Array 15 (2022) 100240

Contents lists available at ScienceDirect

Array

journal homepage: www.elsevier.com/locate/array

Efficient alignment-based average delay time estimation in fluctuating delayed propagation

Atsuyoshi Nakamura *, Tatsuya Hayashi

Graduate School of Information Science and Technology, Hokkaido University, Japan

ARTICLE INFO

Keywords:
Alignment
Time delay estimation
Dynamic time warping
Dynamic programming

ABSTRACT

We propose an alignment-based average delay time estimation algorithm between two time series in the propagation of time-varying delay. Though the number of the minimum cost alignments may be exponential in the length of the time series, the proposed algorithm takes account of all such alignments, and as a post-alignment process, it runs in time linear in the number of the nodes in the *minimum cost alignment graph*, which is at most the length squared. The efficiency of our algorithm is confirmed through numerical experiments compared to the naive enumeration algorithm using recursive calls to traverse the graph.

1. Introduction

An alignment between strings or time series is the position correspondence between them and the minimum-cost alignment for an application-dependent cost function is used in many areas including bioinformatics and signal processing.

Though the purpose of an alignment is different depending on its application, here we consider using it for average delay time estimation between time series. For two time series, the difference between corresponding positions in an alignment can be seen as an estimated delay of one time series from the other time series, so an alignment can be used to estimate the average delay time between them in the propagation of time-varying delays.

Time delay estimation among signals has been studied well in the fields of sonar and radar systems, seismology, geophysics, etc. [1]. In most studies of those fields, constant delay for a moment is assumed and the cross-correlation method [2] is most widely used for the estimation. Improvement using more realistic models [3,4] and spatial prediction technique [5] has been done since then.

For the time-varying time delay estimation, a method using a kind of alignment, which is called DTW (Dynamic Time Warping) [6], has been already proposed in the study area of seismology [7]. No consideration, however, has been done yet on how to efficiently calculate average time delay in the case with a large number of the minimum cost alignment paths.

The minimum-cost alignment is unique with high probability when continuous values can be taken in time series, however, there might be many those alignments when finite values only can be taken in them. In such a case, what we can do to estimate the average delay time is to calculate the delay time averaged over all the aligned positions in all the minimum-cost alignments, which we call the *mean time delay by the minimum-cost alignments*. You can enumerate all the minimum-cost alignments and calculate their delay time at each aligned position one by one, then average them. Unfortunately, this strategy is inefficient in the worst case because the number of the minimum-cost alignments can be exponential in the length of the time series.

In this paper, we propose a method to calculate mean delay time by the minimum-cost alignments in time and space linear in the number of vertices in the *minimum-cost alignment path graph*, in addition to $O(T^2)$ time and space needed for length-T time series alignment. In the graph that is composed of cells and minimum-cost edges between them in the alignment cost table for dynamic programming, the minimum-cost alignment path graph is the subgraph induced by the minimum-cost alignment paths. Since each aligned position in the cost table corresponds to a diagonal edge on the path, delay time for each edge is added only once by multiplying the number of the minimum-cost alignment paths that pass through the edge in our method. Our numerical experiments confirm computational efficiency and estimation accuracy of average delay time by our method.

There are two major ways of aligning two sequences so as to match the corresponding positions. One is gap insertion, which is used in bioinformatics [8], and the other is DTW, which is used in speech recognition [9]. For both ways, the minimum cost alignments are known to be calculated efficiently using dynamic programming. We show the calculation method of our mean delay time for each way considering their difference.

* Corresponding author.
E-mail addresses: atsu@ist.hokudai.ac.jp (A. Nakamura), thayashi@ist.hokudai.ac.jp (T. Hayashi).

https://doi.org/10.1016/j.array.2022.100240
Received 12 February 2022; Received in revised form 19 July 2022; Accepted 21 July 2022
Available online 26 July 2022

This paper is organized as follows. In Section 2, we define mean delay time by minimum-cost alignments and show examples of its calculation for gap-based and warping-based costs. Its efficient calculation way for the warping-based cost is proposed with its process for our introduced example, and the time and space complexities are analysed in Section 3. In Section 4, we show the efficiency of our calculation way by numerical experiments compared with the naive calculation way. We conclude by summarizing the paper and describing its future direction in Section 5. Calculation way for the gap-based cost is explained in Appendix to clarify the slight difference in calculations between the two types of costs.

2. Mean delay time by minimum-cost alignments

Notation $[n]$ for any natural number n denotes the set $\{1,2,\dots,n\}$. Let Y be a subset of $\mathbb{R}$. For $i=1,2$, let $\mathbf{s}_i$ denote the time series of length T whose tth value is $s_i[t]\in Y$, that is, $\mathbf{s}_i = s_i[1]\cdots s_i[T]$. Let Π denote the set of *shift function pairs* which is defined as the set of strictly increasing function pairs (π_1,π_2) from $[T]$ to $[2T-1]$ for which $\pi_1([T])\cup\pi_2([T])$ is a set of contiguous natural numbers starting from 1, that is,

$$\Pi = \{(\pi_1,\pi_2)\,|\pi_i(1)<\cdots<\pi_i(T)\quad (i=1,2),\\ \pi_1([T])\cup\pi_2([T]) = [\max(\pi_1([T])\cup\pi_2([T]))]\}.$$

We let Π_1 denote the subset of Π that is composed of pairs (π_1,π_2) satisfying $\pi_1(1)=\pi_2(1)=1$. An *alignment* between $\mathbf{s}_1$ and $\mathbf{s}_2$ defined by a shift function pair $(\pi_1,\pi_2)\in\Pi$ is the position correspondence in which $s_1[\pi_1^{-1}(k)]$ corresponds to $s_2[\pi_2^{-1}(k)]$ for $k\in\pi_1([T])\cap\pi_2([T])$, where π_i^{-1} is the inverse function of π_i.

There are mainly two types of alignment cost functions, warping-based and gap-based. Consider a cost function between values $w : (Y\cup\{\_\})\times(Y\cup\{\_\})\to\mathbb{R}$, where $\_$ is the special value corresponding to a gap. Then, *alignment cost* $S(\mathbf{s}_1,\mathbf{s}_2,(\pi_1,\pi_2))$ is defined by

$$S(\mathbf{s}_1,\mathbf{s}_2,(\pi_1,\pi_2)) = \sum_{k\in\pi_1([T])\cup\pi_2([T])} w(s_1'[k],s_2'[k]).$$

Here, for $i=1,2$,

$$s_i'[k] = s_i[\underline{\pi^{-1}}(k)]$$

in warping-based cost, where $\underline{\pi^{-1}}(k)=\max\{h\mid\pi_i(h)\le k\}$, and

$$s_i'[k] = \begin{cases} s_i[\pi_i^{-1}(k)] & (k\in\pi_i([T]))\\ \_ & (\text{otherwise}),\end{cases}$$

in gap-based cost. Note that $\underline{\pi^{-1}}(k)=\pi^{-1}(k)$ for $k\in\pi([T])$. Then, the *minimum alignment cost* between $\mathbf{s}_1$ and $\mathbf{s}_2$ is $\min_{(\pi_1,\pi_2)\in\Pi_1} S(\mathbf{s}_1,\mathbf{s}_2,(\pi_1,\pi_2))$ for warping-based cost and $\min_{(\pi_1,\pi_2)\in\Pi} S(\mathbf{s}_1,\mathbf{s}_2,(\pi_1,\pi_2))$ for gap-based cost. Let $\Pi^*(\mathbf{s}_1,\mathbf{s}_2)$ be the set of all the minimum-cost alignments (π_1,π_2) between $\mathbf{s}_1$ and $\mathbf{s}_2$. Then, *mean delay time of* $\mathbf{s}_2$ *from* $\mathbf{s}_1$ *by the minimum-cost alignments* is defined as

$$\frac{\sum_{(\pi_1,\pi_2)\in\Pi^*(\mathbf{s}_1,\mathbf{s}_2)}\sum_{k\in\mathrm{I}(\pi_1,\pi_2)}(\pi_2^{-1}(k)-\pi_1^{-1}(k))}{\sum_{(\pi_1,\pi_2)\in\Pi^*(\mathbf{s}_1,\mathbf{s}_2)}|\mathrm{I}(\pi_1,\pi_2)|},$$

where $\mathrm{I}(\pi_1,\pi_2)$ is the set of aligned positions which is defined to be $\pi_1([T])\cap\pi_2([T])\setminus\{1\}$ for warping-based cost and $\pi_1([T])\cap\pi_2([T])$ for gap-based cost. In the next section, we propose an efficient algorithm for calculating the mean delay time by the minimum-cost alignments.

Example 1. Let $Y=\mathbb{R}$ and consider sequences

$(s_1[1],\dots,s_1[10])$ =(1, 1, 0, −1, −1, 1, 1, 2, 0, −1) and
$(s_2[1],\dots,s_2[10])$ =(0, 1, 1, 0, −1, 1, 1, 1, 2, 0).

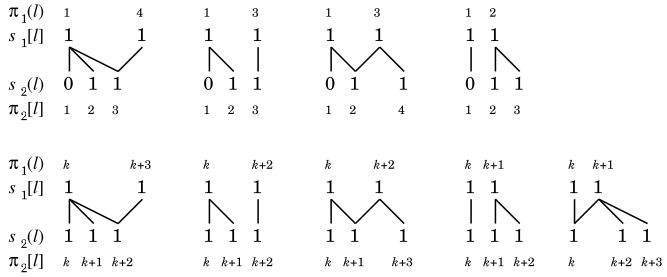
Define cost function w as $w(x,y)=|x-y|$ and shift functions π_1 and π_2 as

$(\pi_1(1),\dots,\pi_1(10))$ =(1, 3, 4, 5, 6, 7, 9, 10, 11, 12) and
$(\pi_2(1),\dots,\pi_2(10))$ =(1, 2, 3, 4, 5, 7, 8, 9, 10, 11).

Then, the alignment between $\mathbf{s}_1$ and $\mathbf{s}_2$ defined by the shift function pairs (π_1,π_2) is one of the minimum warping-based cost alignment with cost 2. (See the following table.)

k	1	2	3	4	5	6	7	8	9	10	11	12
$s_1[\pi_1^{-1}(k)]$	1		1	0	−1	−1	1		1	2	0	−1
$s_2[\pi_2^{-1}(k)]$	0	1	1	0	−1		1	1	1	2	0	
$s_1'[k]$	1	1	1	0	−1	−1	1	1	1	2	0	−1
$s_2'[k]$	0	1	1	0	−1	−1	1	1	1	2	0	0
$w(s_1'[k],s_2'[k])$	1	0	0	0	0	0	0	0	0	0	0	1
$\pi_1^{-1}(k)$	1		2	3	4	5	6		7	8	9	10
$\pi_2^{-1}(k)$	1	2	3	4	5		6	7	8	9	10	
$\pi_2^{-1}(k)-\pi_1^{-1}(k)$	0		1	1	1		0		1	1	1	

For this alignment, $\mathrm{I}(\pi_1,\pi_2)=\{3,4,5,7,9,10,11\}$ and $(\sum_{k\in\mathrm{I}(\pi_1,\pi_2)}(\pi_2^{-1}(k)-\pi_1^{-1}(k)),|\mathrm{I}(\pi_1,\pi_2)|)=(6,7)$. Considering following variations,



there are 20 alignments that achieve the minimum cost 2 and $(\sum_{k\in\mathrm{I}(\pi_1,\pi_2)}(\pi_2^{-1}(k)-\pi_1^{-1}(k)),|\mathrm{I}(\pi_1,\pi_2)|)$ for them are $(4,5)$ for 6 alignments, $(4,6)$ for 5 alignments, $(4,7)$ for 1 alignment, $(5,6)$ for 5 alignments, $(5,7)$ for 2 alignments and $(6,7)$ for 1 alignment. Thus, the mean delay time of $\mathbf{s}_2$ from $\mathbf{s}_1$ by the minimum-cost alignments is

$$\frac{4\times6+4\times5+4\times1+5\times5+5\times2+6\times1}{5\times6+6\times5+7\times1+6\times5+7\times2+7\times1}=\frac{89}{118}\approx0.754.$$

Example 2. Let $Y=\{0,1\}$ and consider sequences $\mathbf{s}_1=001000100$ and $\mathbf{s}_2=000100010$. For $\alpha\ge2$, we consider alignments of time series $\mathbf{s}_1$ and $\mathbf{s}_2$ using symmetric cost function $w(x,y)$ defined as follows:

$$w(x,y)=\begin{cases}0 & ((x,y)=(0,0),(1,1))\\ 1 & ((x,y)=(0,\_),(\_,0))\\ \alpha & ((x,y)=(0,1),(1,0))\\ \infty & ((x,y)=(1,\_),(\_,1)(\_,\_)).\end{cases}\tag{1}$$

In the alignment using this cost function, each value 1 in one sequence is strongly preferred to be aligned to value 1 in the other sequence by shifting positions unless their position difference is large ($2\times$(position difference) $>\alpha$) or the number of letters 1 is different.

Consider the case with $\alpha=3$. Then, the minimum gap-based alignment cost is 2 and there are 6 alignments whose alignment costs are the minimum. One of the minimum cost alignments between $\mathbf{s}_1$ and $\mathbf{s}_2$ is defined by shift functions $(\pi_1(1),\dots,\pi_1(9))=(2,3,4,5,6,7,8,9,10)$ and $(\pi_2(1),\dots,\pi_2(9))=(1,2,3,4,5,6,7,8,10)$. (See the following table.)

k	1	2	3	4	5	6	7	8	9	10
$s_1[\pi_1^{-1}(k)]$		0	0	1	0	0	0	1	0	0
$s_2[\pi_2^{-1}(k)]$	0	0	0	1	0	0	0	1		0
$s'_1[k]$	␣	0	0	1	0	0	0	1	0	0
$s'_2[k]$	0	0	0	1	0	0	0	1	␣	0
$w(s'_1[k], s'_2[k])$	1	0	0	0	0	0	0	0	1	0
$\pi_1^{-1}(k)$		1	2	3	4	5	6	7	8	9
$\pi_2^{-1}(k)$	1	2	3	4	5	6	7	8		9
$\pi_2^{-1}(k) - \pi_1^{-1}(k)$		1	1	1	1	1	1	1		0

For this alignment,

$$\mathrm{I}(\pi_1, \pi_2) = \{2, 3, 4, 5, 6, 7, 8, 10\} \text{ and}$$

$$\left(\sum_{k \in \mathrm{I}(\pi_1, \pi_2)} (\pi_2^{-1}(k) - \pi_1^{-1}(k)), |\mathrm{I}(\pi_1, \pi_2)| \right) = (7, 8).$$

Similarly,

$$\left(\sum_{k \in \mathrm{I}(\pi_1, \pi_2)} (\pi_2^{-1}(k) - \pi_1^{-1}(k)), |\mathrm{I}(\pi_1, \pi_2)| \right)$$

for the other best alignments are

$$(5, 8), (6, 8), (6, 8), (7, 8), (8, 8),$$

so the mean delay time of $\mathbf{s}_2$ from $\mathbf{s}_1$ by the minimum-cost alignments is

$$\frac{5 + 6 \times 2 + 7 \times 2 + 8}{8 \times 6} = \frac{39}{48} = 0.8125.$$

3. Efficient calculation

The calculation of mean delay time by the minimum cost alignments between two sequences is a time-consuming task when there are many minimum cost alignments. As a solution to this issue, we propose a fast algorithm for this task. In this section, we explain the way of calculating the mean delay time for the warping-based cost. See Appendix with respect to the way of calculating the mean delay time for the gap-based cost.

First, review the popular calculation algorithm for the minimum cost alignment using dynamic programming. Consider the alignment for two time series $\mathbf{s}_1 = s_1[1] \cdots s_1[T]$ and $\mathbf{s}_2 = s_2[1] \cdots s_2[T]$. Denote $D(t_1, t_2)$ be the minimum alignment cost between $s_1[1] \cdots s_1[t_1]$ and $s_2[1] \cdots s_2[t_2]$. Then, $D(t_1, t_2)$ can be represented as the following recursive formula.

$$D(t_1, t_2) = \begin{cases} w(s_1[1], s_2[1]) & (t_1 = t_2 = 1) \\ D(t_1, t_2 - 1) + w(s_1[1], s_2[t_2]) & (t_1 = 1, t_2 > 1) \\ D(t_1 - 1, t_2) + w(s_1[t_1], s_2[1]) & (t_1 > 1, t_2 = 1) \\ \min\{D(t_1 - 1, t_2), D(t_1, t_2 - 1), D(t_1 - 1, t_2 - 1)\} & \\ \qquad + w(s_1[t_1], s_2[t_2]) & (t_1, t_2 > 1). \end{cases}$$

$D(T, T)$ is the minimum alignment cost between $\mathbf{s}_1$ and $\mathbf{s}_2$, and $D(T, T)$ can be calculated by calculating $D(t_1, t_2)$ in the order of $(t_1, t_2) = (1, 1), \ldots, (1, T), (2, 1), \ldots, (2, T), \cdots, (T, 1), \ldots, (T, T)$ using the above recursive formula.

Consider the directed graph $G = (V, E)$ with

$$\begin{aligned} V =& \{(t_1, t_2) \mid t_1, t_2 \in \{1, \ldots, T\}\} \\ E =& \{((t_1, t_2 - 1), (t_1, t_2)) \mid \\ & \qquad D(t_1, t_2) = D(t_1, t_2 - 1) + w(s_1[t_1], s_2[t_2])\} \\ & \cup \{((t_1 - 1, t_2), (t_1, t_2)) \mid \\ & \qquad D(t_1, t_2) = D(t_1 - 1, t_2) + w(s_1[t_1], s_2[t_2])\} \\ & \cup \{((t_1 - 1, t_2 - 1), (t_1, t_2)) \mid \\ & \qquad D(t_1, t_2) = D(t_1 - 1, t_2 - 1) + w(s_1[t_1], s_2[t_2])\}. \end{aligned}$$

Then, all the paths from $(1, 1)$ to (T, T) on G correspond to the minimum cost alignments. We call this graph $G(V, E)$ the *alignment path graph between* $\mathbf{s}_1$ *and* $\mathbf{s}_2$. We also call the induced subgraph of G by all the paths corresponding to the minimum cost alignments, the *minimum cost alignment path graph between* $\mathbf{s}_1$ *and* $\mathbf{s}_2$, and let $G^*(V^*, E^*)$ denote it.

Example 3. D for time series $\mathbf{s}_1$ and $\mathbf{s}_2$ with cost function $w(x, y) = |x - y|$ in Example 1 and its corresponding graph G are shown in Fig. 1. The 20 minimum cost alignments correspond to the paths from $(1, 1)$ to $(10, 10)$ on G.

To calculate the mean delay time of $\mathbf{s}_2$ from $\mathbf{s}_1$ by the minimum-cost alignments, it is enough to calculate two values, $\sum_{(\pi_1, \pi_2) \in \Pi^*(\mathbf{s}_1, \mathbf{s}_2)} \sum_{k \in \mathrm{I}(\pi_1, \pi_2)} (\pi_2^{-1}(k) - \pi_1^{-1}(k))$ and $\sum_{(\pi_1, \pi_2) \in \Pi^*(\mathbf{s}_1, \mathbf{s}_2)} |\mathrm{I}(\pi_1, \pi_2)|$. Each minimum-cost alignment (π_1, π_2) can be represented by a path

$$(\underline{\pi_1^{-1}}(1), \underline{\pi_2^{-1}}(1)), (\underline{\pi_1^{-1}}(2), \underline{\pi_2^{-1}}(2)), \ldots, (\underline{\pi_1^{-1}}(T'), \underline{\pi_2^{-1}}(T'))$$

with $(\underline{\pi_1^{-1}}(1), \underline{\pi_2^{-1}}(1)) = (1, 1)$ and $(\underline{\pi_1^{-1}}(T'), \underline{\pi_2^{-1}}(T')) = (T, T)$ in G, where $T' = \max(\pi_1(T) \cup \pi_2(T))$. Furthermore, for $k \geq 2$,

$$k \in \mathrm{I}(\pi_1, \pi_2) \Leftrightarrow \underline{\pi_i^{-1}}(k - 1) + 1 = \underline{\pi_i^{-1}}(k) \text{ for } i = 1, 2$$

holds. In G, let $B(t_1, t_2)$ be the number of paths from (t_1, t_2) to (T, T) and let $F(t_1, t_2)$ be the number of paths from $(1, 1)$ to (t_1, t_2). Then, the number of the minimum-cost alignments (π_1, π_2) that contains edge $((t_1 - 1, t_2 - 1), (t_1, t_2))$ is calculated as $F(t_1 - 1, t_2 - 1)B(t_1, t_2)$. Using the above facts, $\sum_{(\pi_1, \pi_2) \in \Pi^*(\mathbf{s}_1, \mathbf{s}_2)} |\mathrm{I}(\pi_1, \pi_2)|$ and $\sum_{(\pi_1, \pi_2) \in \Pi^*(\mathbf{s}_1, \mathbf{s}_2)} \sum_{k \in \mathrm{I}(\pi_1, \pi_2)} (\pi_2^{-1}(k) - \pi_1^{-1}(k))$ can be calculated as

$$\begin{aligned} & \sum_{(\pi_1, \pi_2) \in \Pi^*(\mathbf{s}_1, \mathbf{s}_2)} |\mathrm{I}(\pi_1, \pi_2)| \\ =& \sum_{((t_1 - 1, t_2 - 1), (t_1, t_2)) \in E^*} F(t_1 - 1, t_2 - 1) B(t_1, t_2) \text{ and} \\ & \sum_{(\pi_1, \pi_2) \in \Pi^*(\mathbf{s}_1, \mathbf{s}_2)} \sum_{k \in \mathrm{I}(\pi_1, \pi_2)} (\pi_2^{-1}(k) - \pi_1^{-1}(k)) \\ =& \sum_{((t_1 - 1, t_2 - 1), (t_1, t_2)) \in E^*} (t_2 - t_1) F(t_1 - 1, t_2 - 1) B(t_1, t_2). \end{aligned}$$

$B(t_1, t_2)$ can be represented as the following recursive formula:

$$B(t_1, t_2) = \begin{cases} 1 & ((t_1, t_2) = (T, T)) \\ \mathbb{1}\{((t_1, t_2), (t_1, t_2 + 1)) \in E\} B(t_1, t_2 + 1) & (t_1 = T, t_2 < T) \\ \mathbb{1}\{((t_1, t_2), (t_1 + 1, t_2)) \in E\} B(t_1 + 1, t_2) & (t_1 < T, t_2 = T) \\ \mathbb{1}\{((t_1, t_2), (t_1, t_2 + 1)) \in E\} B(t_1, t_2 + 1) & \\ \quad + \mathbb{1}\{((t_1, t_2), (t_1 + 1, t_2)) \in E\} B(t_1 + 1, t_2) & \\ \quad + \mathbb{1}\{((t_1, t_2), (t_1 + 1, t_2 + 1)) \in E\} B(t_1 + 1, t_2 + 1) & (t_1, t_2 < T), \end{cases}$$

where $\mathbb{1}\{\cdot\}$ is an indicator function, that is, $\mathbb{1}\{\cdot\} = 1$ if '$\cdot$' holds and 0 otherwise. $B(t_1, t_2)$ for the vertex (t_1, t_2) in V^* can be obtained by starting from $B(T, T)$ and calculating $B(t_1, t_2)$ in reverse lexicographic order of (t_1, t_2) using this recursive formula.

$F(t_1, t_2)$ can be also expressed by recursive formula as follows:

$$F(t_1, t_2) = \begin{cases} 1 & ((t_1, t_2) = (1, 1)) \\ \mathbb{1}\{((t_1, t_2 - 1), (t_1, t_2)) \in E\} F(t_1, t_2 - 1) & (t_1 = 1, t_2 > 1) \\ \mathbb{1}\{((t_1 - 1, t_2), (t_1, t_2)) \in E\} F(t_1 - 1, t_2) & (t_1 > 1, t_2 = 1) \\ \mathbb{1}\{((t_1, t_2 - 1), (t_1, t_2)) \in E\} F(t_1, t_2 - 1) & \\ \quad + \mathbb{1}\{((t_1 - 1, t_2), (t_1, t_2)) \in E\} F(t_1 - 1, t_2) & \\ \quad + \mathbb{1}\{((t_1 - 1, t_2 - 1), (t_1, t_2)) \in E\} F(t_1 - 1, t_2 - 1) & (t_1, t_2 > 1), \end{cases}$$

$F(t_1, t_2)$ for the vertex (t_1, t_2) in V^* can be obtained by calculating $F(t_1, t_2)$ in lexicographic order of (t_1, t_2) using this recursive formula.

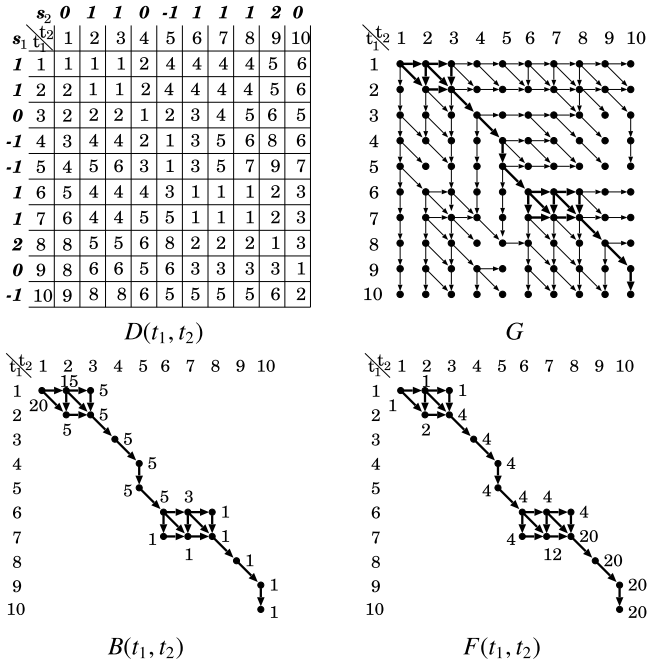


Fig. 1. D for time series $\mathbf{s}_1$ and $\mathbf{s}_2$ with cost function $w(x, y) = |x - y|$ in Example 1, its corresponding graph G, and B and F on the minimum cost paths. The directed edges in the paths from $(1, 1)$ to $(10, 10)$ on G are bolded. $B(t_1, t_2)$s and $F(t_1, t_2)$s for (t_1, t_2) only in the paths corresponding to the minimum cost alignments are shown and $B(t_1, t_2)$s and $F(t_1, t_2)$s for other (t_1, t_2) are 0 and not needed to be calculated.

Example 4. $B(\cdot, \cdot)$ and $F(\cdot, \cdot)$ for time series $\mathbf{s}_1$ and $\mathbf{s}_2$ with cost function $w(x, y) = |x - y|$ in Example 1 is shown in Fig. 1. From the values in B and F, $\sum_{(\pi_1,\pi_2)\in\Pi^*(\mathbf{s}_1,\mathbf{s}_2)} |\mathrm{I}(\pi_1, \pi_2)| = 2\times1\times5+3\times4\times5+2\times4\times1+2\times20\times1 = 118$ and $\sum_{(\pi_1,\pi_2)\in\Pi^*(\mathbf{s}_1,\mathbf{s}_2)} \sum_{k\in\mathrm{I}(\pi_1,\pi_2)} (\pi_2^{-1}(k) - \pi_1^{-1}(k)) = 0\times1\times5+1\times1\times5+2\times1\times4\times5+0\times4\times5+0\times4\times1+1\times4\times1+2\times1\times20\times1 = 89$. Thus, the mean delay time by the minimum cost alignments is $89/118 \approx 0.754$, which coincides with the calculation in Example 1.

Table D and graph G, which are non-original parts, can be constructed in $O(T^2)$ time and space. The rest process is the original part for calculating the mean delay time of $\mathbf{s}_2$ from $\mathbf{s}_1$ by the minimum cost alignments. The construction of tables B and F is done in $O(|V^*|)$ time and space. Calculation of $\sum_{(\pi_1,\pi_2)\in\Pi^*(\mathbf{s}_1,\mathbf{s}_2)} |\mathrm{I}(\pi_1, \pi_2)|$ and $\sum_{(\pi_1,\pi_2)\in\Pi^*(\mathbf{s}_1,\mathbf{s}_2)} \sum_{k\in\mathrm{I}(\pi_1,\pi_2)} (\pi_2^{-1}(k) - \pi_1^{-1}(k))$ using tables B and F is also done in $O(|V^*|)$ time and space. So totally, the rest part specific to our task can be calculated in $O(|V^*|)$ time and space.

The naive calculation using the recursive formula for B can be realized by recursive calls, but its time complexity linearly depends on the number of the minimum cost alignments, which can be exponential to $|V^*|$.

4. Numerical experiment

4.1. Data generation

We generate 100 integer sequence pairs $(\mathbf{s}_1, \mathbf{s}_2)$ for each length $T = 100, 200, \ldots, 1000$ and noise scale $\sigma = 0$ (without noise) as follows. Note that operator % is the modulus operator.

$$s_1[t] = \begin{cases} 0 & (t = 1) \\ s_1[t-1] + r_{1,t} & (t \geq 2) \end{cases}$$

$$s_2[t] = \begin{cases} r_{2,t} & (t = 1) \\ s_2[t-1] + r_{2,t} & (t = 2 \ \&\ \Delta = 2) \\ s_1[t - \Delta_t] + n_t & (\text{otherwise}) \end{cases}$$

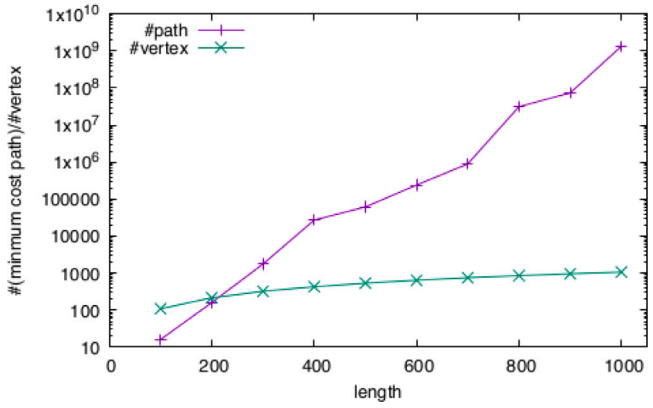


Fig. 2. The numbers of the minimum-cost paths and vertices averaged over 100 sequence pairs with length= $100, \ldots, 1000$ in the minimum cost alignment path graphs.

where

$r_{1,1}, \ldots, r_{1,T}, r_{2,1}, r_{2,2} \sim$ uniform distribution over

$$\{-50, -49, \ldots, 49, 50\},$$

$n_1, \ldots, n_T \sim$ uniform distribution over

$$\{-50\sigma, -50\sigma - 1 \ldots, 50\sigma - 1, 50\sigma\},$$

$$\Delta_t = \begin{cases} 1 & (t = 2 \ \&\ \Delta = 1) \\ 2 & (t = 3 \ \&\ \Delta = 2) \\ \Delta_{t-1} \text{ with prob. } 0.9 & ((t = 3 \ \&\ \Delta = 1) \text{ or } t > 3) \\ 1 + (\Delta_{t-1} \% 2) \text{ with prob. } 0.1 & ((t = 3 \ \&\ \Delta = 1) \text{ or } t > 3) \end{cases} \quad (2)$$

$$\Delta = \begin{cases} 1 & \text{with prob. } 0.9 \\ 2 & \text{with prob. } 0.1 \end{cases}$$

Sequence $\mathbf{s}_1$ is a random sequence in which $s_1[1] = 0$ and the difference between two consecutive terms follows uniform distribution over the set of integers between -50 and 50. Random variable Δ, which takes 1 with probability 0.9 and 2 with probability 0.1, decides whether $s_1[1]$ propagates to $\mathbf{s}_2$ at $t = 2$ or $t = 3$, that is, $s_2[2] = s_1[1]$ if $\Delta = 1$ and $s_2[3] = s_1[1]$ if $\Delta = 2$. For $t \leq \Delta$, $s_2[1]$ and $s_2[t] - s_2[t-1]$ is generated according to uniform distribution over $\{-50, \ldots, 50\}$. For $t > \Delta$, $s_2[t]$ takes $s_1[t - \Delta_t]$, where delay time Δ_t is 1 or 2, and Δ_t takes the same value as Δ_{t-1} with probability 0.9 and a different value with probability 0.1. The number of the minimum-cost paths and vertices averaged over 100 sequence pairs for each length= $100, \ldots, 1000$ in the minimum-cost alignment path graphs is plotted in Fig. 2. You can see that the number of vertices increases linearly, but the number of the minimum-cost paths increases exponentially.

We also generate 100 noisy length-1000 sequence pairs by setting noise scale $\sigma = 0.1, 0.2, \ldots, 1.0$ to check the estimation accuracy of average delay time.

4.2. Experimental setting

Algorithms are implemented by C++ language and executed in Mac Pro (Late 2013) (CPU: 8-Core Intel(R) Xeon(R) E5-1680 v2 @ 3.00 GHz, Memory: 64 GB).

4.3. Results

In the calculation of mean delay time by all the minimum-cost paths, we compare the computational efficiency of our proposed method with that of a naive depth-first search method. The naive depth-first method searches all the minimum-cost alignment paths by traversing the minimum-cost alignment path graph from vertex (T, T) in a depth-first manner.

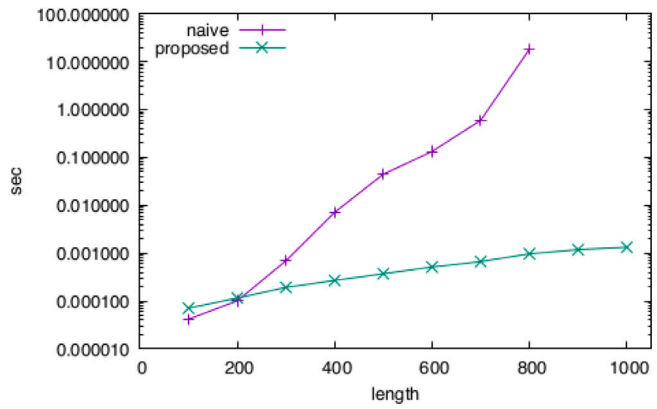


Fig. 3. Average processing time of calculating average delay time by proposed and naive algorithms for sequence pairs with length= $100, \ldots, 1000$. Each average processing time is averaged over 100 sequence pairs. We gave up measuring processing time for length-900 and length-1000 sequence pairs for the naive algorithm due to its longness.

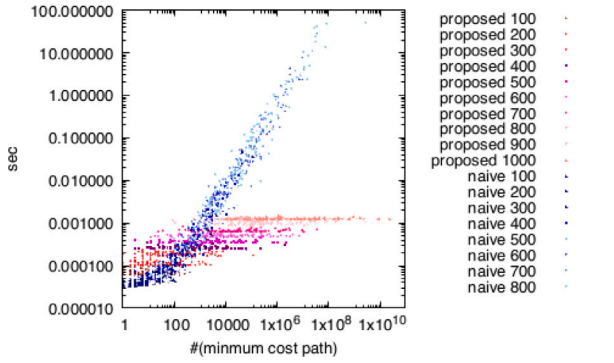


Fig. 4. Scatter (#(minimum-cost paths), processing time) plot of the proposed and naive algorithms.

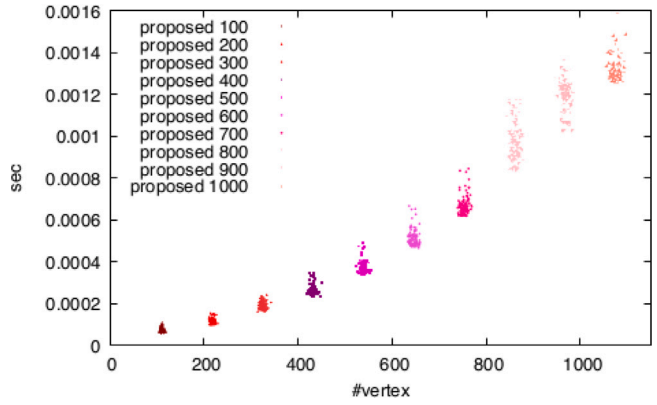


Fig. 5. Scatter (#(vertex number in the minimum-cost alignment path graph), processing time) plot of the proposed algorithm.

The result is shown in Fig. 3. The average processing time of our proposed algorithm increases close to linearly while that of the naive algorithm increases exponentially. The relation between them looks similar to the relation between the numbers of vertices and minimum-cost alignment paths. In fact, the processing time of the naive algorithm linearly increases as a function of the number of the minimum-cost alignment paths though that number does not affect the processing time of the proposed algorithm, as shown in the scatter plot of Fig. 4. A close to linear dependency on the number of vertices can be confirmed for the proposed algorithm by the scatter plot of Fig. 5.

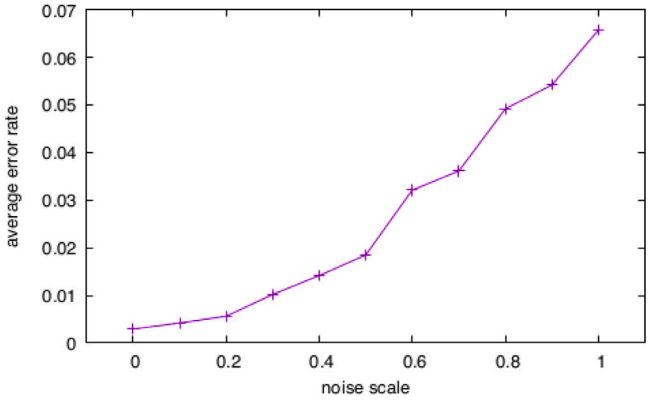


Fig. 6. Average error rate of the proposed algorithm for noise scale $\sigma = 0.1, \ldots, 1.0$.

Finally, we checked the accuracy of the average delay time estimated by our proposed method. The ground truth of average delay time is defined as Δ_t generated by Eq. (2) averaged over $t = \Delta + 1, \ldots, T$. Its error rate is defined as the absolute difference between estimated and true average delay time divided by true average delay time. Error rates averaged over 100 sequence pairs with length 1000 for noise scale $\sigma = 0, 0.1, 0.2, \ldots, 1.0$ are shown in Fig. 6. You can see that the error rates are very small and the rate is less than 7% even when the noise width is the same as the fluctuation width ($\sigma = 1.0$).

5. Conclusions and future work

The proposed alignment-based average delay time estimation algorithm is efficient in the case with a large number of minimum cost alignments; its computation time, excluding the alignment time, increases linearly in the number of nodes in the *minimum cost alignment path graph*, which is at most the time series length squared while the number of the minimum cost alignments may be exponential in the length. In our numerical experiments, the number of the minimum cost alignments between 101-value-range length-1000 time series is more than a billion on average, which indicates that we cannot ignore the issue of a large number of the minimum cost alignments in some applications. Our method can be also applied to time series whose delay time fluctuates intensely, such as stock price and sales histories, so it might be interesting to use it for analyses of such time series in data mining. In fact, our technique developed in this paper was used in the edge set estimation of a propagation graph in applications[1] of stock price and cells' firing analyses [10]. We are now considering the possibility of applying our method to various other fields.

CRediT authorship contribution statement

Atsuyoshi Nakamura: Conceptualization, Methodology, Software, Writing. **Tatsuya Hayashi:** Methodology, Software.

[1] Paper [10] focuses on how to estimate directed edge in a propagation graph using time delay sum (mean delay time that is not divided by the number of aligned positions), and no description is there on how to efficiently calculate it.

Declaration of competing interest

One or more of the authors of this paper have disclosed potential or pertinent conflicts of interest, which may include receipt of payment, either direct or indirect, institutional support, or association with an entity in the biomedical field which may be perceived to have potential conflict of interest with this work. Atsuyoshi Nakamura reports financial support was provided by Japan Society for the Promotion of Science.

Acknowledgements

We would like to thank Prof. Kazuki Horikawa and Prof. Tamiki Komatsuzaki for motivating us to study this research. This work was supported by JSPS KAKENHI Grant Number JP18H05413, Japan.

Appendix. Calculation of mean delay time for the gap-based cost

In the alignment between two sequences $\mathbf{s}_1$ and $\mathbf{s}_2$, either $\mathbf{s}_1$ or $\mathbf{s}_2$ must not be a null time series for the warping-based cost, but both $\mathbf{s}_1$ and $\mathbf{s}_2$ can be null time series for the gap-based cost. Thus, the minimum alignment cost $D(t_1,t_2)$ between $s_1[1]\cdots s_1[t_1]$ and $s_1[1]\cdots s_1[t_2]$ for $t_1 = 0$ or $t_2 = 0$ is needed to be calculated, where $s_1[1]\cdots s_1[0]$ represents the null time series. The recursive formula of $D(t_1,t_2)$ for the gap-based cost is the following:

$$D(t_1,t_2) = \begin{cases} 0 & (t_1 = t_2 = 0) \\ D(t_1,t_2-1) + w(\_, c_2[t_2]) & (t_1 = 0, t_2 > 0) \\ D(t_1-1,t_2) + w(c_1[t_1], \_) & (t_1 > 0, t_2 = 0) \\ \min\left\{\begin{array}{l} D(t_1-1,t_2) + w(c_1[t_1], \_) \\ D(t_1,t_2-1) + w(\_, c_2[t_2]) \\ D(t_1-1,t_2-1) + w(c_1[t_1], c_2[t_2]) \end{array}\right\} & (t_1, t_2 > 0). \end{cases}$$

The directed graph $G(V,E)$ whose paths represent the minimum cost alignments can be constructed as

$$V = \{(t_1,t_2) \mid t_1, t_2 \in \{0,1,\ldots,T\}\}$$

$$\begin{aligned} E = {} & \{((t_1,t_2-1),(t_1,t_2)) \mid D(t_1,t_2) = D(t_1,t_2-1) + w(\_, c_2[t_2])\} \\ & \cup \{((t_1-1,t_2),(t_1,t_2)) \mid D(t_1,t_2) = D(t_1-1,t_2) + w(c_1[t_1], \_)\} \\ & \cup \{((t_1-1,t_2-1),(t_1,t_2)) \mid D(t_1,t_2) = D(t_1-1,t_2) + w(c_1[t_1], c_2[t_2])\}. \end{aligned}$$

All the paths from $(0,0)$ to (T,T) on G correspond to the minimum cost alignments. The number $B(t_1,t_2)$ of paths from (t_1,t_2) to (T,T) can be represented by the same recursive formula as that for the warping-based cost. The recursive formula of $F(t_1,t_2)$ is

$$F(t_1,t_2) = \begin{cases} 1 & ((t_1,t_2) = (0,0)) \\ \mathbb{1}\{((t_1,t_2-1),(t_1,t_2)) \in E\} F(t_1,t_2-1) & (t_1 = 0, t_2 > 0) \\ \mathbb{1}\{((t_1-1,t_2),(t_1,t_2)) \in E\} F(t_1-1,t_2) & (t_1 > 0, t_2 = 0) \\ \begin{array}{l} \mathbb{1}\{((t_1,t_2-1),(t_1,t_2)) \in E\} F(t_1,t_2-1) \\ +\mathbb{1}\{((t_1-1,t_2),(t_1,t_2)) \in E\} F(t_1-1,t_2) \\ +\mathbb{1}\{((t_1-1,t_2-1),(t_1,t_2)) \in E\} F(t_1-1,t_2-1) \end{array} & (t_1, t_2 > 0). \end{cases}$$

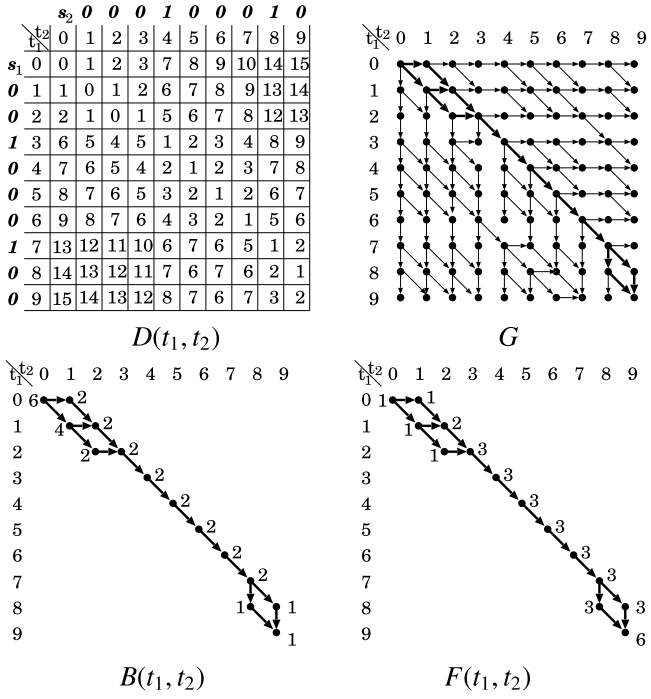


Fig. A.7. D for time series $\mathbf{s}_1$ and $\mathbf{s}_2$ with cost function (1) setting $\alpha = 3$ in Example 2, its corresponding graph G, and B and F on the minimum cost paths. The directed edges in the paths from $(0,0)$ to $(9,9)$ on G are bolded. $B(t_1,t_2)$s and $F(t_1,t_2)$s for (t_1,t_2) only in the paths corresponding to the minimum cost alignments are needed to be calculated.

Example 5. D for time series $\mathbf{s}_1$ and $\mathbf{s}_2$ with cost function (1) setting $\alpha = 3$ in Example 2, its corresponding graph G, the number $B(t_1,t_2)$ of paths from (t_1,t_2) to $(9,9)$ and the number $F(t_1,t_2)$ of paths from $(0,0)$ to (t_1,t_2) on the minimum cost paths in G are shown in Fig. A.7. The minimum cost alignments correspond to the paths from $(0,0)$ to $(9,9)$ in G. From the values in B and F, we can calculate the mean delay time by the minimum cost alignments as

$$\begin{aligned} & 0\times1\times4 + 0\times1\times2 + 1\times1\times2 + 1\times2\times2 \\ & \qquad +5\times1\times3\times2 + 1\times3\times1 + 0\times3\times1 \\ & \frac{}{1\times4 + 2\times1\times2 + 2\times2 + 5\times3\times2 + 3\times1 + 3\times1} = \frac{39}{48} = 0.8125, \end{aligned}$$

which coincides with the calculation in Example 2.

References

[1] Chen J, Huang Y, Benesty J. Time delay estimation. In: Huang Y, Benesty J, editors. Audio signal processing for next-generation multimedia communication systems. Norwell, MA: Kluwer; 2004, p. 197–227.

[2] Knapp C, Carter G. The generalized correlation method for estimation of time delay. IEEE Trans Acoust Speech Signal Process 1976;24(4):320–7. http://dx.doi.org/10.1109/TASSP.1976.1162830.

[3] Manickam T, Vaccaro R, Tufts D. A least-squares algorithm for multipath time-delay estimation. IEEE Trans Signal Process 1994;42(11):3229–33. http://dx.doi.org/10.1109/78.330381.

[4] Benesty J. Adaptive eigenvalue decomposition algorithm for passive acoustic source localization. J Acoust Soc Am 2000;107(1):384–91. http://dx.doi.org/10.1121/1.428310.

[5] Haykin S. Radar array processing for angle of arrival estimation. In: Array signal processing (A85-43960 21-32). Englewood Cliffs. 1985, p. 194–292.

[6] Sakoe H, Chiba S. Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans Acoust Speech Signal Process 1978;26(1):43–9. http://dx.doi.org/10.1109/TASSP.1978.1163055.

[7] Hale D. Dynamic warping of seismic images. Geophysics 2013;78(2):S105–15. http://dx.doi.org/10.1190/geo2012-0327.1.

[8] Mount DW. Bioinformatics - sequence and genome analysis. 2nd ed.. Cold Spring Harbor Laboratory Press; 2004.

[9] Juang B-H. On the hidden Markov model and dynamic time warping for speech recognition—A unified view. AT&T Bell Lab Tech J 1984;63(7):1213–43.

[10] Hayashi T, Nakamura A. Propagation graph estimation from individuals' time series of observed states. Sci Rep 2022;12(6078). http://dx.doi.org/10.1038/s41598-022-10031-3.