合成できるという利点がある。逆に、variational angles と比べると、多くのベクトルを合成するため緊密性が低くなるという問題がある。

4.1 Variational angles

ベクトル合成はただ闇雲に実行すればよいわけではなく、AMSS から得られる類似度と上界関数から得られる類似度の値が、できるだけ近くなるように合成することが望ましい。そこで、region sector の中心角に制約をかけて、似ているベクトルのみから region sector を構成するようにベクトル合成する。具体的には、合成途中のベクトル系列から作られる region sector の中心角を表すパラメータ θ_{var} が、しきい値 θ_{thresh} より小さければベクトルを合成する。なお、しきい値が小さければ小さいほど、もとのベクトル系列と合成ベクトルがより近い形状となるため緊密性は高くなる。ここで、

$$\theta_{var}[n] = \vec{q}_{com} \cdot U_{ang}[n] - \vec{q}_{com} \cdot L_{ang}[n] \quad (16)$$

であるので、 θ_{var} がすべての次元において θ_{thresh} よりも小さければ、一つの合成ベクトルとして合成される。

variational angles によるベクトル合成アルゴリズムを表 1 に示す。 $\vec{Q}_{com}$ は合成ベクトルを記録する配列である。まずはじめに、しきい値 $Thresh$ を決定する。しきい値は region sector の中心角を決定する値である。また、 $Calc_variational_angle(\vec{q}_j, \vec{q}_i)$ は、 $\vec{q}_j$ と $\vec{q}_i$ の間の角 θ_{var} を計算する関数である。次に合成する $\vec{q}_j$ の角度と θ_{var} を比較して、 θ_{var} の値がしきい値 $Thresh$ を超えるまでそれまでのベクトルをまとめて合成する。

4.2 Variational area

variational area とは、ベクトルを合成した場合に生じる情報損の量を領域で表したものである。例えば、

表 1 Variational angles によるベクトル合成のアルゴリズム

Table 1 Vector combinations algorithm by variational angles.

	Combine_angle ($\vec{Q}$) = $\vec{Q}_{com}$
1	start;
2	$\vec{Q} = (\vec{q}_1, \vec{q}_2, \dots, \vec{q}_L)$; //vector sequence
3	$\vec{Q}_{com} = ()$; //combined vector sequence
4	$Thresh = \theta_{thresh}$;
5	$i = 2$;
6	$j = 1$;
7	while ($i \leq N$)
8	$\theta_{var} = Calc_variational_angle(\vec{q}_j, \vec{q}_i)$;
9	if ($\theta_{var}[1] \leq Thresh \&\& \dots \&\& \theta_{var}[n] \leq Thresh$)
10	$i++$;
11	continue;
12	$\vec{q}_{com} = vector_combine(\vec{Q}, j, i-1)$;
13	add ($\vec{Q}_{com}, \vec{q}_{com}$);
14	$j = i$;
15	$i++$;
16	continue;
17	return $\vec{Q}_{com}$;
18	end;

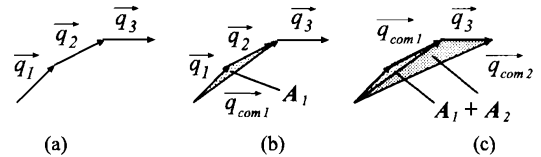


図 5 Variational area

Fig. 5 Variational area.

図 5(a) のベクトル系列のうち、 $\vec{q}_1$ と $\vec{q}_2$ によりベクトル合成が行われた場合 (図 5(b)), 領域 A_1 がこの合成によって生じる variational area となる。また、図 5(c) のように $\vec{q}_1$, $\vec{q}_2$, $\vec{q}_3$ からベクトル合成された場合、 $A_1 + A_2$ が variational area となる。すなわち、ベクトル $\vec{q}_1$ と $\vec{q}_2$ を合成するかどうか決めるとき、variational area A_1 を評価し、 A_1 がしきい値よりも小さい場合、 $\vec{q}_1$ と $\vec{q}_2$ が合成される。同様に、ベクトル $\{\vec{q}_1, \vec{q}_2, \vec{q}_3\}$ の合成については $(A_1 + A_2)$ が評価される。

ベクトル合成のアルゴリズムを表 2 に示す。まずはじめに、しきい値 $Thresh$ を決定する。具体的には、全体時系列の variational area を求めて (表 2 の 3 行目)、情報損の許容量を決定する値 w ($0.0 \leq w \leq 1.0$) を設定する。 w が大きければ、情報損は大きくなり、ベクトル系列本来の形から離れるが、逆にデータ圧縮量は大きくなる。なお、しきい値は $Thresh = A_{total} * w$ として計算される。以上の考え方に従って、一つずつ合成するベクトルを追加

表 2 Variational area によるベクトル合成の例
Table 2 Vector combinations algorithm by variational area.

	Combine_area ($\hat{Q}$) = $\hat{Q}_{com}$
1	start ;
2	$\hat{Q} = (\hat{q}_1, \hat{q}_2, \dots, \hat{q}_N)$; //vector sequence
3	$\hat{Q}_{com} = ()$; //combined vector sequence
4	$A_{total} = \text{Calc_total_variational_area}(\hat{Q})$;
5	$Thresh = A_{total} * w$;
6	$i = 2$;
7	$j = 1$;
8	while ($i \leq N$)
9	$A_{partial} = \text{Calc_variational_area}(i, j)$;
10	if ($A_{partial} \leq Thresh$)
11	$i++$;
12	continue ;
13	$\hat{q}_{com} = \text{vector_combine}(\hat{Q}, j, i-1)$;
14	$\text{add}(\hat{Q}_{com}, \hat{q}_{com})$;
15	$j = i$;
16	$i++$;
17	continue ;
18	return $\hat{Q}_{com}$;
19	end ;

し、その都度 $\text{Calc_variational_area}(i, j)$ により、ベクトル系列 i, j 間の累積 variational area を調べ、しきい値 $Thresh$ を超えるまでベクトル合成する。

5. 実 験

本章では、以下の点に従って、我々の手法が有効であることを示す。

- (1) ベクトル合成の性能
- (2) AMSS のための上界関数の効果

まず、ベクトル合成に関する実験として、データ圧縮の効果を確かめる。そして、効率的な類似度計測の観点から、AMSS の上界関数を用いた場合の検索の正当性を証明する。検索には $k=1$ の場合の近傍検索を用いる。近傍検索とは、問合せ Q と数値 k が与えられたときに、 Q に最も類似した k 個のデータを検索するものである。

上界関数を用いた近傍検索では、実際に検索を行う前に、データベース中のすべての時系列に対して、合成ベクトル列を作成しておく。そして、問合せ時系列 Q が与えられると、表 3 のアルゴリズムを用いて近傍検索を行う。

まずは、variational area か variational angles を用いて Q から合成ベクトル列 $\hat{Q}_{com}$ を作成する。次にデータベースから合成ベクトル列 $\hat{C}_{com}$ を一つ取り出して $UBAMSS(\hat{Q}_{com}, \hat{C}_{com})$ を計算する。これがこれまでの最大類似度より大きい場合には、本来の類似

表 3 上界関数を用いた近傍検索アルゴリズム
Table 3 1 nearest neighbor search algorithm with upper bounds.

	search(Q, C)
1	start ;
2	make $\hat{Q}_{com}$ from Q ;
3	$maxsimilarity = -\infty$;
4	get a $\hat{C}_{com}$ from database;
5	if $\hat{C}_{com}$ does not exist
6	return result;
7	calculate $UBAMSS(\hat{Q}_{com}, \hat{C}_{com})$;
8	if $UBAMSS(\hat{Q}_{com}, \hat{C}_{com}) < maxsimilarity$
9	get a time series C from database;
10	calculate $AMSS(Q, C)$;
11	if $AMSS(Q, C) > maxsimilarity$
12	result = C , $maxsimilarity = AMSS(Q, C)$;
13	go to 4.
14	end ;

表 4 データセットの性質
Table 4 Character of datasets.

Dataset	Length	character
balloon	4,002	complicated
buoy sensor	55,964	complicated
burst	9,382	smooth
network	18,000	smooth
shuttle	6,000	smooth

度 $AMSS(Q, C)$ を計算する。これも最大類似度より大きい場合には最大類似度を更新する。そして、次の合成ベクトル列を取り出して同様の処理を行う。これをすべての時系列に対して行って、最終的に最大類似度をもつ時系列を検索結果として返す。

5.1 ベクトル合成によるデータ圧縮

データ圧縮の効果を確かめるために、variational angles のしきい値を $0 \leq \theta_{thresh} \leq 45$, variational area における情報損の許容量を決定する重み w を $0.0 \leq w \leq 0.5$ の範囲とし、ベクトル合成の実験を行う。なお、本実験では、[4] で使用されている一次元のデータセットを用いている。このデータは時系列のデータマイニングのために無償で提供されているデータであり、ノイズを含むデータ、周期性のあるデータなど様々な 32 のデータセットで構成されている。その中で、本節では表 4 のものを用いている。また、次節からの実験においても同一のデータセットを用いている。

図 6 は variational angles を用いたデータ圧縮の結果を示している。横軸は region sector の中心角のしきい値 θ_{thresh} 、縦軸は圧縮率 *compression rate* である。圧縮率は、もとの時系列の長さに対する圧縮された時系列の長さの割合で表している。結果から、

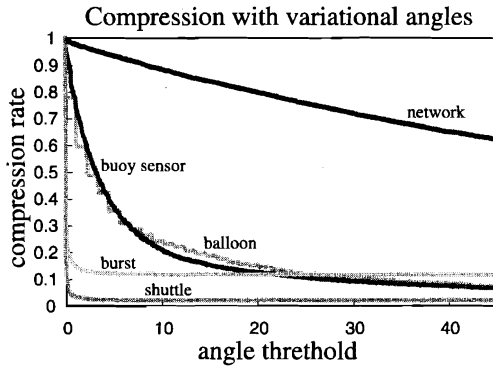


図 6 Variational angles を用いたデータ圧縮の結果
Fig. 6 Data reduction by variational angles.

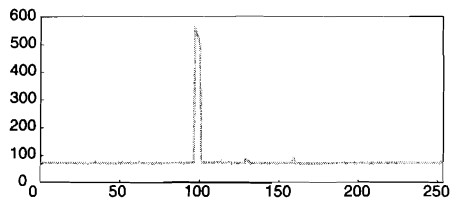


図 7 時系列 network の軌跡
Fig. 7 Shapes of trajectory of data "network."

shuttle, burst は小さいしきい値でも適切にベクトル合成されていることを表している。これは、これらのデータの軌跡が滑らかで、時系列中のベクトルの方向がそろっているためである。

一方, balloon と buoy sensor の圧縮性能は悪く、しきい値を大きくしないと多くのベクトルが合成されないことが分かる。この原因は、時系列中のベクトルが振動しており、小さなしきい値の制約では、ほとんどベクトル合成されないためである。また, network は図 7 のような滑らかな軌跡をしているにもかかわらず、他のデータと比較して、圧縮の性能が悪い結果となっている。これは、network の座標値がサンプリングレートと比較して極端に大きいことが原因である。実際, network はサンプリング点 100 あたりで起こっている変局点のほかにも、小さな振動を繰り返して、時系列が変化している。座標値がサンプリングレートと比較して小さい場合、振動時の角度変化も微小なものとなり、適切な圧縮を行うことができる。しかし, network のように座標値が大きい場合では、大きな変局点以外の角度変化でもしきい値より大きな値となるために適切な圧縮が行われない。このことから、variational angles によるベクトル合成は、滑らかな

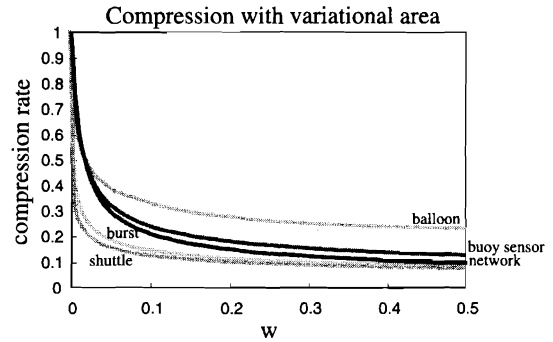


図 8 Variational area を用いたデータ圧縮の結果
Fig. 8 Compression rate by variational area.

軌跡をもつデータに対して有効であることが分かる。しかしながら、データのサンプリングと座標値の関係によっては有効でない場合もある。

次に variational area を制約条件として用いたベクトル合成の結果を図 8 に示す。横軸は w 、縦軸は圧縮率 *compression rate* を示している。variational area によるベクトル合成では、variational angles のようにしきい値による圧縮率のばらつきが少なく、すべてのデータに対して適切に圧縮できていることが分かる。

この実験より、ベクトル合成により十分にデータ量が圧縮できることが証明された。これは時系列間の類似度の計算を効率化するために有効である。しかし、ベクトル系列の大部分を合成してしまうと、上界関数による類似度ともの類似度との緊密性が低くなり、多くの時系列を検索対象から除くことができなくなる。このようなデータ圧縮と上界の性能の関係については次節以降で説明する。

5.2 上界の緊密性

まず上界の緊密性を調べる。緊密性は上界（下界）関数ともの類似度関数との比で表され、1.0 に近いほど、上界（下界）関数の性能が高いといえ、上界関数を用いた検索時に多くの検索候補数の削減を期待できる。ここでは、[4] で掲載されている既存の DTW の下界関数と比較を行う。対象とするのは LB-Kim [8], LB-Yi [14], LB-Keogh [4] である。

LB-Kim では、各時系列から特徴ベクトルを抽出し、下界が定義され、LB-Yi の下界関数は、時系列の値の最大値と最小値を用いて定義されている。そして、LB-Keogh はワーピング制約内の最大値と最小値を用いて定義されている。

本実験では、Keogh らの研究で用いたものと同じ一

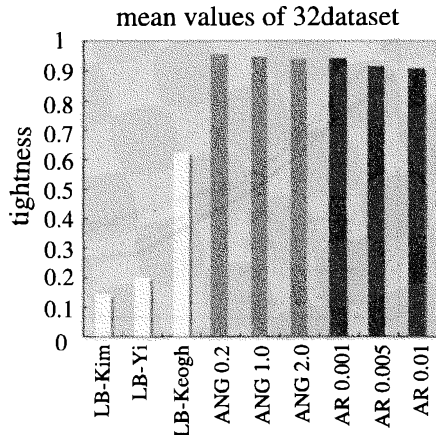


図 9 緊密性の平均
Fig. 9 Means of tightness.

次元データのデータセットを用いている [4].

実験のために、それぞれの 32 のデータセットから長さ 254 の時系列をランダムに 50 個抽出する。そして、50 個のうちの 1 個から他の 49 個の時系列を比較して、緊密性を計算する。この過程をすべての時系列で行い、平均値を算出する。

すべてのデータセットの緊密性の平均を図 9 に示す。本実験では、AMSS の上界の緊密性を variational angles ($\theta_{threth} = 0.2, 1.0, 2.0$) と variational angles ($w = 0.001, 0.005, 0.01$) について調べている。同様に、DTW の下界である LB-Kim, LB-Yi, LB-Keogh についても調べている。

結果から、variational angles, variational area によるベクトル合成は、既存手法で最も緊密性の高い LB-Keogh より更に緊密であることが確認できる。これは、本手法によるデータ圧縮で定義された上界が、類似時系列検索の効率化に有効であることを示している。すなわち、緊密性の高い上界を用いれば、より低い計算コストでの類似時系列の検索が可能になる。

5.3 上界によるフィルタリング

最後に、類似する時系列データの検索におけるフィルタリングの効果を示す。上界を用いた検索を実行し、フィルタリング率を求め、[4] で掲載されている DTW の下界関数によるフィルタリング率と比較する。フィルタリング率とは、どれだけ似ていない検索対象を外すことができたかを表す値で、1.0 に近いほど上界 (下界) 関数の性能が良いといえる。更に、この値が高いほど、検索にかかる時間を短縮することができる。

表 5 検索結果の比較
Table 5 Comparison of retrieval results.

	AMSS	AMSS with Upper Bound
data0	data3	data3
data1	data17	data17
data2	data19	data19
data3	data0	data0
data4	data38	data38
data5	data22	data22
data6	data39	data39
data7	data9	data9
data8	data9	data9
data9	data44	data44

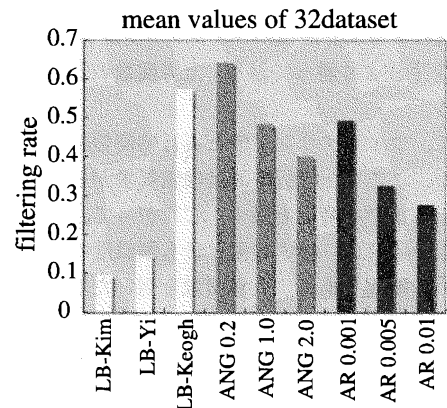


図 10 フィルタリング率の平均
Fig. 10 Means of filtering rate.

また、フィルタリングにより false dismissal を生じないことを付録 2. で数学的な証明により、上界関数の類似度がもとの類似度よりも大きくなることを示している。そして、すべての検索対象を比較すると、実際に false dismissal が生じないことを確認できた。確認結果の一部として、データ network の data0~data9 に対して検索を行った確認結果を表 5 に示す。ここで、上界関数には variational angles ($\theta_{threth} = 1.0$) を用いている。

そして、すべてのデータセットのフィルタリング率の平均を図 10 に示す。本実験では、ベクトル合成には variational angle ($\theta_{threth} = 0.2, 1.0, 2.0$) と variational angles ($w = 0.001, 0.005, 0.01$) を用いて上界を計算し、フィルタリング率を調べている。同様に、DTW の下界である LB-Kim, LB-Yi, LB-Keogh についても調べている。

結果から、variational angles ($\theta_{threth} = 0.2$) による上界のフィルタリング率が、最も高い値を示している。また、variational angle ($\theta_{threth} = 1.0, 2.0$) と

variational angles ($w = 0.001, 0.005, 0.01$) による上界のフィルタリング率は LB-Keogh と比較すると性能が悪くなっている。これは、AMSS の類似度の定義によるものだと考えられる。

具体的には、DTW では $0 \sim \infty$ までの距離を計算し、類似度を算出しているが、AMSS の類似度、上界関数はともに $0 \sim 1$ までの制限された値しかとることがない。また、比較する時系列が似ていれば似ているほど、類似度は高くなり、緊密性も必然的に高くなる。そのため、緊密性は LB-Keogh より高いが、フィルタリング率は低いという結果が生じている。しかし、ベクトル合成による上界関数のフィルタリング率は LB-Kin, LB-Yi 手法より勝っており、上界関数の妥当性が確認できる。

次に、フィルタリング率と実行時間短縮率の関係を調べる。ここで、実行時間は検索要求がきてから検索を終えるまでの時間を指す。したがって、上界関数を用いない場合は、検索対象と最も類似度の高いデータを求める時間であり、上界関数を用いた場合では表 3 の検索アルゴリズムを実行する時間である。実行時間短縮率は両者の時間の比により表している。使用するデータは 32 のデータセットから特に balloon, shuttle というデータを用いている。これは、shuttle は滑らかな軌跡を描くデータであり、balloon は振幅の激しいデータという特徴をもっていることによる。これらの二つのデータに variational angles, variational area を適用した場合の有効性を検証する。なお、variational angles, variational area はそれぞれ、しきい値によって圧縮率が異なるので、同じ尺度で比較するために、データの圧縮率とフィルタリング率と実行時間短縮率の関係を調べている。

図 11 は balloon, shuttle におけるフィルタリング率と実行時間短縮率を示している。また、DTW に LB-Keogh を用いた場合の実行時間短縮率は balloon では 0.53, shuttle では 0.4 となっている^(注2)。この図において、横軸はベクトル合成を行ったときの圧縮率、縦軸はフィルタリング率と実行時間短縮率である。 T_ang , T_area はそれぞれ、variational angles, variational area を用いた場合の実行時間短縮率、 F_ang , F_area はそれぞれ、variational angles, variational area を用いた場合のフィルタリング率を表している。balloon, shuttle ともに、フィルタリング率が増加すると、実行時間が短縮されることが分かる。すなわち、高いフィルタリング率により、効率的

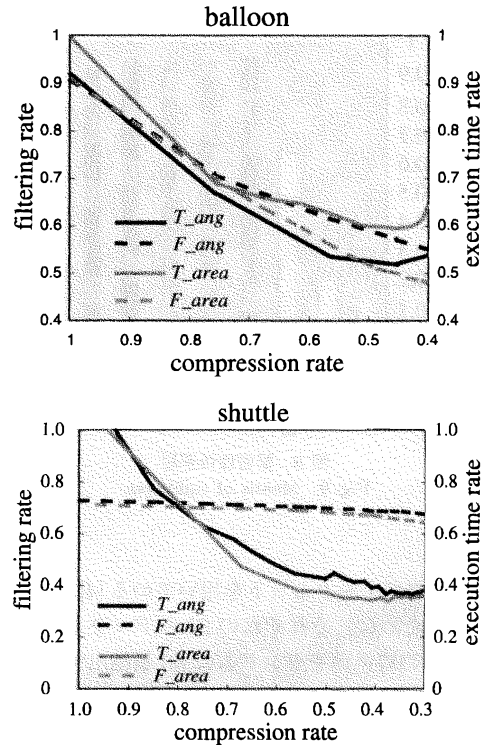


図 11 balloon と shuttle のフィルタリング率と実行時間の短縮率
Fig. 11 Filtering rate and execution time rate of balloon and shuttle.

な検索が可能になっている。

また、shuttle では variational area の優位性がわずかに確認できる。これは、shuttle が滑らかな軌跡であるために、variational angles では多くのベクトルを一度に合成しすぎ、緊密性が低くなっていることが原因であると考えられる。逆に、balloon では variational angles の優位性が確認できる。これは、balloon が振動の激しいデータであるために、variational area では繰り返す方向の違うベクトルを合成してしまい、緊密性が下がっていることが原因であると考えられる。しかしながら、両手法とも、フィルタリングを用いない場合に比べて、LB-Keogh と同程度の検索効率である 40~50%程度の時間で検索できている。

6. む す び

本研究では、類似時系列の検索を高速化するため、

(注2)：実行時間短縮率については [4] で言及がなかったため、[4] に掲載されているように、我々が実装したものである。

表 6 ラベル 4 の test データに対する DTW, AMSS の検索結果上位 3 個

Table 6 Retrieval result of DTW and AMSS for test data of label 4.

AMSS		DTW	
data	class	data	name
train[83]	4	train[392]	4
train[178]	12	train[321]	4
train[114]	4	train[307]	15

ベクトル合成を用いた上界関数によるフィルタリングを実装した。本手法は、false dismissalを生じず、時系列間の類似度を効率的に計算することができる。このことを実証するために、上界が成り立つことを数学的帰納法によって証明した。ベクトル合成の評価実験では、データ量削減の正当性を証明することができた。また、上界によるフィルタリングを評価して、ベクトル合成が有効であることを示した。

本手法は、「AMSS の時間伸縮による脆弱性」、「variational angles の汎用性」「variational area によるベクトル合成」、「上界関数の緊密性」という問題点を抱えている。

まず、時間伸縮による脆弱性について検証する。AMSS の分類性能自体は [1] で議論を行っており、[5] の一次元のデータセットにおける分類性能は付録の表 A.1 のようになっている。ここで、DTW は最適なワーピング制約を用いたもので、LCSS については 2 点間の差異が 10% 以内であれば類似となるようにしている。そして、最も誤差が少なかった値を太字で示している。これより、AMSS が類似度測定関数として妥当なものであることが確認できる。

しかし、詳細を見ると AMSS は時間伸縮に弱いことが分かる。例えば、データセット 50words 中のラベルが 4 である test というデータを用いた場合、DTW, AMSS の検索結果のうち上位 3 個を表 6 に示す。ここで、train[i] は i 番目の訓練データを示す。

表より、両手法とも正しく検索ができていることが確認できるが、AMSS のラベル 12 のデータの方がラベル 4 のデータよりも類似度が高い結果となっている。これはデータ 50words が時系列中で時間伸縮を起こしており、AMSS では時間伸縮により適切な類似度を求められていないことが原因である。

時間伸縮の脆弱性について更に詳細に検討する。図 12 のような時系列があった場合、図 13(a) は図 12 の時間を 2 倍に伸張したデータ、図 13(b) は時間を 1/2 に短縮したデータを表す。これらのデータのうち

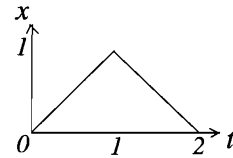


図 12 基準時系列
Fig. 12 Basic time series data.

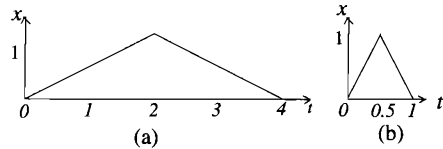


図 13 時間伸縮による時系列の変化
Fig. 13 Transition by time warping.

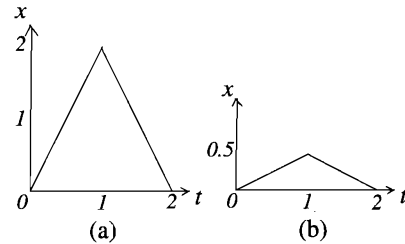


図 14 振幅の変化による時系列の変化
Fig. 14 Transition by amplitude.

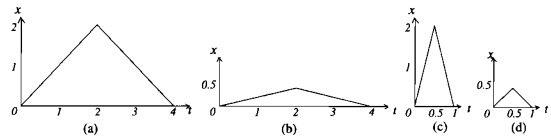


図 15 時間伸縮と振幅の変化を組み合わせた時系列の変化
Fig. 15 Transition by time warping and amplitude.

図 12 と図 13 (a), (b) 間の類似度を DTW, AMSS で求める。DTW では類似度がともに 0 となり、AMSS ではともに 0.95 となる。DTW の類似度は AMSS と異なり、0 に近いほど、時系列が似ているといえる。そのため、時間の伸縮に対して DTW では類似度の変化が起こらず、AMSS に比べて優位が確認できる。

また、図 14, 図 15 のように、振幅の変化や、時間伸縮、振幅変化を組み合わせた変化を与えたものに対して同様に計算すると、表 7 のようになる。結果より、AMSS は、時間伸縮、振幅の変化に対して、類似度の変化が起こるが、時間の伸張と振幅の増加が同時に起こる場合、あるいは時間の短縮、振幅の減少が同時に起こる場合に対しては頑強であることが確認できる。このように、AMSS は DTW と異なり、時系列の

表 7 時系列の変化による類似度の変化
Table 7 Difference of similarity by time series translation.

	時間伸縮		振幅変化		組合せ			
	(a)	(b)	(a)	(b)	(a)	(b)	(c)	(d)
DTW	0	0	1	0.5	1	0.5	1	0.5
AMSS	0.95	0.95	0.95	0.95	1.0	0.86	0.86	1.0

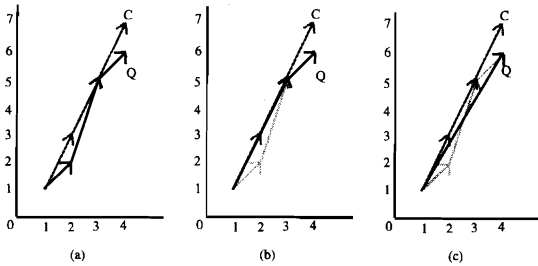


図 16 ベクトル合成による類似度の変化

Fig. 16 Change of similarity by vector composition.

形に依存して類似性が決定するため、単純な時間伸縮には弱いことになる。

次に、variational angles の汎用性について検討する。variational angles は、座標値とサンプリングレートの関係によっては、滑らかな軌跡をしているデータであっても適切にデータ圧縮を行えない。加えて、滑らかな軌跡をしているデータと振動の激しいデータでは、同じしきい値によるデータ圧縮量の差が大きく異なる。このように、一意なしきい値で同様のベクトルを行うことができないため、variational angles は汎用性に欠けている。

次に、variational area についても問題点がある。variational area は、どのようなデータでもデータ量を削減できるが、上界があまり緊密ではない場合がある。これはベクトル合成方法によるものである。例えば、図 16 の (a) のようなベクトル系列 $Q((1,1), (1,3), (1,1))$, $C((1,3), (1,3), (1,3))$ がある場合、それぞれのベクトルの時間軸からの角度は系列 Q では 45 度, 71.6 度, 45 度となり、系列 C では 63.4 度, 63.4 度, 63.4 度となる。これらから類似度を算出すると $\cos(45-63.4) + \cos(71.6-63.4) + \cos(45-63.4) = 2.89$ となる。ここで、(b), (c) のように合成ベクトル系列を生成した場合、上界関数の定義を用いて類似度を求めると (b) のようなベクトル合成では $2 * \cos(71.6-63.4) + \cos(45-63.4) = 2.93$, (c) のようなベクトル合成では $3 * \cos(71.6-63.4) = 2.97$ となる。このように、緊密性の観点からいうと、合成

ベクトル数を増やした方が、緊密性は低くなるが、ベクトルを近似するという観点では、(c) のベクトル合成の方がもとのベクトル系列からの誤差が小さく、全体としては近似の精度が上がる場合も存在する可能性がある。

最後に、上界関数の緊密性について述べる。本論文では、ベクトル系列の最大角度と最小角度という二つの要素から上界関数を定義したが、構成ベクトル系列の角度分布を用いれば、より緊密性を高くすることができ、更なる類似性時系列検索の効率化ができると考えられる。今後は、これらの弱点を改善する必要がある。

文 献

- [1] 中村哲也, 瀧 敬士, 上原邦照, “AMSS: 時系列データの効率的な類似度測定手法,” 信学論 (D), vol. J91-D, no. 11, pp. 2579–2588, Nov. 2008.
- [2] D.J. Berndt and J. Clifford, “Finding patterns in time series: A dynamic programming approach,” *Advances in Knowledge Discovery and Data Mining*, pp. 229–248. AAAI, 1996.
- [3] G. Das, D. Gunopulos, and H. Mannila, “Finding similar time series,” *Proc. First European Symposium on Principles of Data Mining and Knowledge Discovery*, pp. 88–100, 1997.
- [4] E. Keogh, “Exact indexing of dynamic time warping,” *Proc. 28th International Conference on Very Large Data Bases*, pp. 406–417, 2002.
- [5] E. Keogh, The UCR Time Series Data Mining Archive, 2006. <http://www.cs.ucr.edu/~eamonn/TSDMA/index.html>
- [6] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, “Locally adaptive dimensionality reduction for indexing large time series databases,” *Proc. 2001 ACM SIGMOD International Conference on Management of Data*, pp. 151–162, 2001.
- [7] E. Keogh, X. Xi, L. Wei, and C.A. Ratanamahatana, The UCR Time Series Classification/Clustering Homepage, 2006. http://www.cs.ucr.edu/~eamonn/time_series_data/
- [8] S.-W. Kim, S. Park, and W.W. Chu, “An index-based approach for similarity search supporting time warping in large sequence databases,” *Proc. 17th International Conference on Data Engineering*, pp. 607–614, 2001.
- [9] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, “A symbolic representation of time series with implications for streaming algorithms,” *Proc. 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pp. 2–11, 2003.
- [10] 大桃 諭, 陳 漢雄, 古瀬一隆, 大保信夫, “タイムワーピングに基づく時系列データの類似検索,” *DBSJ*, 分冊 4, no. 1, pp. 17–20, 2005.

- [11] Y. Sakurai, M. Yoshikawa, and C. Faloutsos, "FTW: Fast similarity search under the time warping distance," Proc. 24th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, pp.326–337, 2005.
- [12] Y. Tanaka, K. Iwamoto, and K. Uehara, "Discovery of time-series motif from multi-dimensional data based on MDL principle," Mach. Learn., vol.58, no.2-3, pp.269–300, 2005.
- [13] M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, and E. Keogh, "Indexing multi-dimensional time-series with support for multiple distance measure," Proc. 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp.216–225, 2003.
- [14] B-K. Yi, H.V. Jagadish, and C. Faloutsos, "Efficient retrieval of similar time sequences under time warping," Proc. 14th International Conference on Data Engineering, pp.201–208, 1998.

付 録

1. AMSS の分類性能

一次元時系列に対して, Euclidean 距離, DTW, LCSS, AMSS による 1-Nearest Neighbor を用いた分類性能を表 A.1 に示す. 実験には, UCR Time Series Classification/Clustering Page [7] で配布されている 20 個のデータセットを使用している.

2. 上界の証明

上界関数を用いたフィルタリングで false dismissal を生じさせないためには, 不等式 (15) を証明する必要がある. 不等式 (15) が成り立つことは, 数学的帰納法によって証明できる. なお, 合成ベクトル $\vec{q}_{com\ g}$ と $\vec{c}_{com\ h}$ は, それぞれ以下の部分時系列のベクトルからなるものとする.

$$\vec{q}_{com\ g} = \vec{q}_s, \dots, \vec{q}_x, \dots, \vec{q}_t \quad (\text{A.1})$$

$$\text{if } g = 1 \text{ then } s = 1, \text{ if } g = N' \text{ then } t = N$$

$$\vec{c}_{com\ h} = \vec{c}_u, \dots, \vec{c}_y, \dots, \vec{c}_v \quad (\text{A.2})$$

$$\text{if } h = 1 \text{ then } u = 1, \text{ if } h = M' \text{ then } v = M$$

ここで, 以下の三つの式が成り立つと仮定する.

$$S_{com}(g, h-1) \geq S(x, u) \quad (\text{A.3})$$

$$S_{com}(g-1, h) \geq S(s, y) \quad (\text{A.4})$$

$$S_{com}(g-1, h-1) \geq S(s, u) \quad (\text{A.5})$$

この仮定のもとで, 以下の三つの式が成り立つことを示す.

表 A.1 最近傍分類におけるユークリッド距離, DTW, LCSS, AMSS の性能

Table A.1 Classification performance of Euclidean distance, warping window DTW, LCSS, AMSS with 1-nearest neighbor.

Dataset	Euclidean	DTW	LCSS	AMSS
Gun-Point	0.087	0.087	0.040	0.000
Trace	0.240	0.010	0.090	0.000
Fish	0.217	0.160	0.114	0.040
OSU Leaf	0.483	0.384	0.306	0.103
Swedish Leaf	0.213	0.157	0.290	0.104
Coffee	0.25	0.179	0.214	0.143
Adiac	0.389	0.391	0.675	0.345
Beef	0.467	0.467	0.500	0.433
50 Words	0.369	0.242	0.352	0.242
Lighting-2	0.246	0.131	0.213	0.180
Face (four)	0.216	0.114	0.307	0.261
Face (all)	0.286	0.192	0.393	0.275
Lighting-7	0.425	0.288	0.438	0.301
CBF	0.148	0.004	0.084	0.522
Synthetic Control	0.120	0.017	0.273	0.527
Wafer	0.005	0.005	0.017	0.011
ECG	0.120	0.120	0.190	0.170
OliveOil	0.133	0.167	0.500	0.200
Two Patterns	0.09	0.0015	0.000	0.090
Yoga	0.170	0.155	0.149	0.158

$$S_{com}(g, h) \geq S(x, v) \quad (\text{A.6})$$

$$S_{com}(g, h) \geq S(t, y) \quad (\text{A.7})$$

$$S_{com}(g, h) \geq S(t, v) \quad (\text{A.8})$$

式 (10) より, 合成ベクトル間ともののベクトル間の類似度の関係は以下ようになる.

$$Sim(x, y) \leq Sim_{com}(g, h) \quad (\text{A.9})$$

図 A.1 は, 部分ベクトル系列 $\vec{Q}_{com}$ と $\vec{C}_{com}$ の類似度を求めるための再帰関数を, ベクトル $\vec{q}_x, \vec{c}_y$ 間の類似度を用いて表したものである. ここで, 太線の大マスは $\vec{q}_{com\ g}, \vec{c}_{com\ h}$ 間の類似度を表し, 太線のマスの中の小マスは $\vec{q}_x, \vec{c}_y$ 間の類似度を表している.

図 A.1 を見ると, region 4 ((x, v) 時点での類似度 $S(x, v)$) へ到達するには, region 1 ((s, u) 時点での類似度 $S(s, u)$), region 2 ((x, u) 時点での類似度 $S(x, u)$), region 3 ((x, u) 時点での類似度 $S(s, y)$) から小マスを通り, 小マスの類似度である $Sim(x, y)$ を足していけばよい. ここで, 式 (6) の制限を受ける場合に region 1 から region 4 へ行くことを考える. 再帰関数によると斜めの移動では小マスの類似度の 2 倍が足し合わされる. つまり region 4 での類似度は region 1 での類似度に $Sim(x, y)$ の値を $(2Min(v-u, t-s) + abs((v-u) - (t-s)))$ 個足す

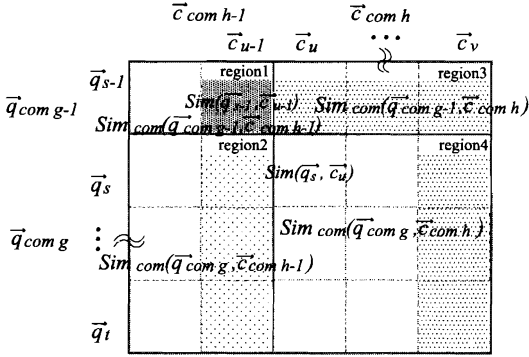


図 A.1 再帰関数の模式図

Fig. A.1 A pattern diagram of recursive function.

ことなる。この結果，不等式 (A.5)，(A.9) から，以下の不等式が得られる。

$$\begin{aligned}
 S(x, v) &\leq S(s, u) + (2\text{Min}(v - u, t - s) \\
 &\quad + |(v - u) - (t - s)|) \text{Sim}(x, y) \\
 &\leq S_{com}(g - 1, h - 1) + (2cn_d + cn_{sub_d}) \\
 &\quad \cdot \text{Sim}_{com}(g, h)
 \end{aligned} \quad (\text{A.10})$$

同様に， $S(x, v)$ が $S(x, u)$ から計算されると，以下の不等式が得られる。

$$\begin{aligned}
 S(x, v) &\leq S_{com}(g - 1, h) + (2cn_h + cn_{sub_h}) \\
 &\quad \cdot \text{Sim}_{com}(g, h)
 \end{aligned} \quad (\text{A.11})$$

また， $S(x, v)$ が $S(s, y)$ から計算されると，以下の不等式が得られる。

$$\begin{aligned}
 S(x, v) &\leq S_{com}(g, h - 1) + (2cn_v + cn_{sub_v}) \\
 &\quad \cdot \text{Sim}_{com}(g, h)
 \end{aligned} \quad (\text{A.12})$$

いかなる x に関しても，これら三つの不等式のいずれも常に真である。これらの不等式と不等式 (13) から，不等式 (A.6) が成り立つ。この証明の過程に従うと，不等式 (A.7) が真であることも同様に証明される。

不等式 (A.8) が成り立つことを示すために，不等式 (A.6)，(A.7) において $x = t, y = v$ とする。そして，不等式 (A.8) で $g = N', h = M'$ とすれば，以下の不等式を得る。

$$S_{com}(N', M') \geq S(N, M) \quad (\text{A.13})$$

最終的に，式 (6)，(14) から，不等式 (15) が成立することが証明できる。

$$\begin{aligned}
 UB_{AMSS}(\hat{Q}, \hat{C}) &= S_{com}(N', M') \\
 &\geq S(N, M) = AMSS(\hat{Q}, \hat{C})
 \end{aligned} \quad (\text{A.14})$$

この結果から，類似する時系列の検索において，false dismissal を生じないことが保証できる。

(平成 20 年 1 月 25 日受付，6 月 11 日再受付)



瀧 敬士

平 19 神戸大・工・情報知能工学卒。現在，同大学院工学研究科博士前期課程在学中。



中村 哲也

平 18 神戸大・工・情報知能工学卒。現在，同大学院工学研究科博士前期課程在学中。



上原 邦昭 (正員)

昭 53 阪大・基礎工・情報工学卒。昭 58 同大学院博士後期課程単位取得退学。同産業科学研究所助手，講師，神戸大学工学部情報知能工学科助教授，同都市安全研究センター教授を経て，現在同大学院工学研究科教授。工博。人工知能，特に機械学習，マルチメディア処理の研究に従事。人工知能学会，計量国語学会，日本ソフトウェア科学会，AAAI 各会員。