SEARCHING THE LONGEST COMMON SUBSEQUENCES IN DISTORTED DATA

Tomáš Kocyan[(a)], Jan Martinovič[(a,b)], Kateřina Slaninová[(a,b)], Daniela Szturcová[(a,c)]

[(a)]IT4Innovations, [(b)] FEECS, Department of Computer Science, [(c)]Institute of Geoinformatics
VŠB - Technical University of Ostrava
17. listopadu 15/2172, 708 33 Ostrava, Czech Republic

tomas.kocyan@vsb.cz, jan.martinovic@vsb.cz, katerina.slaninova@vsb.cz, daniela.szturcova@vsb.cz

ABSTRACT
Many models and artificial intelligence methods work with the inputs in the form of time series. Success of many of them strongly depends on ability to quickly and precisely compare two time series or search the mutual parts. Such ability is especially crucial while recognizing characteristic patterns, indexing, prediction or compression. There are many algorithms able to handle that, however, many of them fail while processing distorted data. Unfortunately, the distortion is natural for many types of data collections, e.g. for measurements of natural phenomena such as precipitations, river discharge volume etc. This paper discusses the possibilities of searching such common subsequences in time series and presents a new approach for searching the longest common subsequences in distorted data. This approach is based on modified the dynamic time warping algorithm, which allows the effective processing distorted time series data.

Keywords: time series, dynamic time warping, longest common subsequence, distorted data

1. INTRODUCTION
Processing and analyzing time series data is very important task in many domains, especially in modeling and simulations. In this domain, time series data is often used as one of the simulation inputs, or can be produced as one of the simulation outputs. For this purpose, it is appropriate to be able to manage this type of data, e.g. describe the data nature, search in data in reasonable time, or to recognize characteristic patterns in a collection. Algorithms providing such functionality usually need a robust mechanism for comparing two time series or identifying their common parts (Kocyan, Martinovič, and Podhorányi, 2014). However, many of such mechanisms fail while processing distorted data (Muller 2007). Unfortunately, the distortion is natural for many types of data collections, e.g. for measurements of natural phenomena such as precipitations, river discharge volume etc. Moreover, methods searching the common subsequences are mostly focused only on processing categorical data. For processing the real data, they have to be modified (Esling and Agon 2012) or some kind of categorization (Lin, Keogh, Wei, and Lonardi 2007) has to be performed.

This paper discusses the possibilities of searching such mutual subsequences in time series and presents a new approach for searching the longest common subsequences in distorted data. It is organized as follows: First, basic approaches for searching the mutual subsequences of categorical data are introduced in Section 2. Second, the DTW algorithm for comparing two distorted sequences will be described. In the Proposed solution, a new approach for searching the mutual subsequences in distorted data will be suggested. At the end, both advantages and disadvantages of the approach will be summarized and the future work will be outlined.

2. COMPARISON OF SEQUENCES
Sequence comparison is widely used in information retrieval and in molecular biology for calculating sequence alignments of proteins (Gusfiel 2008) where several algorithms and their modifications are presented. However, we can use these methods in other areas as well. There are two main basic groups of algorithms known for the comparison of two or more categorical sequences. The first group divides the algorithms by the fact of whether or not the sequences consist of ordered or unordered elements. The second group of algorithms focuses on the comparison of the sequences with different lengths and with possible error or distortion.

2.1. The longest common substring
The basic approach to the comparison of two sequences, in which the order of elements is important, is *The longest common substring method* (LCS). This is used in exact matching problems (Gusfiel 2008). It is obvious from the name of the method that its main principle is to find the length of the common longest substring. Given the two sequences $x = (x_1, x_2, \dots, x_n)$ of the length $n \in \mathbb{N}$ and $y = (y_1, y_2, \dots, y_m)$ of the length $m \in \mathbb{N}$, we can find such substring $z = (z_1, z_2, \dots, z_p)$, $\forall k \in \{1, \dots, p\}(z_k = x_{i+k-1} = y_{j+k-1})$, where $1 \le i \le n - p + 1$ and $1 \le j \le m - p + 1$.

The LCS method respects the order of elements within a sequence. However, the main disadvantage of this method is that it can only find the identical subsequences, which meet the characteristics of substrings. Precisely speaking, this means that the elements in the subsequence must be contiguous. For some domains, in which a large amount of different sequences typically exists, this fact gives too strict limitation to solved the problems.

2.2. The Longest Common Subsequence

Unlike substrings, the objects in a subsequence might be intermingled with other the objects that are not in the sequence. *The longest common subsequence* method (LCSS) allows us to find the common subsequence (Hirschberg 1977). Given the two sequences $x = (x_1, x_2, \dots, x_n)$ of the length $n \in \mathbb{N}$ and $y = (y_1, y_2, \dots, y_m)$ of the length $m \in \mathbb{N}$, we can find the subsequence $z = (z_1, z_2, \dots, z_p)$, $\forall k \in \{1, \dots, p\}(z_k = x_{i_k} = y_{j_k})$, where $i_k < i_{k+1}$ and $j_k < j_{k+1}$. Contrary to the LCS method, the LCSS method allows (or ignores) these extra elements in the sequence, and therefore, it is immune to slight distortions.

2.3. The Time-Warped Longest Common Subsequence

When the similarity between compared sequences is defined as a function using a length of common subsequence, one characteristic of this method can be found. The length of the common subsequence is not immune to the recurrence of the identical elements that may occur only in one of the compared sequences. We can find such situations as a result of inappropriate sampling or any kind of distortion. In some applications it is suitable (or sometimes even required) to eliminate such type of distortions and to work with them like with the equivalent elements. The solution is in another method, the time-warped longest common subsequence (TWLCS) Guo and Siegelmann (2004). This method emphasises the recurrence of elements in one of the compared sequences. Due to this fact, the length of the common subsequence can be, in some cases, longer than the shorter length of the compared sequences.

2.4. Comparison of Real Time Series

Our goal is to search continuous mutual subsequences in distorted real data. However, none of the previously mentioned methods can be directly applied, because they process only a categorical data. There are usually two ways how to adapt them for processing the real data as well: On the one hand, the data can be first categorized as in (Lin, Keogh, Wei, and Lonardi 2007), and then the known methods can be used. On the other hand, some kind of tolerance for equality between two values can be defined as in (Esling and Agon 2012). If such tolerance is not overcame, the values are considered as identical, i.e. belonging into the same category.

Despite the fact there exists some ways how to adapt the methods, it usually brings an unnecessary inaccuracies. They are caused by the degradation real number fineness to an isolated category. We tried to avoid such loss of information while processing the real data, but maintain the main principles of searching subsequences. To deal with a possible distortion, we decided to modify the dynamic time warping algorithm introduced in the following section, and supplement it by useful steps from the previously mentioned methods.

3. DYNAMIC TIME WARPING

Nowadays, searching and comparing the time series databases generated by computers, which consists of accurate time cycles and which achieves a determined finite number of value levels, is a trivial problem. A main attention is focused more likely on the optimization of searching speed. A non-trivial task occurs while comparing or searching the signals, which are not strictly defined and which have various distortions in time and amplitude. As a typical example, we can mention measurement of functionality of human body (ECG, EEG) or the elements (precipitation, flow rates in riverbeds), in which does not exist an accurate timing for signal generation. Therefore, comparison of such sequences is significantly difficult, and almost excluded while using standard functions for similarity (distance) computation. Examples of such signals are presented in Figure 1. A problem of standard functions for similarity (distance) computation consists in sequential comparison of the opposite elements in the both sequences (comparison of elements with the identical indexes).

DTW is a technique for finding the optimal matching of two warped sequences using pre-defined rules (Muller 2007). This approach was used for example for comparison of two voice patterns during an automatic recognition of voice commands (Rabiner 1993). Essentially, it is a non-linear mapping of particular elements to match them in the most appropriate way. The output of such DTW mapping of sequences from Figure 1 can be seen in Figure 2.

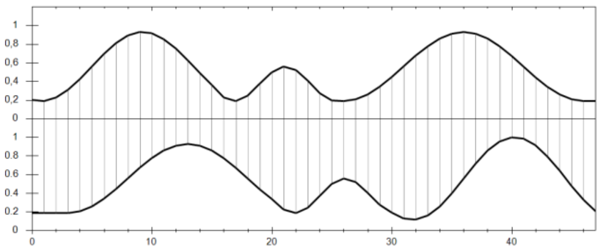

Figure 1: Standard Metrics Comparison

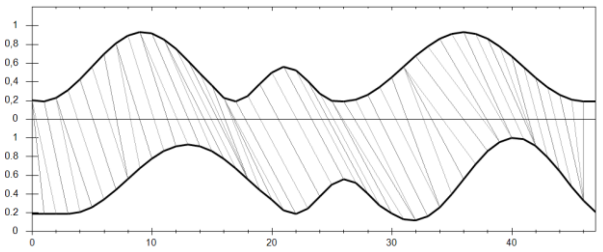

Figure 2: DTW Comparison

The main goal of DTW method is a comparison of two time dependent sequences x and y, where $x =$

$(x_1, x_2, \dots, x_n)$ of length $n \in \mathbb{N}$ and $y = (y_1, y_2, \dots, y_m)$ of length $m \in \mathbb{N}$, and to find an optimal mapping of their elements. To compare partial elements of sequences $x_i, y_j \in \mathbb{R}$, it is necessary to define a local cost measure $c: \mathbb{R} \times \mathbb{R} \to \mathbb{R}_{\geq 0}$, where c is small if x and y is similar to each other, and otherwise it is large.

Computation of the local cost measure for each pair of elements of sequences x and y results in a construction of the cost matrix $C \in \mathbb{R}^{n \times m}$ defined by $C(i,j) = c(x_i, y_j)$ (see Fig. 3). Then the goal is to find an alignment between x and y with a minimal overall cost. Such optimal alignment leads through the black valleys of the cost matrix C, trying to avoid the white areas with a high cost. Such alignment is demonstrated in Fig 4.

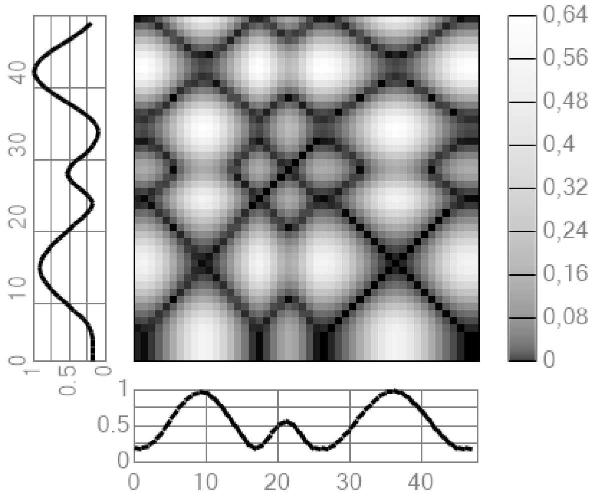

Figure 3: Cost Matrix

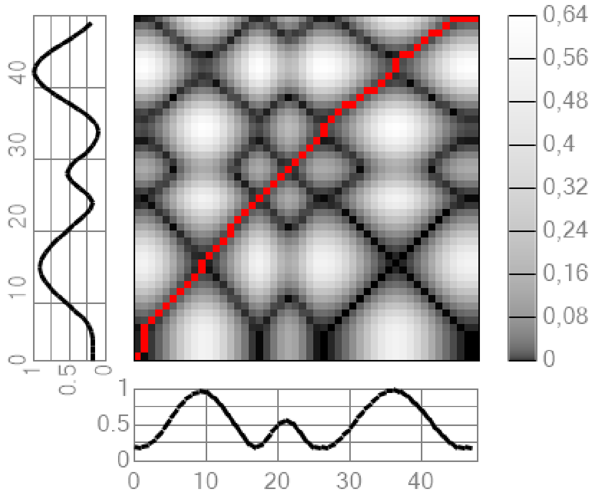

Figure 4: Cost Matrix with Found Warping

Basically, the alignment (called warping path) $p = (p_1, \dots, p_q)$ is a sequence of q pairs (warping path points) $p_k = (p_{kx}, p_{ky}) \in \{1, \dots, n\} \times \{1, \dots, m\}$. Each of such pairs (i,j) indicates an alignment between the ith element of the sequence x and jth element of the sequence y. Moreover, the found warping path has to satisfy the following three conditions:

1. Boundary condition:
   $p_1 = (1,1)$ and $p_q = (n,m)$.
2. Monotonicity condition:
   $p_{1x} \leq p_{2x} \leq \dots \leq p_{qx}$ and
   $p_{1y} \leq p_{2y} \leq \dots \leq p_{qy}$.
3. Step size condition:
   $p_{k+1} - p_k \in \{(1,0), (0,1), (1,1)\}$
   for $k \in \{1, \dots, q-1\}$.

The total cost $c_p(x,y)$ of the found warping path p between sequences x and y is then defined as a sum of partial local costs c:

$$c_p(x,y) = \sum_{k=1}^{q} c(x_{p_{kx}}, y_{p_{ky}})$$

As an optimal warping path p^* between x and y, the warping path having minimal total cost among the set P of all possible warping paths is selected:

$$p^* = \underset{p \in P}{argmin}\{c_p(x,y)\}$$

The DTW "distance" $DTW(x,y)$ between x and y is then defined as:

$$DTW(x,y) = c_{p^*}(x,y)$$

Retrieval of optimal path p^* by evaluating all possible warping paths between sequences x and y leads to an exponential computational complexity. Fortunately, there exists a better way with a $O(n * m)$ complexity based on dynamic programming.

For this purpose, we have to define a function $subSeq: \mathcal{F}^n \times \mathbb{N} \times \mathbb{N} \to \mathcal{F}^m$, where $\mathcal{F}$ is a feature space and $m \leq n$, that creates a new subsequence from a given sequence that is defined as $subSeq(x,a,b) = (x_a, x_{a+1}, \dots, x_b)$, where $a \leq b$. In the rest of the paper, we will use a shortened notation of this function specified as $x_{a:b} = subSeq(x,a,b)$.

For searching the definite optimal warping path, an accumulated cost matrix $D \in \mathbb{R}^{n \times m}$ can be utilized. The partial elements of this matrix are defined as:

$$D(r,s) = DTW(x_{1:r}, y_{1:s})$$

This can be easily computed in following way:

$D(r,1) = \sum_{k=1}^{r} c(x_k, y_1)$ for $r \in \{1, \dots, n\}$,
$D(1,s) = \sum_{k=1}^{s} c(x_1, y_k)$ for $s \in \{1, \dots, m\}$,

$$D(r,s) = min\begin{Bmatrix} D(n-1,m-1), \\ D(n-1,m), \\ D(n,m-1) \end{Bmatrix} + c(x_r, y_s)$$

for $r \in \{2, \dots, n\}$ and $s \in \{2, \dots, m\}$.

Computed accumulated cost matrix for a cost matrix from Fig. 3 can be seen in Fig. 5. It is evident that the accumulation highlights only a single black valley.

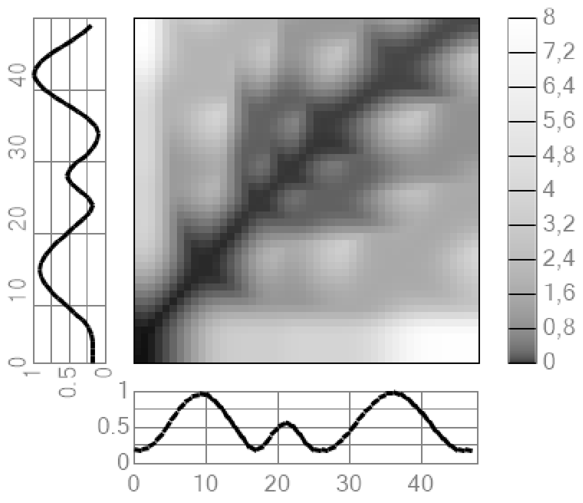

Figure 5: Accumulated Cost Matrix

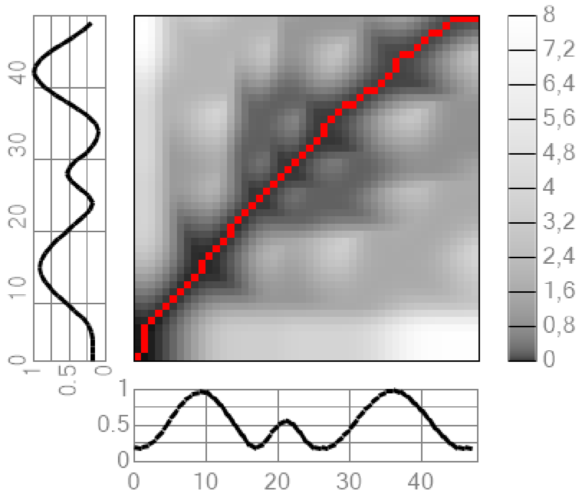

Figure 6: Accumulated Cost Matrix with a Found Warping Path

The optimal path $p^* = (p_1, \dots, p_q)$ is then computed in a reverse order starting with $p_q = (n, m)$ and finishing in a $p_1 = (1,1)$. The rest of the points are defined recursively:

$$p_{k-1} = \begin{cases} (1, p_{ky} - 1) & \text{if } p_{kx} = 1 \\ (p_{kx} - 1, 1) & \text{if } p_{ky} = 1 \\ \underset{(i,j)}{\operatorname{argmin}} \left\{ D(i,j) \mid (i,j) \in \left\{ \begin{matrix} (p_{kx} - 1, p_{ky} - 1), \\ (p_{kx} - 1, p_{ky}), \\ (p_{kx}, p_{ky} - 1) \end{matrix} \right\} \right\} & \text{otherwise} \end{cases}$$

3.1. Subsequence DTW

In some cases, it is not necessary to compare or align the whole sequences. A usual goal is to find an optimal alignment of a sample (a relatively short time series) within the signal database (a very long time series). It is very usual in situations, in which one dispones with a signal database a wants to find the best occurrence(s) of a sample (query). Using the slight modification, the DTW disposes with an ability to search such queries in a much longer sequence. The basic idea is not to penalize the omission in the alignment between x and y that appears at the beginning and at the end of the sequence y. Suppose we have two sequences $x = (x_1, x_2, \dots, x_n)$ of the length $n \in \mathbb{N}$ and $y = (y_1, y_2, \dots, y_m)$ of the much larger length $m \in \mathbb{N}$. The goal is to find a subsequence $y_{a:b} = (y_a, y_{a+1}, \dots, y_b)$ where $1 \leq a \leq b \leq m$ that minimizes the DTW cost to x over the all possible subsequences of y. The modification involves the changed approach in the computation of the first row and column of accumulated cost matrix. The not penalizing the omissions at the beginning and at the end of the sequence y means not accumulating the values in the first column of the accumulated cost matrix, formally:

$$D(1, s) = c(x_1, y_s) \text{ for } s \in \{1, \dots, m\}$$

instead of

$$D(1, s) = \sum_{k=1}^{s} c(x_1, y_k) \text{ for } s \in \{1, \dots, m\}.$$

The remaining values of this matrix are computed in a standard manner as it was described earlier. Then, the warping path connecting the right and left side of the matrix is searched. The searching of the subsequece $y_{a:b}$ begins in a point:

$$(n, b) = \underset{i \in \{1,2,\dots,m\}}{\operatorname{argmin}} \{D(n, i)\}$$

and then it is computed in the same way as in classical DTW until the warping path touches the left side of a matrix in a point $(1, a)$, where $a \in \{1, \dots, b\}$. If the optimal subsequence warping path is defined as $p^* = (p_1, \dots, p_q)$, the $p_1 = (1, a)$ and the $p_q = (n, b)$. By selecting the next greatest value in the last column and searching the warping path from this point again, other paths can be found. An example of such searching the best subsequence alignment can be seen in Fig. 7. The both constructed matrices including the found warping path are then shown in Fig. 8.

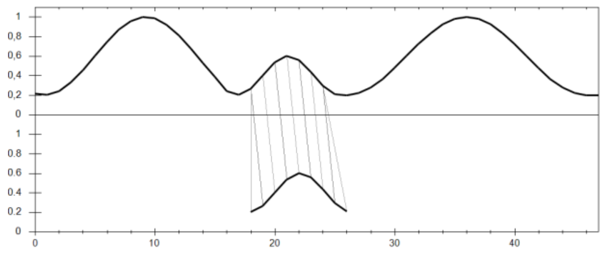

Figure 7: Cost Matrix and Accumulated Cost Matrix for Searching Subsequences

3.2. Searching the mutual subsequences

Despite the fact that the DTW has its own modification for searching subsequences, it works perfectly only in a case of searching an exact pattern in some signal database. However, in real situations, exact patterns are not available because they are surrounded by additional values (Figure 9a), or even repeated several times in the sequence (Figure 9b). Unfortunately, the basic DTW is not able to handle these situations and it fails or returns only a single occurrence of the pattern. To deal with this type of situations, our own DTW modification was created.

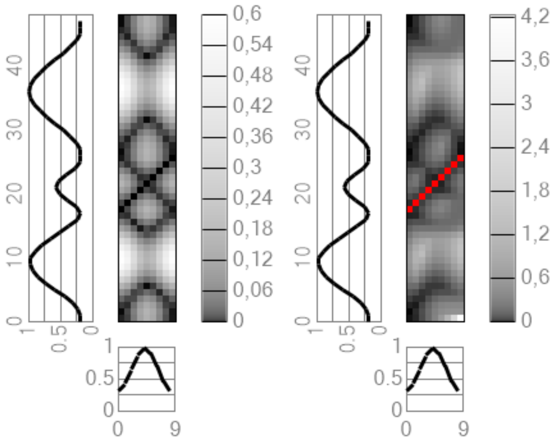
Figure 8: Cost Matrix and Accumulated Cost Matrix for Searching Subsequences

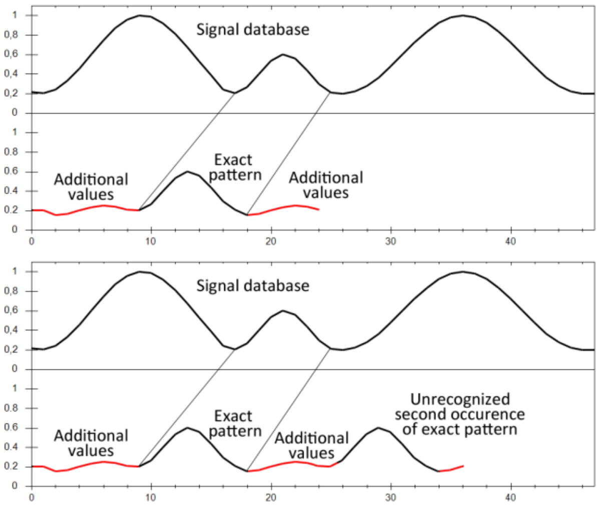

Figure 9: Basic DTW inaccuracies

4. PROPOSED SOLUTION

Proposed solution tries to eliminate weaknesses of previously mentioned approach and allows searching mutual time-warped subsequences respecting predefined conditions. The goal is to find all mutual subsequences as long as possible respecting the following three constrains (numeric thresholds):

- maximal total cost t_t,
- maximal average cost t_a,
- maximal single element cost t_e.

Functionality of the algorithm will be demonstrated on two sequences in the Fig. 10. Illustratively, each of these sequences consists of three same subsequences, but in the different order.

The biggest difference between the classical and subsequence DTW is in the philosophy of searching the warping path. In simple terms, the algorithm does not search the warping path from the upper right corner to the bottom left one (as in the case of classical DTW in Fig. 11a) and also it does not connect the opposite sides of a matrix (as in the case of subsequence DTW in Fig. 11b). The main idea is to find warping paths as long as possible from any element to another one, parallel to a diagonal, as it is outlined in Fig. 11c. Moreover, the constructed warping path have to respect an adjusted constraints.

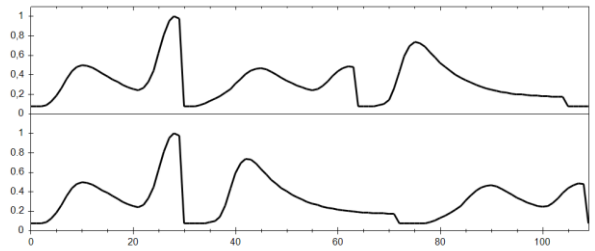
Figure 10: Sample Sequences

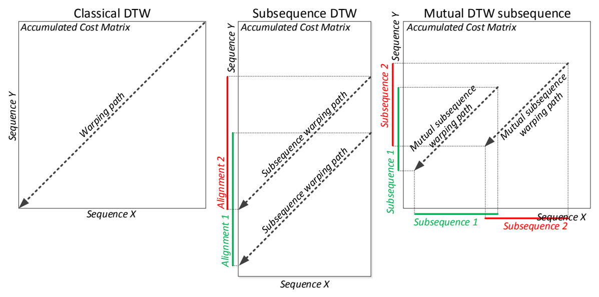

Figure 11: Philosophy of Searching the Warping Path

Visualized cost matrix in Fig. 12 suggests the optimal way for searching the warping paths. There are two important tasks to solve: Where exactly is the best to start the searching of the warping paths and how to adapt the computation of accumulated cost matrix for searching mutual subsequences.

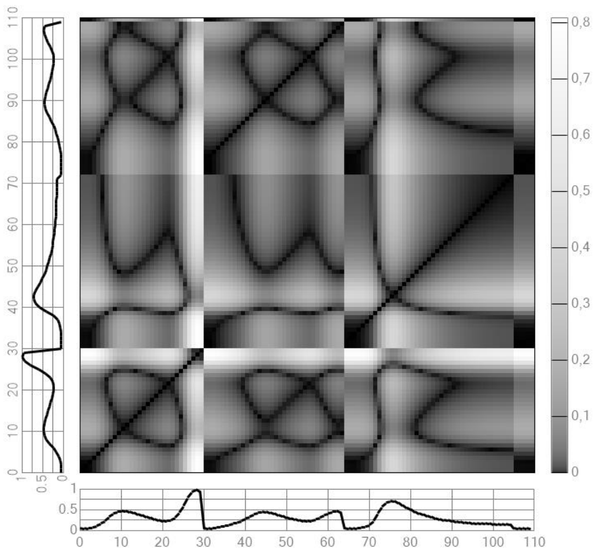
Figure 12: Visualized Cost Matrix

4.1. Determining the starting points

Unlike the DTW or Subsequence DTW, there is no clearly defined point (or set of points) where to start searching the warping path. In this case, it is necessary to pick one or more matrix elements that are interesting in some manner. Test showed that local minima can be utilized as potential path starts. As the local minima such elements of the matrix are considered, whose values are less or equal than the value of surrounding values (expect those in diagonal way). In another words, the element is considered as a local minimum if its value is lower or equal than the neighboring item above, below, right, left, below right and above left. Items below right and above left are ignored, because these elements are parallel to diagonal and lie on a potential same path. Formally, the set of minima M is defined as:

$$\begin{aligned} M = \{(i,j) \in \mathbb{N}^2 \mid\, & i < m \wedge j < n \wedge \\ & \wedge C(i,j) < C(i-1,j+1) \\ & \wedge C(i,j) < C(i-1,j) \\ & \wedge C(i,j) < C(i,j-1) \\ & \wedge C(i,j) < C(i,j+1) \\ & \wedge C(i,j) < C(i+1,j-1) \\ & \wedge C(i,j) < C(i+1,j)\} \end{aligned}$$

Fig. 13 shows that minima (red dots), found by this way for sequences from Fig. 10, exactly highlight the center of the dark areas and suggest the meaningful points.

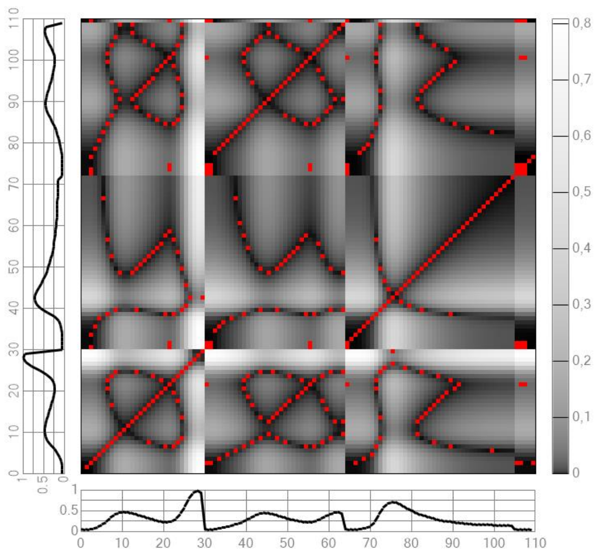

Figure 13: Visualized Cost Matrix Containing Minima

4.2. Computation of accumulated matrix

Once the potential paths' beginnings are found, searching the mutual subsequences may start. However, a standard accumulated matrix cannot be used, because the values are accumulated from the bottom left corner. Because of that, also constructed warping paths are "attracted" to the bottom left corner. It is mostly evident on the left and bottom side of the matrix in Fig. 14, where the color goes light instead of getting dark.

An original subsequence DTW allows searching the best alignment of a sample within the signal database by not penalizing omissions alignment at the beginning and end of the longer sequence by the following setting:

$D(1,s) = c(x_1, y_s)$ for $s \in \{1,\ldots,m\}$.

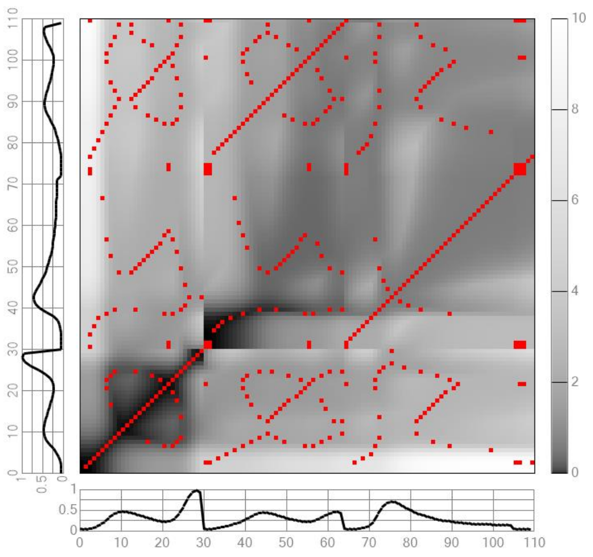

Figure 14: Accumulated Cost Matrix

Our approach additionally does not penalize even omissions of the alignment at the beginning and at the end of the shorter sequence by setting:

$D(r,1) = c(x_r, y_1)$ for $r \in \{1,\ldots,n\}$.

Accumulated cost matrix in Fig. 15 shows that subsequences located along borders are now clearly visible. However, subsequences located in the middle or at the ends of the sequences are still hidden. Moreover, the visualization suggests that warping will be "attracted" to the borders of a matrix, not diagonally as it is wanted.

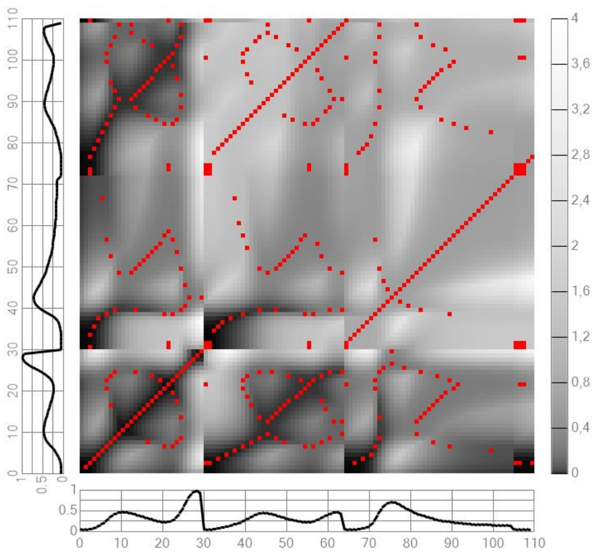

Figure 15: Modified Accumulated Cost Matrix

To solve this situation, the set of minima can be utilized again. We supplemented the computation of the accumulated cost matrix by zeroing the items located in minima.

$$D(r,s) = \begin{cases} 0 & if\ (r,s) \in M \\ min \begin{Bmatrix} D(n-1,m-1), \\ D(n-1,m), \\ D(n,m-1) \end{Bmatrix} + c(x_r, y_s) & otherwise \end{cases}$$

for $r \in \{2, \dots, n\}$ and $s \in \{2, \dots, m\}$.

This zeroing causes resets of accumulation in minima points and changing the direction of warping path to the closest minimum. This effect is mostly evident in Fig. 16 on the right middle side.

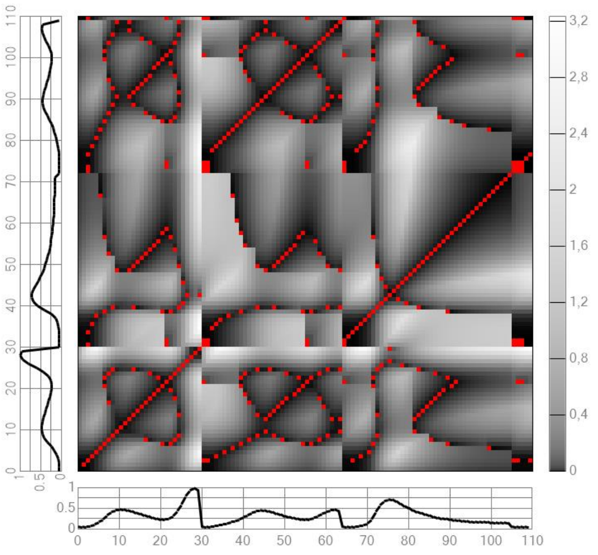

Figure 16: Modified Accumulated Cost Matrix with Zeroed Minima

4.3. Searching the subsequent warping path

Searching the mutual subsequence warping path is performed in the same way as in classical DTW. The only difference is in selecting the local minima as starting points, as it was defined in paragraph 4.1.

The local minima are taken one after another (from the top right corner of a matrix) and from the corresponding matrix element a new warping path is constructed. For a faster run of this algorithm, each of the minima does not have to be examined. A minimum contained in any of the already existing warping paths can be skipped.

Each of such warping paths is constructed with no restrictions, i.e. it is constructed until it touches the bottom or the left side of the matrix, regardless the thresholds defined in Sec. 4. It is done because it is not possible to evaluate some of the thresholds while constructing the warping path, because the overall view onto it is missing. It is evident especially for the maximal average cost t_a, where evaluating the threshold during the construction of a warping path may cause its premature end. This will be demonstrated in the following simple example.

Imagine the situation we want to find a subsequence with a maximal average cost $t_a = 0.2$ and the found warping path points' costs are:

$$c_s = \left(c\left(x_{p_{5x}}, y_{p_{5y}}\right), c\left(x_{p_{4x}}, y_{p_{4y}}\right), \dots, c\left(x_{p_{1x}}, y_{p_{1y}}\right)\right) = (0,0,1,0,0)$$

If the average cost will be computed and evaluated point by point, the construction of the path will end just after the third point, because:

$$c_{s_{1:3}} = \frac{0+0+1}{3} = 0.\overline{3} > t_a = 0.2$$

However, this ending of the construction is unnecessarily, because the average costs of the whole warping points is:

$$c_{s_{1:5}} = \frac{0+0+1+0+0}{5} = 0.2 \leq t_a = 0.2,$$

The interruption was caused by a single greater value, which does not have important meaning in the global view. To avoid this inaccuracies, the warping paths with no limitations are constructed first and then the final mutual subsequences exactly respecting the adjusted thresholds are extracted.

4.4. Extraction of mutual subsequences

In general, once the unlimited paths in 4.3 are found, the goal is to extract set of all subsequences $S = \{s | s = p_{u:v}\}$ for each warping path $p = (p_1, \dots, p_q)$, where following conditions are satisfied:

- $c_t(s) \leq t_t$,
- $c_a(s) \leq t_a$
- $c_e(s) \leq t_e$,
- found sequence $s = p_{a:b}$ is not a subsequence of any other found sequence $u = p_{c:d}$ also satisfying defined conditions, where $1 \leq c \leq a \leq b \leq d \leq q$.

The t_t, t_a, t_e are thresholds defined in Sec. 4 and the $c_t(s)$, $c_a(s)$ and $c_e(s)$ for a subsequence s are costs defined as follows:

- Total cost $c_t(s) = \sum_{k=u}^{v} c(x_{p_{kx}}, y_{p_{ky}})$,
- average total cost $c_a(s) = \frac{c_t(s)}{v-u+1}$,
- maximal item cost
$$c_e(s) = \max_{k \in \{u,\dots,v\}} \left\{c(x_{p_{kx}}, y_{p_{ky}})\right\}$$

To ensure extracting all the mutual subsequences and to avoid unnecessary shortening of them, the all combinations have to be examined. For this purpose, a two triangular matrices are constructed – matrix of total costs and a matrix of average costs. The partial elements of the matrices symbolize computed sums of costs (respectively averages of costs) between the i-th and j-th item of a warping path. Formally:

$C_t \in \mathbb{R}^{n\times m}$ defined by $C_{t_{ij}} = \sum_{k=i}^{j} c(x_{p_{kx}}, y_{p_{ky}})$.

$C_a \in \mathbb{R}^{n\times m}$ defined by $C_{a_{ij}} = \frac{\sum_{k=i}^{j} c(x_{p_{kx}}, y_{p_{ky}})}{j-i+1}$.

From these matrices, a sets of pairs describing the start and the end of mutual subsequences can be derived:

$$B_t = \{(i,j) \in \mathbb{N}^2 \mid C_t(i,j) \le t_t\}$$
$$B_a = \{(i,j) \in \mathbb{N}^2 \mid C_t(i,j) \le t_a\}$$

To receive all mutual subsequences respecting both maximal cost threshold and maximal average cost threshold, the intersection of these sets have to be defined as:

$$B_s = B_t \cap B_a$$

Unfortunately, an appropriate matrix for maximal item cost cannot be built, so we used the set of forbidden indexes which cannot be included in any mutual subsequence. At these indexes, the costs in a warping path are larger than the threshold t_e:

$$I_f = \left\{ i \in \mathbb{N} \mid c\left(x_{p_{ix}}, y_{p_{iy}}\right) > t_e \right\}$$

The final set of pairs respecting all the thresholds are:

$$B_f = \left\{(i,j) \in B_s \middle| \neg\exists k \in I_f \; [i \le k \le j]\right\}$$

However, this pairs include all the possible mutual subsequences including also the subsequences of subsequences. In another words, this set can contain a pair $(0,10)$ as well as its subsequence $(2,8)$. The $(2,8)$ have to be ignored, because the goal is to find as long mutual subsequence as it is possible.

Omissions of such subsequences of subsequences can be easily done by ignoring all pairs having their begins and ends inside of another pair:

$$B = \left\{(i,j) \in B_f \mid \neg\exists (k,l) \in B_f [k \le i \wedge l \ge j]\right\}$$

The final pairs $(i,j) \in B$ specify the indexes of the start and the end warping points $p_{i:j}$ enclosing the warping points of a mutual common subsequence. The values of mutual subsequences are there defined as:

$$x_s = (x_{p_{ix}}, x_{p_{(i+1)x}}, \dots, x_{p_{jx}})$$
$$y_s = (y_{p_{iy}}, y_{p_{(i+1)y}}, \dots, y_{p_{jy}})$$

An example of such found mutual subsequences for the sequences from Fig. 10 can be found in Fig. 17. The corresponding warping paths are also visualized in the cost matrix in Fig. 18.

However, this example is presented only for a demonstration purposes, because the processed sequences are artificially constructed from the identical sequences and they are not a subject of the real distortion. Visualization of many sequences with corresponding combinations of algorithm's parameters, unfortunately, exceeds the scope of this paper.

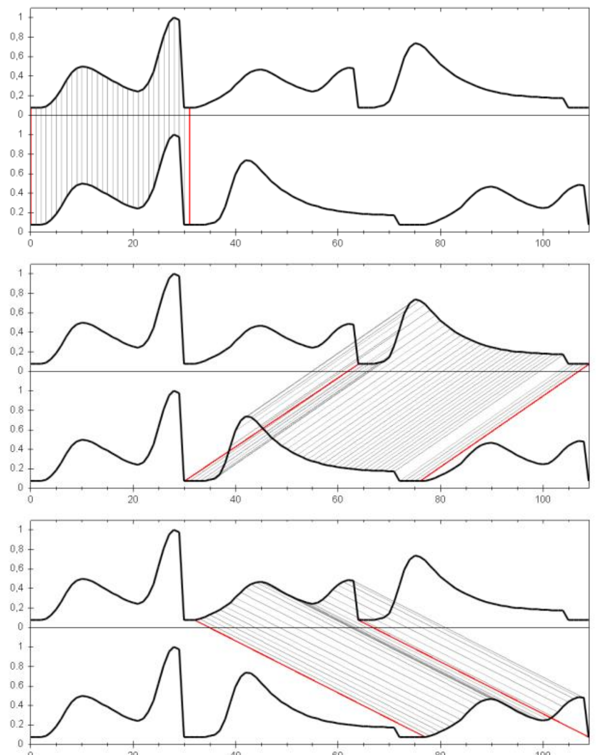

Figure 17: Found Mutual Subsequences

CONCLUSION

Tests showed that the proposed algorithm is able to find mutual subsequences in distorted time series. Since the algorithm is mostly based on DTW and mutually independent parts, it is very easy to parallelize the computations and rapidly speed it up. Future work will be focused on such optimization and automatic adjustment of algorithm's parameters.

ACKNOWLEDGEMENT

This work was supported by the European Regional Development Fund in the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070) and the national budget of the Czech Republic via the Research and Development for Innovations Operational Programme, by the project New creative teams in priorities of scientific research (reg. no. CZ.1.07/2.3.00/30.0055), supported by Operational Programme Education for Competitiveness and by the internal grant agency of VŠB - Technical University of Ostrava, Czech Republic, under the project no. SP2015/157 'HPC Usage for Transport Optimisation based on Dynamic Routing'.

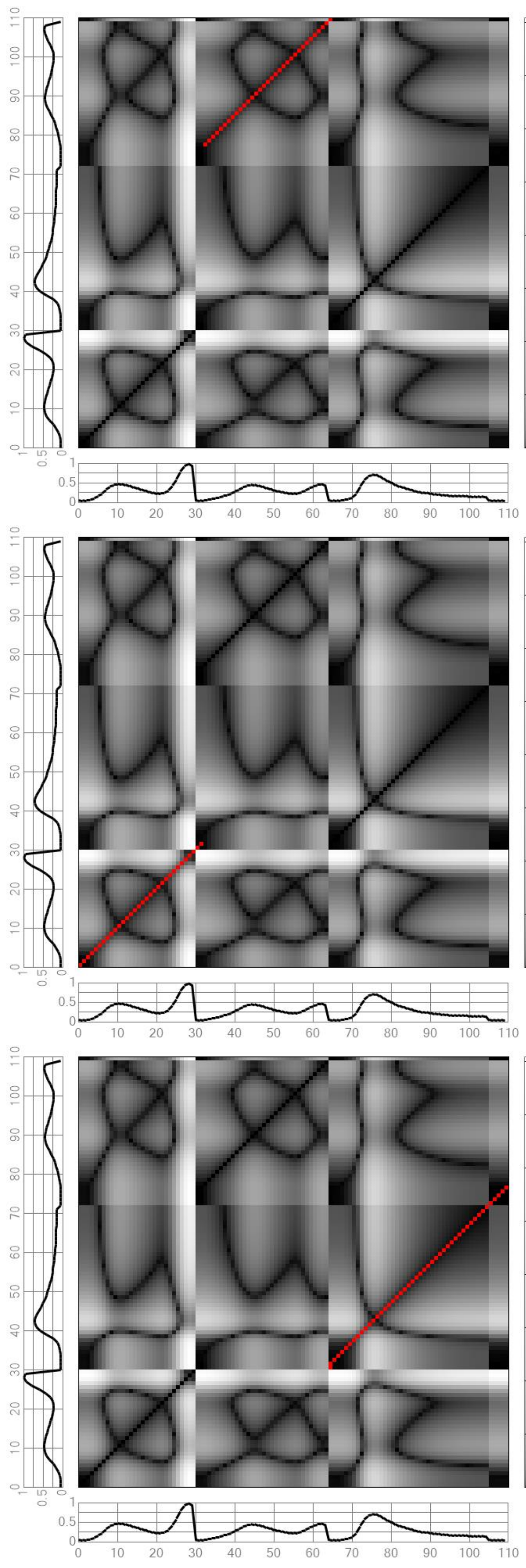
Figure 18: Found Warping Paths

REFERENCES

Cohen, P. R., Adams, N., and Heeringa, B, 2007. Voting Experts: An Unsupervised Algorithm for Segmenting Sequences. *In Journal of Intelligent Data Analysis*.

Chen, J., Huang, K., Wang, F., and Wang, H., 2009. E-learning Behavior Analysis Based on Fuzzy Clustering. *In Proceedings of International Conference on Genetic and Evolutionary Computing*.

Esling, P., Agon C., 2012. Time-series data mining. *ACM Computing Surveys*, Volume 45, Issue 1, November 2012, Article number 12.

Goldin, D., Kanellakis, P., 1995. On similarity queries for time-series data: Constraint specification and implementation. *1st International Conference on the Principles and Practice of Constraint Programming*, pages 137–153, France.

Guo, A., Siegelmann, H., 2004. Time-Warped Longest Common Subsequence Algorithm for Music Retrieval. *In Proceedings of 5th International Conference on Music.*

Information Retrieval ISMIR 2004, pages 258–261. Universitat Pompeu Fabra, 2004.

Gusfield, D., 2008. Algorithms on Strings, Trees and Sequences, *Computer Science and Computational Biology*. Cambridge University Press.

Hirschberg, D. S., 1977. Algorithms for the Longest Common Subsequence Problem. *ACM, 24:664–675*.

Keogh, E., Pazzani, M., 2000. Scaling up dynamic time warping for datamining applications. *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. Pages 285-289

Kocyan, T., Martinovič, J., Dráždilová, P., and Slaninová K., 2013. Recognizing Characteristic Patterns In Distorted Data Collections. *Proceedings of The European Modeling and Simulation 2013*.

Kocyan, T., Martinovič, J., and Podhoráҧyi, M., 2014. Searching and indexing distorted data collections. *Proceedings of The European Modeling and Simulation 2014*.

Lin, J., Keogh, E., Wei, L., Lonardi, S., 2007. Experiencing SAX: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, v.15 n.2, p.107-144, October 2007.

Miller M., Wong P., and Stoytchev A., 2009. Unsupervised Segmentation of Audio Speech Using the Voting Experts Algorithm. *Proceedings of the Second Conference on Artificial General Intelligence (AGI)*.

Muller, M., 2007. Dynamic Time Warping. *Information Retrieval for Music and Motion*, Springer, ISBN 978-3-540-74047-6, 69--84.

Rabiner, L., Biing-Hwang Juang, 1993. *Fundamentals of Speech Recognition,* Prentice Hall, ISBN: 0130151572.