Scaling and compressing melodies using geometric similarity measures

L. E. Caraballo[1], J.M. Díaz-Báñez[1], F. Rodríguez[1], V. Sánchez-Canales[1], and I. Ventura[1]

1 University of Seville
`lcaraballo@us.es,dbanez@us.es,fabrodsan@us.es,vscanales@us.es,iventura@us.es`

Abstract

Melodic similarity measurement is of key importance in Music Information Retrieval. In this paper, we use geometric matching techniques to measure the similarity between two monophonic melodies. We propose efficient algorithms for optimization problems inspired in two operations on melodies: scaling and compressing. In the scaling problem, an incoming query melody is scaled forward until the similarity measure between the query and the reference melody is minimized. The compressing problem asks for a subset of notes of a given melody so that the matching cost between the selected notes and the reference melody is minimized.

1 Introduction

Musicological and computational studies on rhythmic and melodic similarity have given rise to a number of geometric problems [6]. A melody can be codified as a consecutive sequence of musical notes and each note can be represented by a point (point-representation) [3] or a horizontal segment (segment-representation) in a time pitch value [5]. In this paper we study two problems that arise in Musical Information Retrieval (MIR): scaling and melody compressing. Scaling is used for tempo variation [4] and compression for clustering [2]. Given a reference melody, a query melody and a similarity measure, in the *scaling problem* the incoming query is scaled forward in the horizontal direction to find the minimum similarity measure between it and the reference melody. The *compressing problem* of a given melody asks to select k notes of such melody so that the similarity measure between the reference and the simplified melody is minimized. The study of measures for melodic similarity is of key importance for a music retrieval system. Techniques based on geometric similarity measures have been used in the literature. In [1], the following geometric matching technique for music similarity measurement was proposed: Each note is represented as a horizontal line segment so that a sequence of notes can be described as a rectangular contour in a 2D coordinate system, in which the horizontal and vertical coordinates correspond to time and pitch value, respectively. Then, the used similarity measure between two melodies is the minimum area between them. They solve the problem of searching a query melodic segment of length m into a reference melody of length n in $O(nm \log n)$ time. The optimal matching is obtained by moving from left to right and bottom to up the query segment until the matched area is minimized.

This research has been supported by the project GALGO (Spanish Ministry of Economy and Competitiveness, MTM2016-76272-R AEI/FEDER,UE) and the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 734922.

36th European Workshop on Computational Geometry, Würzburg, Germany, March 16–18, 2020.
This is an extended abstract of a presentation given at EuroCG'20. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

1.1 Problems statement

Let $R = (R_1, R_2, ..., R_n)$ and $Q = (Q_1, Q_2, ..., Q_m)$ be sequences representing two melodies with $m < n$. R is the reference melody from the data set and Q is the query melody to be matched. In the point-representation, we assume that the points representing the notes are just the middle points of the horizontal segments in the corresponding segment-representation. We also call melodic contour to a segment-representation.

In the following, we introduce two geometric measures to compute similarity and two operations on a melody that are the main ingredients of the problems.

Area: The region between two melodies with the same duration in the segment-representation can be partitioned into rectangles with vertical edges supported by vertical straight lines passing through the ending points of the segments. The area between two melodies is defined as the sum of the areas of the rectangular regions of the partition. See Figure 1a).

t-Monotone Matching: Let $R = \{R_i = (x_i, p_i), i = 1, \cdots, n\}$ and $Q = \{Q_j = (t_j, q_j), j = 1, \cdots, m\}$ be two melodies within the point-representation. In a t-monotone matching between R and Q, we map each point of R to its nearest t-coordinate point in Q, that is, the left- or the right-point in time. The unmatched points of Q are associated to their t-coordinate nearest in R. See Figure 1b). Note that this matching is designed for music matching and does not use the Euclidean distance. The cost of the note Q_j, $\phi(Q_j)$, is given by the sum of the l_1-distances between Q_j and its matched points in R and the total cost of the matching is

$$\phi(R, Q) = \sum_{Q_j \in Q} \phi(Q_j).$$

ε-scaling: Consider a segment-representation of two melodies R and Q. Let $X = (x_0 = 0, x_1, x_2, ..., x_n)$ and $T = (t_0 = 0, t_1, t_2, ..., t_m)$ be the partitions on time given by the R and Q, respectively. Given $\varepsilon > 0$, we define the *ε-scaling* on the query Q, $S_\varepsilon(Q)$, as the operation of increasing by ε the length of each segment of Q but keeping static the starting point of Q. Thus, after an ε-scaling, X does not change and T is transformed to $T + \varepsilon = (t_0, t_1 + \varepsilon, t_2 + 2\varepsilon, t_3 + 3\varepsilon, \cdots, t_m + m\varepsilon)$. The query can be scaled forward until the two melodies have the same time duration. Thus, $0 \leq \varepsilon \leq \frac{x_n - t_m}{m}$. Note that the pitch of the notes are unchanged in the scaling operation. Now, for a point-representation of R and Q, $R = \{(x_i, p_i), i = 1, \cdots, n\}$ and $Q = \{(t_j, q_j), j = 1, \cdots, m\}$, the *$\varepsilon$-scaling* of Q is given by $S_\varepsilon(Q) = \{(t_1 + \frac{\varepsilon}{2}, q_1), (t_2 + \varepsilon, q_2), \cdots, (t_m + \frac{m}{2}\varepsilon, q_m))\}$.

k-compressed melody: Given a melody R with n notes in the segment-representation, a k-compressed melody, Q_k, is a melody composed by k segments such that Q_k and R have the same duration and each segment of Q_k contains at least a segment of R. Figure 2b). For the point-representation, a k-compressed melody Q_k is a subset of k notes of R. Figure 2a).

In this paper we study the following optimization problems:

▶ Problem 1.1. Given two melodies R and Q in the segment-representation, compute the value of $\varepsilon > 0$ such that the area between R and $S_\varepsilon(Q)$ is minimized.

▶ Problem 1.2. Given two melodies R and Q in the point-representation, compute the value of $\varepsilon > 0$ such that the cost of the t-monotone matching between R and $S_\varepsilon(Q)$ is minimized.

▶ Problem 1.3. Given a melody R in the point-representation, compute a k-compressed melody Q_k so that the cost of the t-monotone matching between R and Q_k is minimized.

▶ Problem 1.4. Given a melody R with n notes in the segment-representation, compute a k-compressed melody Q_k so that the area between R and Q_k is minimized.

Note that in the scaling problems, the reference melody is fixed whereas the query melody is dynamic. However, the compressing problems ask for an optimal selection of the notes in the reference melody. Figures 1 and 2 illustrate the problems.

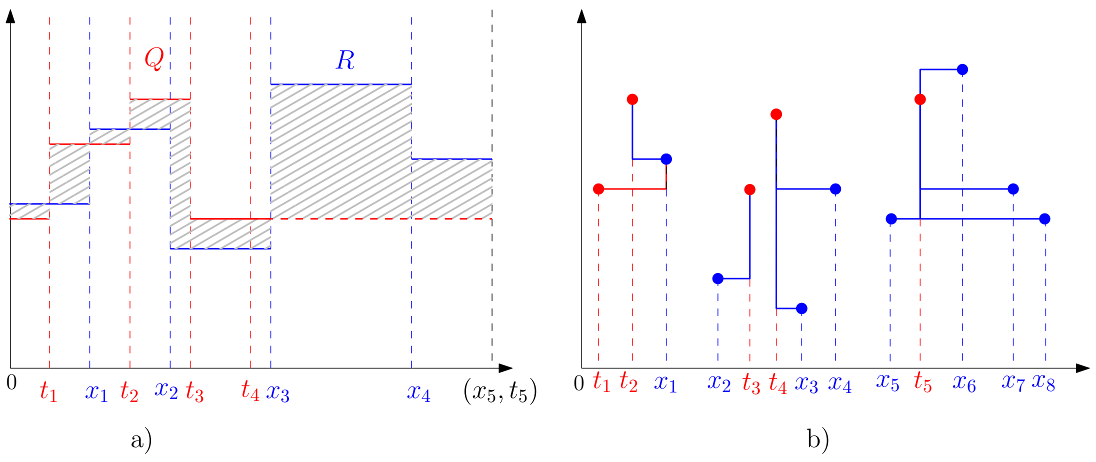

Figure 1 a) Area between the reference R and the query Q, Problem 1.1. b) t-Monotone matching between the reference R (blue) and the query Q (red), Problem 1.2.

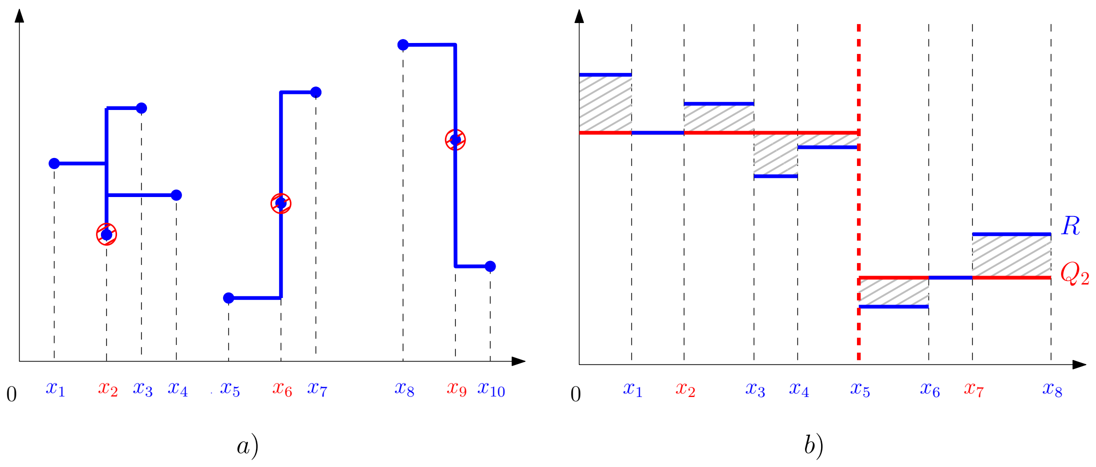

Figure 2 a) A 3-compressed melody from an input of 10 notes, Problem 1.3. b) Area between a melody R and a 2-compressed melody Q_2, Problem 1.4.

2 Scaling

2.1 Area as similarity measure

We assume that the last segment of the scaled query $S_\varepsilon(Q)$ is extended so that the duration of R and $S_\varepsilon(Q)$ is the same. Thus, the area between the melody R and $S_\varepsilon(Q)$ is the sum of $O(m+n)$ rectangles as illustrated in Fig 1a). Denote by $A_{RQ}(\varepsilon)$ the area between R and $S_\varepsilon(Q)$ as a function of ε. Observe that after an ε-scaling of the query for a big enough ε, at least one pair of vertical edges coincide, that is, $x_i = t_j$ for some i, j. At this instant, a rectangle disappears and after that, a new rectangle appears. We call this value of ε an *event*.

Also note that between two events, the area of some rectangles increase, others decrease and others are unaffected. In fact, the type of a rectangle can be determined by its vertical edges. Rectangles can be classified in four types: Type C_0, with vertical edges $[x_i, x_{i+1}]$; type C_1, with vertical edges $[t_j, t_{j+1}]$; type C_2, with vertical edges $[x_i, t_j]$ and type C_3, with vertical edges $[t_j, x_i]$. Figure 3 illustrates an event in which a rectangle of type C_3 disappears.

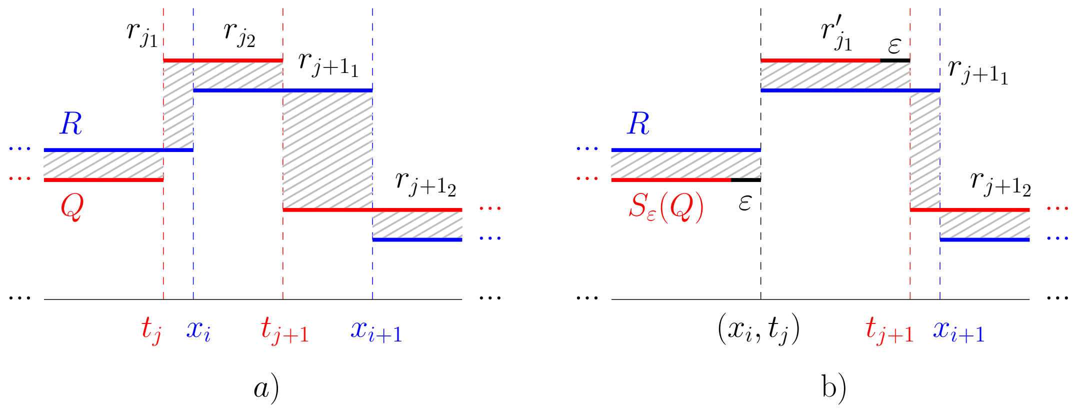

Figure 3 a) Rectangles between R and Q. b) After the scaling $S_\varepsilon(Q)$, an event is found when t_j touches x_i. Rectangle r_{j_1} disappears and r_{j_2} is renamed as r'_{j_1}.

The following result allows us to discretize the problem:

▶ **Lemma 2.1.** *$A_{RQ}(\varepsilon)$ is a piecewise linear function.*

As a consequence of Lemma 2.1, the minimum area is reached at the events and our approach is to efficiently reevaluate the area in each event and take the minimum value.

▶ **Lemma 2.2.** *$A_{RQ}(\varepsilon)$ can be updated between two consecutive events in O(1) time.*

▶ **Theorem 2.3.** *Problem 1.1 can be solved in $O(nm \log m)$ time.*

2.2 t-Monotone matching as similarity measure

Let $R = \{(x_i, p_i), i = 1, \cdots, n\}$ and $Q = \{(t_j, q_j), j = 1, \cdots, m\}$ be two melodies in the point-representation. The t-Monotone matching between R and Q generates two sets of pairs of points. Let P_1(resp. P_2) be the set of pairs such that $x_i \leq t_j$ (resp. $x_i > t_j$). Observe that for a small $\varepsilon > 0$, after the scaling $S_\varepsilon(Q)$ of Q, the costs related to pairs in P_1 (resp. P_2) increase (decrease). The sets P_1 and P_2 are dynamics in the scaling operation, that is, t_j is moving from left to right but x_i is static. For the sake of simplicity, for any ε in the scaling process, we use (x_i, t_j) or (i, j) to refer the pair with abscissas x_i and t_j. We call events to the values of $\varepsilon > 0$ where the pair (i, j) can change. We have three type of events:

1. The instant at which $x_i = t_j$: the pair (i, j) changes from P_2 to P_1.
2. The instant at which the bisector of t_j and t_{j+1} passes through x_i: the pair $(i, j+1)$ in P_2 becomes the pair (i, j) in P_1.
3. The instant at which t_j passes trough the bisector of x_i and x_{i+1}: the pair (i, j) in P_1 becomes the pair $(i+1, j)$ in P_2.

▶ **Lemma 2.4.** *The cost $\phi(R, S_\varepsilon(Q))$ is a linear function between two consecutive events.*

▶ **Lemma 2.5.** *The cost $\phi(R, S_\varepsilon(Q))$ can be updated between two consecutive events in $O(1)$ time.*

Using Lemma 2.4 and Lemma 2.5, the problem can be solved by efficiently updating the local optimum values reached at the events and we can prove:

▶ **Theorem 2.6.** *Problem 1.2 can be solved in $O(nm\log(n+m))$ time.*

3 Compressing

3.1 t-Monotone matching as similarity measure

Let $R = \{R_i = (x_i, p_i), i = 1, \cdots, n\}$ be a sequence of notes according the point-representation and $k \in \{1, \cdots, n\}$. Let $C_k = \{c_1, \ldots, c_k\} \subseteq R$ be a k-set of R. Consider a t-monotone matching between R and C_k and set $\phi(R, C_k) = \sum_{c_i \in C_k} \phi(c_i)$.

▶ **Definition 3.1.** Let $C_k = \{c_1, \ldots, c_k\}$ be a k-set of R.

1. Set $\overleftarrow{R} = \{R_i \in R : x(R_i) \leq x(c_k)\}$. The *left partial evaluation* of C_k is defined as:
$$\overleftarrow{\phi}(R, C_k) = \phi(\overleftarrow{R}, C_k).$$
2. We say that C_k is *left optimal* if $\overleftarrow{\phi}(C_k) \leq \overleftarrow{\phi}(C'_k)$ for all $C'_k = \{c'_1, \ldots, c'_k\}$ where $c'_k = c_k$.
3. The j-prefix of C_k, is the j-set formed by the first j points of C_k.

The following result can be easily proven by contradiction:

▶ **Lemma 3.2.** *Let $C^*_k = \{c^*_1, \ldots, c^*_k\}$ be an optimal k-set of S. Then, for all $1 \leq j \leq k$ the j-prefix of C^*_k is left optimal.*

▶ **Corollary 3.3.** *Let $C_j = \{c_1, \ldots, c_j\}$ be a j-set which is left optimal. Then every prefix of C_j is also left optimal.*

Above properties allow us to solve the problem with dynamic programming. A sketch of the idea is as follows. Let p_i and $p_{i'}$ be two consecutive points in a k-set. Let $W_{ii'}$ be the cost of the t-monotone matching for points between p_i and $p_{i'}$. Then, for a k-set $C_k = \{p_{i_1}, \ldots, p_{i_k}\}$, where $1 \leq i_j \leq k$ and $i_j < i_{j+1}$, the total cost $\phi(C_k)$ can be rewritten as $W_{0,i_1} + W_{i_1,i_2} + \cdots + W_{i_k,(n+1)}$, where W_{0,i_1} (resp. $W_{i_k,(n+1)}$) denotes the assignment cost of the points to the left (resp. right) of p_{i_1} (resp. p_{i_k}).

Now, assume that we have preprocessed the values $W_{ii'}$, for all $0 \leq i < i' \leq n+1$. Consider the tables $C[i,j]$ and $P[i,j]$ whose keys i and j are integers in the intervals $[1,n]$ and $[1,k]$. The cell $C[i,j]$ stores the cost of the left optimal j-set C_j that ends using p_i as the j-th point of the subset. The cell $P[i,j]$ stores the index $i' < i$ of the $(j-1)$-th point of C_j.

▶ **Theorem 3.4.** *Assuming that the values $W_{i',i}$ are already known, the optimum k-set can be computed in $O(kn^2)$ time.*

▶ **Lemma 3.5.** *We can compute all the values $W_{i',i}$ in $O(n^2)$ time.*

▶ **Theorem 3.6.** *Problem 1.3 can be solved in $O(kn^2)$ time.*

3.2 Area as similarity measure

Given a melodic contour R with n notes, we want to find a k-compression Q_k of R that minimizes the area between R and Q_k. Let $X = (x_0 = 0, \ldots, x_n)$ and $T = (t_0 = 0, \ldots, t_k)$ be the partitions on the time interval $[0, x_n = t_k]$ of R and Q_k, respectively. Let $P = (p_1, \ldots, p_\ell)$, $\ell \leq n$ be the set of different pitch values among the notes of R.

▶ **Lemma 3.7.** *Let R be a melodic contour with time partition $X = (x_0 = 0, \ldots, x_n)$. Let $0 < k \leq n$ be a natural number. There exists a melody contour Q_k^* of k notes and time partition $T = \{t_0 = 0, \ldots, t_k = x_n\}$ that minimizes the area difference with R overall all the melodic contours of k notes starting at time x_0 and ending at time x_n and fulfills the following two properties:*

1. *$T \subseteq X$ and,*
2. *each note of Q_k^* contains at least a note of R.*

Recall that the notes of a k-compression of R contain notes of R. The previous lemma says that there exists an optimal melodic contour with k notes that is a k-compression.

Let R be a melodic contour with time partition $X = \{x_0, \ldots, x_n\}$. For every $x_0 \leq t \leq x_n$, we denote by $R_{\overleftarrow{t}}$ the *prefix melody* of R with time partition $X_{\overleftarrow{t}} = \{x_0, \ldots, x_i, t\}$, where $x_i < t \leq x_{i+1}$. The following result establishes an optimal substructure of a solution.

▶ **Lemma 3.8.** *Given three values $1 \leq j \leq k$, $j \leq i \leq n$ and $p \in P$, denote by $S_p(i, j)$ the set of all the melodies formed by j notes starting at time x_0 and ending at time x_i whose last note μ has pitch $p \in P$ and starts at some time $x \in X$, $x < x_i$. Then, there exists a melody $C \in S_p(i, j)$ that minimizes the area difference with $R_{\overleftarrow{x_i}}$ over all the melodies in $S_p(i, j)$ such that $C_{\overleftarrow{x_{i'}}}$ is an optimum $(j-1)$-compression melody of $R_{\overleftarrow{x_{i'}}}$, where $x_{i'}$ is the starting time of the last note of C.*

Based on the previous result, dynamic programming can be used and we can prove:

▶ **Theorem 3.9.** *Problem 1.4 can be solved in $O(k\ell n)$ time, where ℓ is the number of different pitch values of R.*

References

1 G. Aloupis, T. Fevens, S. Langerman, T. Matsui, A. Mesa, Y. Nuñez, D. Rappaport, and G. Toussaint. Algorithms for computing geometric measures of melodic similarity. *Computer Music Journal*, 30(3):67–76, 2006.

2 R. Cilibrasi, P. Vitányi, and R. de Wolf. Algorithmic clustering of music based on string compression. *Computer Music Journal*, 28(4):49–67, 2004.

3 R. Clifford, M. Christodoulakis, T. Crawford, D. Meredith, and G. A. Wiggins. A fast, randomised, maximal subset matching algorithm for document-level music retrieval. In *ISMIR*, pages 150–155, 2006.

4 J.-S. Roger Jang, H.-R. Lee, and Ming-Yang K. Content-based music retrieval using linear scaling and branch-and-bound tree search. In *IEEE International Conference on Multimedia and Expo, 2001. ICME 2001.*, pages 74–74, 2001.

5 D. O. Maidín. A geometrical algorithm for melodic difference. *Computing in musicology: a directory of research*, (11):65–72, 1998.

6 G. Toussaint. Computational geometric aspects of rhythm, melody, and voice-leading. *Computational Geometry*, 43(1):2–22, 2010.