Segmentation of Trajectories on Nonmonotone Criteria

BORIS ARONOV, Polytechnic Institute of NYU
ANNE DRIEMEL, TU Eindhoven
MARC VAN KREVELD, MAARTEN LÖFFLER, and FRANK STAALS, Utrecht University

In the trajectory segmentation problem, we are given a polygonal trajectory with n vertices that we have to subdivide into a minimum number of disjoint segments (subtrajectories) that all satisfy a given criterion. The problem is known to be solvable efficiently for *monotone* criteria: criteria with the property that if they hold on a certain segment, they also hold on every subsegment of that segment. To the best of our knowledge, no theoretical results are known for nonmonotone criteria.

We present a broader study of the segmentation problem, and suggest a general framework for solving it, based on the *start-stop diagram*: a 2-dimensional diagram that represents all valid and invalid segments of a given trajectory. This yields two subproblems: (1) computing the start-stop diagram, and (2) finding the optimal segmentation for a given diagram. We show that (2) is NP-hard in general. However, we identify properties of the start-stop diagram that make the problem tractable and give a polynomial-time algorithm for this case.

We study two concrete nonmonotone criteria that arise in practical applications in more detail. Both are based on a given univariate attribute function f over the domain of the trajectory. We say a segment satisfies an *outlier-tolerant criterion* if the value of f lies within a certain range for at least a given percentage of the length of the segment. We say a segment satisfies a *standard deviation criterion* if the standard deviation of f over the length of the segment lies below a given threshold. We show that both criteria satisfy the properties that make the segmentation problem tractable. In particular, we compute an optimal segmentation of a trajectory based on the outlier-tolerant criterion in $O(n^2 \log n + kn^2)$ time and on the standard deviation criterion in $O(kn^2)$ time, where n is the number of vertices of the input trajectory and k is the number of segments in an optimal solution.

Categories and Subject Descriptors: F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Geometrical problems and computations*

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Trajectory, segmentation, geometric algorithms, dynamic programming

Work on this article has been partially supported by the Netherlands Organisation for Scientific Research (NWO) under project no. 639.021.123, 612.001.022, and 612.065.823, and by EU Cost Action IC0903 (MOVE). Work on this article by B.A. has been partially supported by NSA MSP Grant H98230-10-1-0210 and by NSF Grants CCF 08-30691 and CCF 11-17336.
Authors' addresses: B. Aronov, Department of Computer Science and Engineering, Tandon School of Engineering, New York University, New York, NY 11201 USA; email: boris.aronov@nyu.edu; A. Driemel, Department of Mathematics and Computer Science, TU Eindhoven, PO Box 513, 5600 MB Eindhoven, The Netherlands; email: a.driemel@tue.nl; M. van Kreveld and M. Löffler, Utrecht University, Department of Information and Computing Sciences, PO Box 80.089, 3508 TB Utrecht, The Netherlands; emails: {m.j.vankreveld, m.loffler}@uu.nl; F. Staals, MADALGO, Center for Massive Data Algorithmics, Department of Computer Science, Aarhus University, The IT-park, Åbogade 34, DK-8200 Aarhus N, Denmark; email: f.staals@cs.au.dk.

DOI: http://dx.doi.org/10.1145/2660772

ACM Reference Format:
Boris Aronov, Anne Driemel, Marc van Kreveld, Maarten Löffler, and Frank Staals. 2015. Segmentation of trajectories on nonmonotone criteria. ACM Trans. Algorithms 12, 2, Article 26 (December 2015), 28 pages.
DOI: http://dx.doi.org/10.1145/2660772

1. INTRODUCTION

Trajectory data is ubiquitous today. Researchers in different fields study the movements of animals [Bovet and Benhamou 1988; Calenge et al. 2009; Gurarie et al. 2009], hurricanes [Stohl 1998], traffic [Li et al. 2010], or other moving objects [Dodge et al. 2009] by analyzing their spatiotemporal trajectories. Movement data is often collected using GPS, and large sets of such data have been collected in the past few years. A trajectory is usually modeled as a continuous function from a time interval (the domain) to the plane, but is collected as a discrete sequence of timestamped locations. By linearly interpolating the locations, we obtain a continuous piecewise-linear curve as the image of the function. We can derive *attributes* from this data: functions defined on the domain of the trajectory, such as speed, heading, or acceleration. In some applications, trajectories may also have additional data stored with each timestamp. Some of these attribute functions, such as *speed* and *heading*, are piecewise constant when we assume linear interpolation.

Segmentation. An important analysis task is to segment a trajectory: partition the input trajectory into a minimum number of *segments* such that each segment satisfies a given criterion [Mann et al. 2002]. A criterion is usually defined based on an attribute of the trajectory. For example, for the attribute "speed," we could segment the trajectory such that the minimum and maximum speed within any segment differ by at most h km/h, or by at most a factor of two. Figure 1 illustrates a segmentation of a trajectory into segments of similar speed. This way, a segment of the trajectory represents a contiguous subtrajectory for which a property is stable in some sense. This aids the automated analysis, since such segments are often meaningful features of the trajectory. In the analysis of the trajectories of birds, for example, the goal is to extract the stretches where a certain activity is observed, such as soaring, directional flight, sleeping, and so forth. For more examples, see Nathan et al. [2008] and van Moorter et al. [2010]. One may draw the comparison to the segmentation of an image into contiguous patches of similar color or texture. In these problems, one is also interested in decomposing the image into meaningful portions, which represent objects shown [Fu and Mui 1981].

Previous research on similar segmentation problems has been done in animal movement studies [Dodge et al. 2009; Nathan et al. 2008] and time-series analysis [Anagnostopoulos et al. 2006; Chundi and Rosenkrantz 2009; Terzi and Tsaparas 2006]; see also Zhou et al. [2011]. The solutions proposed in the cited work provide no guarantees for individual segments in the segmentation. Instead, either they are heuristics or they optimize a global error criterion when a desired number of segments is specified. In the latter case, dynamic programming is a common approach [Mann et al. 2002]. The one exception is the research of Buchin et al. [2011]; see the discussion later.

Monotonicity. A criterion C is *monotone* if it has the property that if C is satisfied on some segment S of a trajectory, then it is also satisfied on any subsegment $S' \subseteq S$. Buchin et al. [2011] provide a framework for segmenting trajectories based on a monotone criterion or a combination of several monotone criteria. Many criteria are monotone by nature. However, if the trajectory data is noisy, or a type of behavior is briefly interrupted, it is desirable to weaken the monotonicity requirement. For example, instead of requiring that the difference in speed within a segment is at most h, we could require that this is the case for at least 95% of the time within each segment.

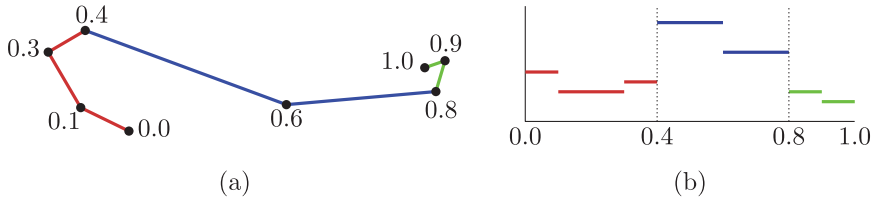


Fig. 1. (a) A trajectory, obtained by linearly interpolating seven locations measured at irregular times. (b) The speed along the trajectory is a piecewise-constant function. We may divide the trajectory into three segments with "similar" speed: [0.0, 0.4], [0.4, 0.8], [0.8, 1.0].

A short burst in speed does not cause more segments in the segmentation with such a criterion. Another example is using a threshold for the standard deviation of speed within a segment.

Our results. To segment a trajectory based on nonmonotone criteria, we introduce the notion of the *start-stop diagram*. This allows us to split the problem into two subproblems: computing the start-stop diagram and computing an optimal segmentation for a given start-stop diagram. We show that the latter problem is NP-hard in general. However, a polynomial-time solution exists if the start-stop diagram has certain properties. We identify these properties in Section 3 and present an algorithm to compute an optimal segmentation. In Section 4, we consider how combining multiple criteria affects the properties of the start-stop diagram. In Section 5, we present lower-bound constructions that show the tightness of some of our upper bounds. Section 6 considers the issue of computing the start-stop diagram; we show how to efficiently compute the start-stop diagram for two specific criteria. One such criterion requires that the minimum and maximum values of a piecewise-constant function f on each segment have at most a given difference while allowing a certain percentage of outliers. The second criterion requires that on each segment the standard deviation of f is below a certain threshold on each segment. We show that these criteria satisfy the properties that make the segmentation problem tractable. Although our framework was designed for trajectory segmentation, our techniques are also applicable to different problems. In particular, in Section 7, we discuss several variants of line simplification and show that some of these fit our framework.

1.1. Discrete Versus Continuous Trajectories

As is often the case in geographical information science, we have a choice how to model the input data. The *true* input is a continuous curve in the plane, which accurately describes the location of a moving entity over time. However, we cannot measure this curve exactly: the best we can do is measure the location of the entity at certain given times, resulting in a sequence of timestamped points in the plane. Two popular ways to approximate the input data are *discrete* and *continuous* trajectories. In the discrete case, we represent the trajectory by the sequence of measured locations directly. In the continuous case, we represent the trajectory by a piecewise linear curve, which should represent the measured data well (at any time of measurement, the corresponding location on the piecewise linear curve should be close to the measured location). Finding a good approximation of a piecewise linear curve is a well-studied problem [Chan and Chin 1992; Douglas and Peucker 1911; Imai and Iri 1988; Ramer 1972].

One advantage of using a discrete representation is that no input data is lost, and because of the often fine and uniform resolution, the resulting data is easier to work with. The main advantage of using a continuous representation is that it allows one to store irregular information more concisely: a part of the trajectory with little changes in direction and speed may be represented by a single edge, while parts of the trajectory

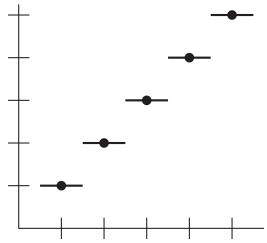


Fig. 2. The representations of the speed along the trajectory in the case of discrete segmentation (points) and continuous segmentation (line segments).

that describe a complex motion can use a finer resolution. As a result, the data can be stored more compactly, and time-intensive algorithms can still be run on the data. One may compare the situation to the differences in using grid-based or triangulation-based representations of terrain (elevation) data.

In the segmentation problem, in the discrete case, we wish to segment the set of input points directly (meaning we are given a discrete set of times that we can subdivide), while in the continuous case, we have the freedom to cut at any time we like. As we will see, the continuous segmentation problem is algorithmically more intricate than the discrete version.

Apart from the fact that we can run continuous segmentation algorithms on smaller input instances, it may also result in a smaller output, even when the input has comparable size. Consider the following example, which shows that comparable situations can give a significantly different number of segments in an optimal segmentation. Consider the discrete function $f(i) = i$ for all $i \in \{1, \ldots, n\}$ and the very similar function $f(x) = \lfloor x + \frac{1}{2} \rfloor$ for $x \in [\frac{1}{2}, n + \frac{1}{2}]$; see Figure 2 for $n = 5$.

Suppose we segment on standard deviation with threshold 0.499. Then the discrete segmentation must take each index separately, because two consecutive indices give a standard deviation of 0.5 and this only increases if we take longer segments. The continuous segmentation can take as the first segment the first constant part with $f(x) = 1$ and nearly all of the second part with $f(x) = 2$. Similarly, the second segment can take the small remaining part where $f(x) = 2$, the entire part where $f(x) = 3$, and nearly all of the part where $f(x) = 4$ (but slightly less than before). The construction scheme leads to nearly half as many segments in the minimal continuous segmentation as in the minimal discrete segmentation.

2. GENERAL APPROACH

We study the problem of segmenting a trajectory with respect to a criterion. In the continuous case, this problem can be defined as follows. We define a trajectory T as a function from the interval $I = [0, 1]$ to $\mathbb{R}^2$ (or $\mathbb{R}^d$) and a subtrajectory, also called *segment*, $T[a, b]$ as the function restricted to the subinterval $[a, b] \subseteq I$. A criterion C is a function $C : I \times I \to \{\textsc{True}, \textsc{False}\}$, which is defined on all possible segments of T. We say an interval $[a, b] \subseteq I$ *satisfies* a criterion C if $C(a, b) = \textsc{True}$; in this case, we call the segment *valid*. A partitioning of I (or of T) into nonoverlapping segments whose union covers I is called a *segmentation*. A segmentation of *size* k can be denoted by its segments $[\tau_0, \tau_1], [\tau_1, \tau_2], \ldots, [\tau_{k-1}, \tau_k]$; $\tau_0 = 0$ and $\tau_k = 1$. A segmentation is *valid* if and only if all of its segments are valid, and *segmenting* a function refers to partitioning into valid segments. We say a valid segmentation is *minimal* (optimal) with respect to C if its size is minimum; the segmentation problem is to compute a valid minimal segmentation. We will often omit the word "valid" because only valid segmentations are useful.

A criterion is often based on an *attribute function* $f : I \to \mathbb{R}$ that has the same domain I as T, for example, the function that maps time to speed at that time. A

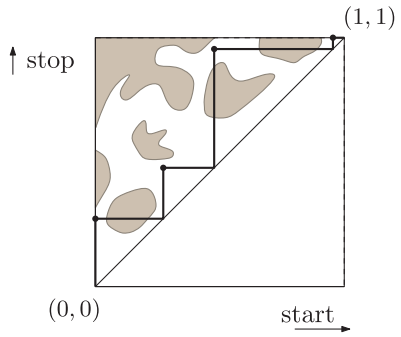


Fig. 3. The ssd and a valid segmentation into four segments.

segmentation of I based on the function f trivially induces a segmentation of T (see Figure 1); therefore, we may use the term "segmenting f" to denote the resulting segmentation. We assume that f is piecewise constant and we call the points where the value of f changes the *breakpoints*. The portions between consecutive breakpoints, where f stays constant, are called *pieces*.

The discrete segmentation problem is defined analogously. Here, a function $f : U \to \mathbb{R}$, where $U = \{1, \ldots, n\}$ is an index set, is given. A segmentation is a partition of U into disjoint contiguous subsequences called *discrete segments*. The discrete segmentation problem on a given criterion is to compute a segmentation into a minimum number of valid discrete segments. We will address the *weighted* version: f is given as a sequence $S = s_1, \ldots, s_n$ of pairs $s_i = (v_i, w_i)$, where v_i is the value of piece i and w_i is its weight. Values and weights influence the criterion and therefore the validity ofsegments.

2.1. The Start-Stop Diagram

To compute a minimal segmentation of f, we define the *start-stop diagram*. Consider the parameter space of the set of subintervals of I. For any candidate segment $[a, b]$, we associate the start parameter a with the horizontal axis and the stop parameter b with the vertical axis in the diagram. For continuous segmentation, any point (a, b) in this diagram, with $a < b$, represents a candidate segment. Thus, the set of candidate segment s is represented by the points in the upper left triangle of the unit square. The set of points that represent valid segments defines the *free space* in the start-stop diagram. The remaining points constitute the *forbidden space*. A segmentation of I into a sequence of segments $[\tau_0, \tau_1], [\tau_1, \tau_2], [\tau_2, \tau_3], \ldots, [\tau_{k-1}, \tau_k]$, $\tau_0 = 0$ and $\tau_k = 1$, corresponds to a *staircase* in the start-stop diagram whose convex vertices correspond to the points (τ_i, τ_{i+1}) and whose concave vertices (τ_i, τ_i) lie on the main diagonal. Note that a segmentation is valid if and only if all convex vertices lie in the free space. See Figure 3 for an example. The size of a staircase, that is, the number of convex vertices, corresponds to the size of the segmentation. The start-stop diagram is reminiscent of the free space diagram used to compute the Fréchet distance [Alt and Godau 1995]. However, in that problem, one is interested in a path that stays inside the free space entirely.

For discrete segmentation, we consider the *start-stop matrix*, a simplification of the start-stop diagram that is an $n \times n$ matrix B with Boolean values, where the entry $B(i, j)$ at the ith column and jth row from below corresponds to the value of the criterion function at (i, j).[1]

[1]Usually a matrix is indexed by row first and column second. We switch the indices to be consistent with the continuous case, and because it feels more natural in light of the application.

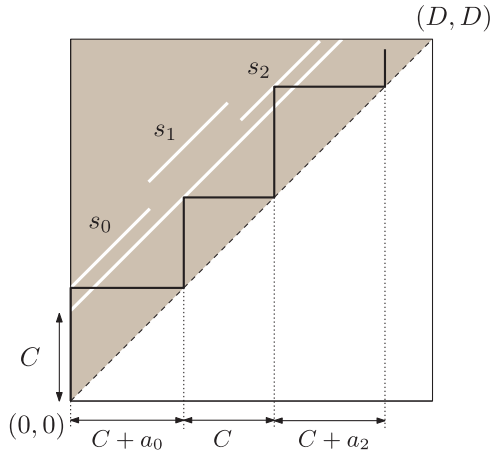


Fig. 4. Reduction from subset sum to the abstract segmentation problem. The staircase shows the sum $a_0 + a_2$.

2.2. Segmenting in the Discrete Case

In the discrete case, it is relatively easy to compute a minimal segmentation by using dynamic programming. If one treats $B(\cdot, \cdot)$ as the adjacency matrix of an unweighted directed acyclic graph, a minimal segmentation corresponds precisely to a shortest path from 1 to n and can be found in $O(n^2)$ time [Dasgupta et al. 2008].

THEOREM 2.1. *Given an $n \times n$ start-stop matrix, one can check if a minimal discrete segmentation exists, and if so compute it, in $O(n^2)$ time and space.*

2.3. Segmenting in the Continuous Case

Continuous segmentation is much harder. In the *abstract segmentation problem*, we are given a decomposition D of the triangle spanned by $(0, 0)$, $(0, 1)$, and $(1, 1)$ into points; a-monotone bounded-degree curve segments (with horizontal axis a); and faces, each of which is marked either *free* or *forbidden*. Let n be the total complexity of D. In the resulting start-stop diagram, we need to find a staircase that represents a valid, minimal segmentation. We show that even testing the existence of a valid segmentation is already NP-hard.

THEOREM 2.2. *The abstract segmentation problem is NP-hard.*

PROOF. We reduce from SUBSET-SUM. Let $a_0, \ldots, a_{n-1}$ be a set of positive integers and let B be the desired subset sum. We generate an instance of the abstract segmentation problem by constructing $n + 1$ line segments in the start-stop diagram; these line segments are precisely the free space.

Let $A = \sum_{i=0}^{n-1} a_i$, $C = A + 1$, and $D = (n + 1)C + B$. We generate a start-stop diagram of size $[0, D] \times [0, D]$, for ease of description; formally, it needs to be scaled to fit in the unit square. One line segment s of the free space has its endpoints at $(0, C)$ and at $(D - C, D)$ (see Figure 4).

For each a_i we create a line segment s_i with endpoints $(iC, (i + 1)C + a_i)$ and $(iC + A, (i + 1)C + a_i + A)$. This segment lies a_i units vertically above the first line segment s in the start-stop diagram. The placement of the segments is such that each staircase must choose to have its first convex vertex on s or s_0, its second convex vertex on s or s_1, and so on. Each subsequent step places the concave vertex C units further along the diagonal if s is chosen, and $C + a_i$ units if s_i is chosen. Each staircase that ends at

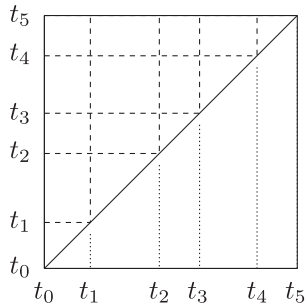


Fig. 5. The subdivision of the ssd into cells according to the breakpoints of f.

(D, D) has exactly $n+1$ convex vertices, the last one necessarily on s. Furthermore, we can end at (D, D) if and only if the chosen s_i are such that the corresponding values a_i sum up to B.

The length of the segments s_i is chosen so that we can select s_i no matter what segments have been picked before. At the same time, the step size of at least C in every step makes sure that we cannot make more than one step on the same s_i. Clearly, the reduction is polynomial. □

3. SOLVING THE CONTINUOUS SEGMENTATION PROBLEM

Even though the general continuous segmentation problem is NP-hard, we identify some properties of the start-stop diagram that make the problem tractable. In the following, we assume that the start-stop diagram is given to us in the form of an abstract segmentation problem as defined in Section 2.3. In addition, we assume that we are given a grid decomposition of this diagram of size $n \times n$ or smaller. When f is a piecewise-constant function, for example, there is a natural decomposition of the start-stop diagram into a grid by the breakpoints of f; see Figure 5. In this case, each cell of the grid corresponds to a set of candidate segments where the start and stop points lie on fixed pieces of f. The cells incident to the diagonal are triangular and represent candidate segments with start and stop points lying in the same piece.

Computing the reachable space. Recall that a minimal segmentation corresponds to a minimum-link staircase in the start-stop diagram. We define the *i-reachable space* R_i as the set of points on the diagonal that can be reached from $(0, 0)$ by a valid staircase of size at most i. Whenever it causes no confusion, we identify the diagonal of the start-stop diagram with I. In particular, we identify the (connected components of the) reachable space R_i with a set of *intervals* on I. The number of intervals is the *complexity* of the reachable space. For a fixed i, the *incoming i-reachable intervals* of a cell $C_{i'j'}$ are the intervals of R_i intersected with the i'th column. We call the union of vertical (horizontal, respectively) lines intersecting an interval X the vertical slab (horizontal slab, respectively) induced by X. Consider the valid staircases of size at most $i+1$ that have their last convex vertex in such a vertical slab V_X. We call the connected components of the set of points on the diagonal that can be reached by such a staircase the *outgoing $(i+1)$-reachable intervals* produced by X.

The following algorithm describes an iterative procedure to compute R_i, for $i = 1, 2, \ldots$, starting with $R_0 = \{(0, 0)\}$. The algorithm stops when it has found the smallest value of i for which R_i contains point $(1, 1)$. Let this value be k. An actual minimum-link staircase, and hence a minimal segmentation, can then be extracted from the sets $R_0, \ldots, R_k$.

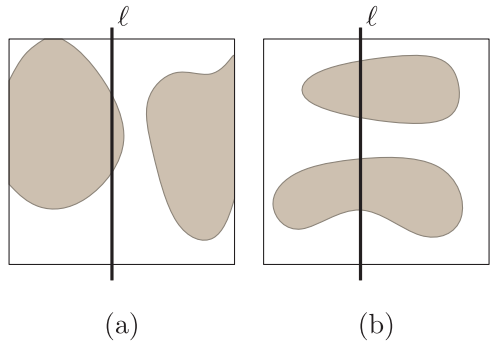


Fig. 6. (a) A verticonvex cell. (b) A tunnel cell.

ALGORITHM 1: ReachableSpace

Input: A start stop-diagram S
Output: The sets of reachable space $R_0, \ldots, R_k$, where k is the smallest i such that R_i contains an interval with point $(1, 1)$.
$i \leftarrow 0$; $R_0 \leftarrow \{(0, 0)\}$
while $(1, 1) \notin R_i$ **do**
  foreach *interval* X *in* R_i **do**
    Consider the intersection of the free space in S with the vertical slab induced by X and project it horizontally back to the diagonal.
  end
  The union of R_i and these projections, over all X, forms R_{i+1}.
  $i \leftarrow i + 1$
end
return $R_0, \ldots, R_i$

We will make this algorithm more concrete when proving Theorem 3.2, and we will see that the running time depends on the shape and the distribution of the forbidden space in each cell. For example, a monotone criterion yields a monotone curve in the start-stop diagram. The region above the curve is the forbidden space, and the region below the curve is the free space.

We call an object *verticonvex* if and only if its intersection with any vertical line ℓ is at most a single interval.[2] A start-stop diagram cell (or row) is *verticonvex* when the forbidden region within it is; see Figure 6(a). We call a cell a *tunnel cell* if any vertical line ℓ intersects the forbidden space in that cell in at most two intervals; see Figure 6(b) for an example. We will show that, for a given $n \times n$ start-stop diagram, we can compute a minimal segmentation in k segments in $O(kn^2)$ time if all cells are verticonvex, and in $O(k^2n^2)$ time if each row contains at most one tunnel cell and all other cells are verticonvex. If one allows cells where a vertical line can intersect the forbidden space in three or more intervals (i.e., double-tunnels), the problem becomes NP-hard as seen in the construction in the proof of Theorem 2.2. We will see later that even if we have only tunnel cells but more than one tunnel cell in each row, the reachable space may have exponential complexity.

[2]A verticonvex region is also called "x-monotone" in the literature (here it would be "a-monotone"); we choose to use the new term to avoid confusion with "monotone criteria."

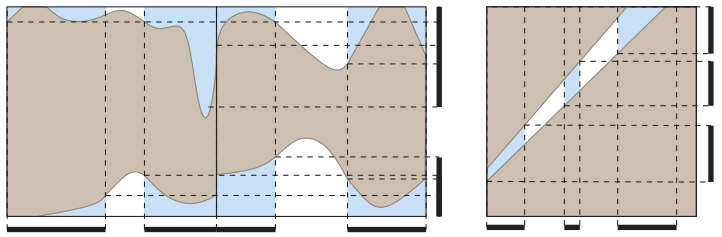


Fig. 7. (a) Any row of verticonvex cells produces at most two i-reachable intervals. (b) A tunnel cell can produce more than two i-reachable intervals.

3.1. Verticonvex Cells Only

Consider the reachable space algorithm (Algorithm 1). The intersection of a vertical line with the free space in a verticonvex cell consists of at most two (possibly empty) intervals: one connected to the top and one to the bottom of the row. The horizontal projection onto the diagonal preserves this property, and the union of the projections of this type also consists of at most two intervals (see Figure 7(a)).

OBSERVATION 3.1. *For any i and any row, the reachable space R_i produced by the incoming intervals in the verticonvex cells in the row consists of at most two intervals. One of these intervals is connected to the top of the row, and the other one is connected to the bottom of the row.*

We now prove:

THEOREM 3.2. *Given an $n \times n$ start-stop diagram in which the forbidden space in each cell is verticonvex and has constant complexity, we can compute a minimal segmentation in $O(kn^2)$ time, where k is the size of a minimal segmentation.*

PROOF. Assume that for the given instance of the problem, a valid segmentation exists. We specialize the procedure used in Algorithm 1. Recall that the reachable space can be encoded as a set of intervals, each of which corresponds to a connected component on the diagonal. Since all cells are verticonvex, Observation 3.1 implies that the i-reachable space, for any i, consists of $O(n)$ such intervals and as such we can store it as a set of $O(n)$ values. Given the reachable space R_i, we compute the reachable space R_{i+1} in a row-by-row manner. By Observation 3.1, the $(i + 1)$-reachable space restricted to any row consists of two intervals, one connected to the top and one to the bottom. We "grow" these two intervals, maintaining the top endpoint of the bottom one and the bottom endpoint of the top one while iterating over the cells in the current row and handling the i-reachable intervals incoming to every cell. Observation 3.1 implies that there are at most two incoming i-reachable intervals to any cell, since it is nothing more than the i-reachable space restricted to the column that contains the cell. Furthermore, the complexity of the forbidden space in each cell has constant complexity. Therefore, we spend constant time per cell during this process and $O(n^2)$ time in total for computing R_{i+1} from R_i. We perform this step until the top-right corner $(1, 1)$ is contained in R_{i+1}. Since this happens for $i + 1 = k$, this takes $O(kn^2)$ time. An optimal segmentation can be extracted by standard dynamic programming methods. □

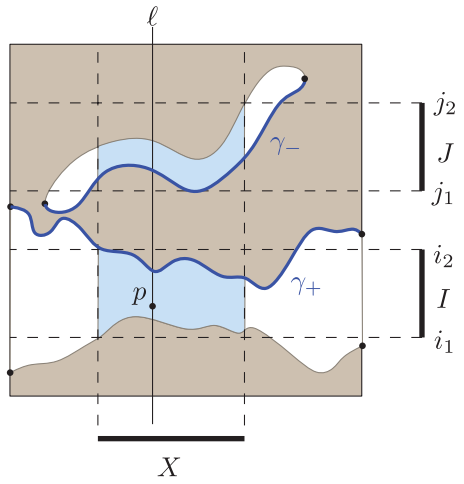


Fig. 8. Illustration to the proof of Lemma 3.3. The vertices of the forbidden space are also shown.

3.2. At Most One Tunnel Per Row

Our reachable space algorithm is efficient when all rows in the start-stop diagram are verticonvex. If a row is not verticonvex, it may produce more than two intervals. Nonetheless, we show that if the start-stop diagram is not too complex—specifically, if each row contains at most one tunnel cell—the problem can still be solved efficiently. We start with the following technical result.

LEMMA 3.3. *Let C be a tunnel cell, and let $F \subseteq C$ be the forbidden space within C. An incoming interval X can produce at most one outgoing interval I such that (1) I is incident neither to the top nor to the bottom of the row, and (2) I does not intersect a horizontal line through a vertex of F.*

PROOF. Assume, for the sake of contradiction, that X produces two intervals $I = [i_1, i_2]$ and $J = [j_1, j_2]$ with this property, with $i_2 < j_1$ (see Figure 8). Let H_I denote the horizontal slab induced by I, define H_J similarly, and let V_X denote the vertical slab induced by X. Since I does not intersect a horizontal line through a vertex of F, H_I does not contain any vertices. The same holds for H_J.

Since X produced the outgoing interval I, there must be at least one point p of free space in $V_X \cap H_I$. Let ℓ be the vertical line through p. Let p_+ be the first intersection of ℓ with the boundary of free space above p; p_+ may coincide with p. Notice that p_+ must lie in the interior of a boundary edge γ_+ of free space: p_+ may not be a vertex, as there are no vertices in H_I. The edge γ_+ may not cross the top boundary of $V_X \cap H_I$, for otherwise I would be larger.

Repeating the construction for segment J, we obtain a free space point $q' \in V_X \cap H_J$ on a different vertical line ℓ'. Arguing as earlier, but now working downward, we obtain a boundary edge γ_- of free space that does not cross the bottom edge of $V_X \cap H_J$. Since γ_- extends all the way across $V_X \cap H_J$, it must cross ℓ within that rectangle and lie below some free space point $q \in \ell \cap H_J$.

We have obtained a contradiction, as now ℓ intersects F at least three times: once below H_I (since I is not incident to the bottom of the row), once between γ_+ and γ_-, and once above H_J (since J is not incident to the top of the row), so C is not a tunnel cell. □

LEMMA 3.4. *Let C be a tunnel cell, and let $F \subseteq C$ be the forbidden space within C. Suppose C has m incoming i-reachable intervals and u is the complexity of the union of the outgoing $(i+1)$-reachable intervals produced by them. Then $u \leq c + m + 2$, where c is the number of vertices of F.*

PROOF. We draw a horizontal line h_v through every vertex v of F. Let H denote this set of lines. We charge every interval in the union of the outgoing intervals that intersects such a line h_v to v. This way, each vertex gets charged at most once, since two intervals that would charge the same line cannot be disjoint. Thus, we have at most c charges of this type.

By Lemma 3.3, each incoming interval can produce at most one outgoing interval that does not intersect a line of H and is incident neither to the top nor to the bottom of the row. Therefore, we can charge those to the incoming intervals with at most m charges.

The remaining intervals in the union are incident either to the top or to the bottom of the row, and of those there can be at most two in total. This completes the proof. □

LEMMA 3.5. *In an $n \times n$ start-stop diagram in which (1) the forbidden space in each cell has constant complexity, (2) each row contains at most one tunnel cell, and (3) the remaining cells are verticonvex, R_i consists of at most $O(in)$ intervals.*

PROOF. Consider the i-reachable space R_i restricted to one row of the grid. It consists of the endpoints of all valid staircases that have their last convex vertex in a cell of this row.

We distinguish two types of staircases: (1) those that have their last convex vertex in a verticonvex cell and (2) those that have it in a tunnel cell. By Observation 3.1, the i-reachable space that is formed by the staircases of type (1) has constant complexity.

As for the staircases of type (2), they all have their last convex vertex in the same cell since there is only one tunnel cell per row. By Lemma 3.4, and since the forbidden space in every cell has constant complexity, the i-reachable space formed by those staircases consists of $m + O(1)$ intervals, where m is the complexity of the $(i-1)$-reachable space R_{i-1} restricted to the column that the tunnel cell lies in.

Thus, we have the following recurrence for the complexity of the i-reachable space restricted to one row: $C(i) \leq C(i-1) + O(1)$. Clearly, $C(1) \leq 3$, and therefore, we have that $C(i) = O(i)$. Since there are n rows, the lemma follows. □

Using essentially the same algorithm as before, we now obtain:

THEOREM 3.6. *Given an $n \times n$ start-stop diagram in which (1) the forbidden space in each cell has constant complexity, (2) each row contains at most one tunnel cell, and (3) the remaining cells are all verticonvex, we can compute a minimal segmentation in $O(k^2n^2)$ time, where k is the size of a minimal segmentation.*

PROOF. We again use Algorithm 1 and traverse the start-stop diagram as described in the proof of Theorem 3.2. Since the complexity of the forbidden space is constant in each cell, we can list a set of m incoming intervals and produce at most $m + O(1)$ outgoing intervals in $O(m)$ time. Again, the bottleneck of the computation is the complexity of the reachable space and the traversal of the cells. □

4. COMBINING CRITERIA

We can also consider segmenting a trajectory based on multiple criteria. For example, we want segments such that two criteria hold simultaneously, or where at least one criterion holds. We can combine criteria into new ones by taking conjunctions,

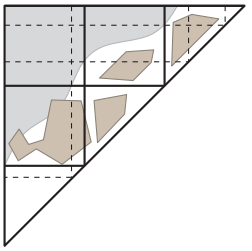


Fig. 9. The ssd for the conjunction of a monotone and a verticonvex criterion. The forbidden space resulting from the monotone criterion is shown in gray. We further subdivide the ssd (dashed lines) such that the boundary of this region γ intersects at most one cell in row and in any column.

disjunctions, and negations, and in general we can build any Boolean combination this way.

Again, we assume that there exists a grid decomposition of size $n \times n$ or smaller of the start-stop diagram for each of the criteria. To obtain the start-stop diagram for the criterion $C_1 \wedge C_2$, we simply overlay their grids and take the union of the forbidden space F_1 of C_1 and the forbidden space F_2 of C_2. Similarly, the forbidden space for $C_1 \vee C_2$ is the intersection of F_1 and F_2. Can we still solve the segmentation problem efficiently on the resulting start-stop diagram?

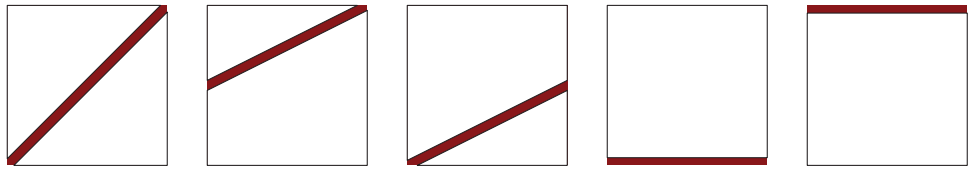
Buchin et al. [2011] observe that the conjunction or disjunction of monotone criteria is again monotone. Similarly, observe that the start-stop diagram of a disjunction of two *verticonvex criteria*, that is, criteria for which the start-stop diagram contains only verticonvex cells, again contains only verticonvex cells. Hence, the disjunction of two verticonvex criteria is again a verticonvex criterion. Since a monotone criterion is also verticonvex, the disjunction of a monotone criterion with a verticonvex criterion results in a verticonvex criterion as well.

For the conjunction of two verticonvex criteria, we take the union of their forbidden spaces. Unfortunately, this can lead to nonverticonvex cells. However, we can show that if one of the criteria is monotone, we can slightly modify the grid of the combined start-stop diagram such that it contains at most one tunnel cell in each row and the remaining cells are verticonvex. By Theorem 3.6, we can therefore still compute a minimal segmentation efficiently.

THEOREM 4.1. *Let C_1 be a monotone criterion, let C_2 be a verticonvex criterion, and let the overlay of their grids have size $n \times n$. If the complexity of the forbidden space in every cell is constant, then we can compute a minimal segmentation with respect to the criterion $C_1 \wedge C_2$ in $O(k^2n^2)$ time, where k is the size of a minimal segmentation.*

PROOF. We argue that there exists $O(n) \times O(n)$ refinement of the overlay of the two grids such that (1) every row contains at most one tunnel cell, and (2) the remaining cells are verticonvex. The claim then follows from Theorem 3.6. Recall that a monotone criterion corresponds to an ab-monotone curve γ in the start-stop diagram (with axes a and b), in which all points above the curve form the forbidden space, and all points below the curve lie in free space. We subdivide the grid such that γ intersects at most one cell in any row and at most one cell in any column (see Figure 9). Since γ is ab-monotone, it intersects an original grid line at most once; this means we add at most $O(n)$ grid lines. The complexity of the forbidden space in each cell is constant.

Clearly, the cells of the refined grid that are not intersected by γ are verticonvex: everything above γ is forbidden space, and the forbidden space in other cells originates only from a verticonvex criterion. The cells that are intersected by γ are tunnel cells



(a) $f_a(x) = [x]$ (b) $f_b(x) = \left[\frac{x+1}{2}\right]$ (c) $f_c(x) = \left[\frac{x}{2}\right]$ (d) $f_d(x) = [0]$ (e) $f_e(x) = [1]$

Fig. 10. Gadget cells used in the lower-bound constructions. The (red) line segment indicates the free space.

since every vertical line intersects the forbidden space at most twice: at most once for the forbidden space resulting from the verticonvex criterion, and at most once for the forbidden space above γ. The claim now follows from Theorem 3.6. □

Note that essentially the same approach can also be used to solve the problem for the conjunction of a verticonvex criterion and the negation of a monotone criterion:

COROLLARY 4.2. *Let C_1 be a monotone criterion, let C_2 be a verticonvex criterion, and let the overlay of their grids have size $n \times n$. If the complexity of the forbidden space in every cell is constant, then we can compute a minimal segmentation with respect to the criterion $\neg C_1 \wedge C_2$ in $O(k^2n^2)$ time, where k is the size of a minimal segmentation.*

5. LOWER BOUNDS

In this section, we present lower bounds on the complexity of the reachable space for several types of start-stop diagrams. In particular, we show that the k-reachable space in a start-stop diagram representing the conjunction of a monotone and a verticonvex criterion may have complexity $\Omega(kn)$, and that if we have more than one tunnel cell per row, then the k-reachable space may have exponential complexity.

Gadgets. We construct the start-stop diagram by gluing together individual cells. Given a start-stop diagram, let C_{ij} be the cell in column i and row j. We identify C_{ij} with the unit square $[0, 1] \times [0, 1]$, which allows us to describe C_{ij} using a combination of gadget cells, or *gadgets* for short. A gadget acts as a function, mapping reachable points in the incoming intervals of the cell to the reachable points in the outgoing intervals. It will, however, be convenient to model each gadget α as a function f_α mapping incoming points $x \in [0, 1]$ to *a set* of points $f_\alpha(x) \subseteq [0, 1]$. This set $f_\alpha(x)$ corresponds to the free space within the cell on the vertical line at x. The gadgets that we will use are $f_\emptyset(\cdot) = \emptyset$, to model cells that are completely forbidden, and the gadgets shown in Figure 10. Note that gadgets (a), (b), and (c) are tunnel cells, while gadgets (d) and (e) are verticonvex cells. We write $[x]$ for the interval (set) consisting of just x and $f(X)$ for the image of a subset $X \subseteq [0, 1]$ under f.

When combining cells C_{ij} and $C_{(i+1)j}$ into the final start-stop diagram, we use the description of $C_{(i+1)j}$ for their shared border. Similarly, we give preference to $C_{i(j+1)}$ over C_{ij} for their shared border. Furthermore, let f_{ij} be the function describing C_{ij}. For row j, we then define

$$R_k^j = \bigcup_{i=1}^{j-1} f_{ij}\left(R_{k-1}^i\right) \text{ and } R_1^j = f_{0j}(0). \tag{1}$$

The set R_k^j denotes the k-reachable space restricted to the jth row (expressed in the coordinates of the row). We can obtain the reachable space R_k by reparameterizing each R_k^j and taking the union over all j.

In all our constructions, the triangular cells incident to the diagonal are completely forbidden. So a staircase ending in the jth row can have at most $j-1$ steps. Thus, $R_k^j = R_{k-1}^j$ for $k \geq j$. (Recall that the k-reachable space corresponds to the valid staircases with at most k steps.)

5.1. At Most One Tunnel Per Row

The running time in Theorem 4.1 (and Theorem 3.6) is a factor k worse than that in Theorem 3.2. We now give an example of an $n \times n$ start-stop diagram resulting from the conjunction of a monotone and a verticonvex criterion in which the complexity of the k-reachable space is $\Omega(kn)$. In this start-stop diagram, every cell has constant complexity, every row contains at most one tunnel cell, and all the remaining cells are verticonvex. Thus, the lower bound of Theorem 5.1 matches the worst-case upper bound we give in Lemma 3.5. This explains the extra factor k in the running time of our algorithm, since by Observation 3.1, the complexity of the k-reachable space is $O(n)$ for any k if all cells are verticonvex.

THEOREM 5.1. *There exists an $n \times n$ start-stop diagram such that (1) the forbidden space is the union of the region above a monotone curve and one verticonvex region per cell, (2) the forbidden space in each cell is polygonal and has constant complexity, and (3) the complexity of the k-reachable space is $\Omega(n^2)$, where k is the size of the optimal segmentation.*

PROOF. We construct a start-stop diagram cell by cell using the gadgets described previously. On the basis of the description of the reachable space in Equation (1), we want the reachable space to satisfy the following recurrence:

$$R_k^j = f_d\big(R_{k-1}^{j-1}\big) \cup f_b\big(R_{k-1}^{j-1}\big) \text{ and } R_1^2 = [0] \cup \left[\frac{1}{2}\right]. \tag{2}$$

To accomplish this, we place the gadgets as follows. For cells C_{ij}, with $j = i+1$ and $j < n$, we use an overlay of gadgets (b) and (d) from Figure 10. For $j = n$ and $i = n-1$, we use gadget (e). All other cells are forbidden. The overall diagram is shown in Figure 11. It can be seen as a conjunction of a monotone criterion and a verticonvex one.

It is easy to prove by induction that the following holds for $j \in 2, \ldots, n-1$:

$$R_{j-1}^j = [0] \cup \bigcup_{i=1}^{j-1} \left[\frac{2^i - 1}{2^i}\right]. \tag{3}$$

In our construction, the minimum size of an optimal segmentation is at least $n-2$. This follows from the fact that the cells C_{ij} with $j \geq i+2$ are completely forbidden. Furthermore, by Equation (3), for any j, all intervals in R_{j-1}^j are pairwise disjoint. Note also that across the rows, the intervals are disjoint. Therefore, the total number of these intervals over all rows is a lower bound on the complexity of the k-reachable space, where k is the size of an optimal segmentation. Thus,

$$\sum_{j=1}^{n-1} \big|R_k^j\big| \geq \sum_{j=2}^{n-1} \big|R_{j-1}^j\big| = \sum_{j=2}^{n-1} j = \Omega(n^2),$$

where $|\cdot|$ denotes the description complexity[3] of the set. This completes the proof. □

[3]Our sets are finite unions of intervals. By the *description complexity* of a set, we mean the minimum number of intervals required to write the set as a union.

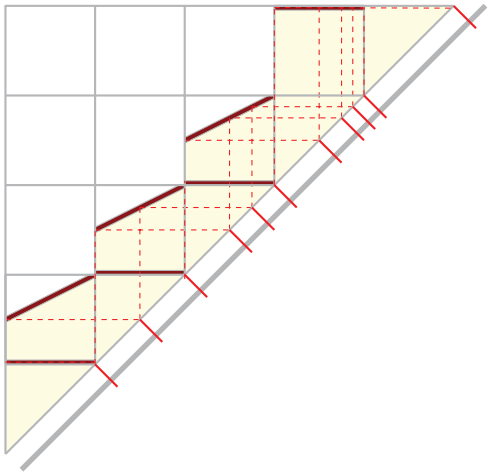


Fig. 11. A start-stop diagram of the conjunction of a monotone criterion and a verticonvex one that results in a reachable space of complexity $\Omega(n^2)$. The forbidden space is shown in light colors and the free space in dark ones: the forbidden space of the monotone criterion (white), the forbidden space of the verticonvex criterion (light yellow), and the free space (dark red).

5.2. More than One Tunnel Per Row

Next, we show that if we allow two tunnel cells per row, then the reachable space can have exponential complexity. The idea of the construction is the following. We use a block of three gadgets (a), (b), and (c) from Figure 10. We arrange the three gadgets so that the complexity of the reachable space is doubled every two rows. The construction is illustrated in Figure 12. Indeed, the tunnel in gadget (a) exactly copies the reachable space in column i to column $i+1$, and the tunnels in gadgets (b) and (c) "compress" both copies of the reachable intervals and project them onto column $i+2$. Since the reachable space in the first column consists of exactly one point, this leads to a reachable space in the last column that consists of an exponential number of isolated points.[4]

THEOREM 5.2. *There exists an $n \times n$ start-stop diagram such that (1) the forbidden space in each cell has constant complexity, (2) each row contains at most two tunnel cells, (3) the remaining cells are all verticonvex, and (4) the complexity of the k-reachable space R_k is $\Omega(2^{n/4})$, where k is the size of the optimal segmentation.*

PROOF. We again use the gadgets from Figure 10. For cell C_{12}, we use gadget (d); for $C_{(n-1)n}$, we use gadget (e). For the cells $C_{i(i+1)}$, we use gadget (a) when i is even and gadget (c) when i is odd. For $C_{i(i+2)}$, we use gadget (b). All remaining cells are forbidden. The overall construction is shown in Figure 12. We express the reachable space in the form of Equation (1).

$$R_k^j = f_c\big(R_{k-1}^{j-1}\big) \cup f_b\big(R_{k-1}^{j-2}\big) = f_c\big(f_a\big(R_{k-2}^{j-2}\big)\big) \cup f_b\big(R_{k-1}^{j-2}\big) \text{ and } R_2^2 = R_1^2 = [0]. \tag{4}$$

[4]It may appear that in this construction one could simply refine the grid by adding a horizontal grid line through the center of every row and obtain a start-stop diagram where every row has at most one tunnel. This would seem to contradict Lemma 3.5, which states that in such a start-stop diagram the i-reachable space has complexity $O(in)$. However, since the grid has to be symmetric, we also have to add a vertical line through the point where the new horizontal line crosses the main diagonal. It so happens that the resulting vertical line again cuts a tunnel cell into two, resulting in two tunnel cells per row. Incrementally adding lines in this manner until there is at most one tunnel in every row leads to a grid of exponential size.

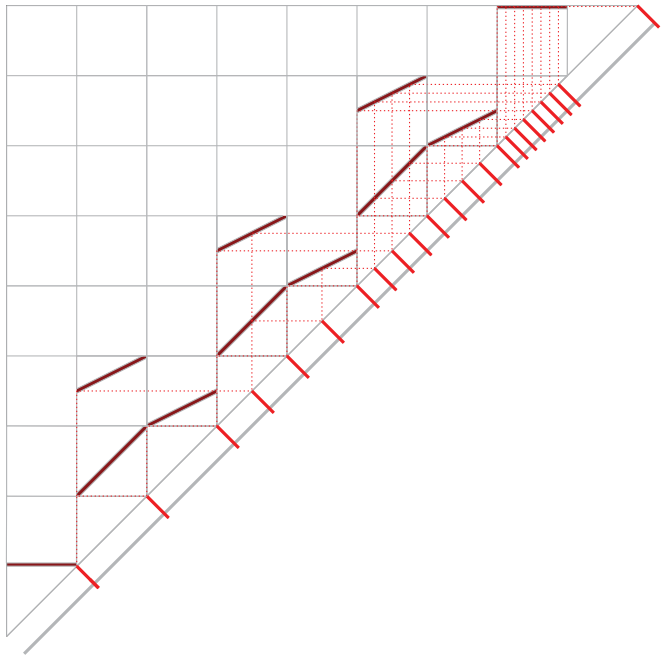


Fig. 12. A start-stop diagram with at most two tunnels per row that results in a reachable space of exponential complexity; forbidden space is shown in white.

We claim that the following holds for even j larger than 2:

$$R_{j-1}^{j} = \bigcup_{i=0}^{2^{j/2-1}-1} \left[\frac{i}{2^{j/2-1}} \right]. \tag{5}$$

We prove this by induction. For $j = 2$, this is clearly true. For $j \to j+2$, using Equation (4), we obtain

$$R_{j+1}^{j+2} = f_c(R_{j-1}^{j}) \cup f_b(R_j^j).$$

By the induction hypothesis for R_{j-1}^{j} and since $R_j^j = R_{j-1}^{j}$:

$$\begin{aligned} R_{j+1}^{j+2} &= \bigcup_{i=0}^{2^{j/2-1}-1} \left[\frac{i}{2^{j/2}} \right] \cup \bigcup_{i=0}^{2^{j/2-1}-1} \left[\frac{i}{2^{j/2}} + \frac{1}{2} \right] \\ &= \bigcup_{i=0}^{2^{j/2-1}-1} \left[\frac{i}{2^{j/2}} \right] \cup \bigcup_{l=2^{j/2-1}}^{2^{j/2}-1} \left[\frac{l - 2^{j/2-1} + 2^{j/2-1}}{2^{j/2}} \right] \\ &= \bigcup_{i=0}^{2^{j/2}-1} \left[\frac{i}{2^{j/2}} \right]. \end{aligned}$$

This proves Equation (5) for even j. Since the intervals generated by this sequence are pairwise disjoint, the complexity of R_{j-1}^{j} is at least $2^{j/2}$. A valid segmentation has to have at least $\lfloor n/2 \rfloor$ steps, since all cells C_{ij} with $j \geq i+3$ are completely forbidden. Thus, the complexity of the k-reachable space, where k is the size of the

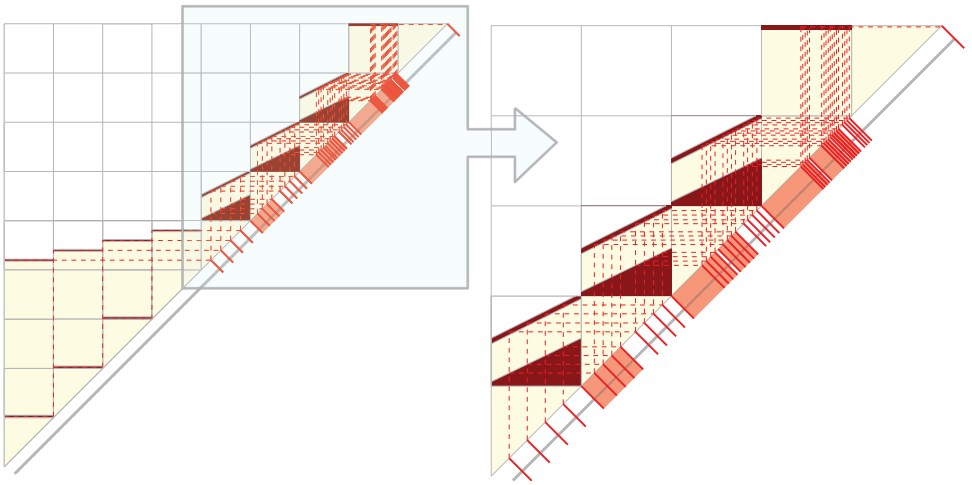


Fig. 13. A start-stop diagram of the conjunction of a monotone criterion and a verticonvex one that results in a high-complexity incremental reachable space. The forbidden space is shown in light colors and the free space in dark colors: the forbidden space of the monotone criterion (white), the forbidden space of the verticonvex criterion (light yellow), and the free space (dark red).

optimal segmentation, is lower bounded by the complexity of $R^{\lfloor n/2 \rfloor}_{\lfloor n/2 \rfloor - 1}$. Therefore, the complexity is at least $\Omega(2^{n/4})$. □

5.3. Incremental Reachability

One might argue that to solve the segmentation problem, we only need to maintain the incremental reachable space, that is, the subset of the diagonal $R_i \setminus R_{i-1}$ reachable with exactly i steps. Next we show that this set can also have high complexity. To this end, we extend the construction given in Theorem 5.1.

Observation 5.3. *There exists an $n \times n$ start-stop diagram such that (1) the forbidden space is the union of the region above a monotone curve and one verticonvex region per cell, (2) the forbidden space in each cell is polygonal and has constant complexity, and (3) the complexity of $R_i \setminus R_{i-1}$ is $\Omega(i^2)$ for $i \in 2, \ldots, \lfloor n/2 \rfloor$.*

Proof. In this construction, we use the gadgets from Figure 10, as well as an additional (type of) gadget similar to gadget (d). This new gadget $g_i = [x_i] = [2i/n]$, for $i \in 1, \ldots, \lfloor n/2 \rfloor$, is essentially a constant function with value x_i. Assume for simplicity of presentation that n is odd and let $m = \lceil n/2 \rceil$ be the index of the middle row. We place the cell with gadget function g_i in the ith column and mth row. Furthermore, there are gadgets of type (d) in the ith column of row $i + 1$ for $i \in 0, \ldots, m - 1$. The diagram for columns $i \geq m$ is a copy of the construction used in Theorem 5.1, which we modify as follows. Instead of using an overlay of gadgets (b) and (d), we overlay gadget (b) with a modification of gadget (c), which maps x to an interval instead of a single value. The gadget function is given by

$$f_g(x) = \left[0, \frac{x}{2}\right].$$

Refer to Figure 13 for an example of the overall construction.

For the reachable space in the mth row, we have

$$R_i^m = \bigcup_{j=1}^{i} [x_j].$$

For the upper part of the diagram (i.e., columns with index greater than m), we have a recurrence similar to that of Equation (2):

$$R_k^{m+q} = f_g(R_{k-1}^{m+q-1}) \cup f_b(R_{k-1}^{m+q-1}).$$

It is easy to prove by induction on q that the following holds:

$$R_{i+q}^{m+q} = \bigcup_{l=0}^{q-1} \left[1 - \frac{1}{2^l}\ ,\ 1 - \frac{1}{2^{l+1}} - \frac{1-x_i}{2^q}\right] \cup \bigcup_{j=1}^{i} \left[1 - \frac{1-x_j}{2^q}\right],$$

for $i \in 1, \ldots, \lfloor \frac{n}{2} \rfloor$ and $m + q < n$.

Now we are interested in the incremental reachable space, which can be written as follows:

$$R_{i+q}^{m+q} \setminus R_{(i-1)+q}^{m+q} = \bigcup_{l=0}^{q-1} \left[1 - \frac{1}{2^{l+1}} - \frac{1-x_{i-1}}{2^q}\ ,\ 1 - \frac{1}{2^{l+1}} - \frac{1-x_i}{2^q}\right] \cup \left[1 - \frac{1-x_i}{2^q}\right].$$

Since the intervals in the previous equation are pairwise disjoint (since $x_i \in (0, 1)$ for all i), we have that

$$\left|R_{i+q}^{m+q} \setminus R_{(i-1)+q}^{m+q}\right| = q + 1,$$

where $|\cdot|$ denotes the description complexity of the set. Therefore, we can write

$$\sum_{p=1}^{m+k-1} \left|R_k^p \setminus R_{k-1}^p\right| \geq \sum_{p=m}^{m+k-1} \left|R_k^p \setminus R_{k-1}^p\right| = \sum_{q=0}^{k-1} \left|R_k^{m+q} \setminus R_{k-1}^{m+q}\right| = \sum_{q=0}^{k-1} (q+1) = O(k^2),$$

using $k = i + q$ for $i \in 1, \ldots, \lfloor \frac{n}{2} \rfloor$. This completes the proof. □

6. COMPUTING THE START-STOP DIAGRAM

In previous sections, we focused on computing minimal segmentations for a given start-stop diagram or start-stop matrix. We now discuss two specific criteria, prove that they are verticonvex, and explain how to construct the corresponding start-stop diagram or start-stop matrix efficiently. For both the discrete and the continuous segmentation problem, this leads to polynomial-time algorithms. The criteria are defined on a piecewise-constant function f. We also show that if f is piecewise linear, then the start-stop diagram may contain more than one tunnel cell per row.

6.1. Segmentation on the Outlier-Tolerant Criterion

For a given piecewise-constant function f, we can segment f such that in each valid segment $[a, b]$ the minimum and maximum values differ by at most h. To accommodate outliers, we require that only a fraction ρ, for a given constant $0 \leq \rho \leq 1$, of a valid segment $[a, b]$ must have its values within a range of extent h, and a fraction $1 - \rho$ of $[a, b]$ may have any value.[5] We present efficient algorithms for computing the start-stop matrix and the start-stop diagram for this criterion. To obtain a segmentation

[5]Using the same algorithm, we can also handle the following similar criterion by using the logarithm of the function. For a given piecewise-constant positive function f, we can segment f so that in each valid segment $[a, b]$ there is a portion V of total length at least $\rho(b - a)$ such that $w_{\max}/w_{\min} \leq h$, where $w_{\min}$ and $w_{\max}$ are the minimum and maximum function values that occur in V.

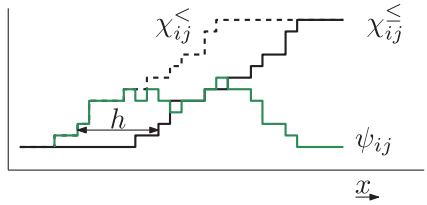


Fig. 14. Obtaining the function ψ_{ij} from $\chi_{ij}^{\leq}$ and $\chi_{ij}^{<}$ shifted by h to the left (dashed).

algorithm for the continuous case, we further show that the forbidden space within each cell of the start-stop diagram is verticonvex.

6.1.1. Discrete Case. We consider f as a sequence $s_1 = (v_1, w_1), \ldots, s_n = (v_n, w_n)$ of (value, weight) pairs. We first give a formal definition of the criterion and investigate its structure. Let S_{ij} be the contiguous subsequence $s_i, \ldots, s_j$ and let λ_{ij} be the total weight of its elements. Let $S_{ij}^{\leq}(x) = \{s_k \in S_{ij} \mid v_k \leq x\}$ denote the set of elements in S_{ij} with value at most x, and let $\chi_{ij}^{\leq}(x) = \sum_{s_k \in S_{ij}^{\leq}(x)} w_k$ be the total weight of these elements. We define $S_{ij}^{<}(x)$ and $\chi_{ij}^{<}(x)$ analogously. The total weight of the elements in S_{ij} with a value in the range $[x, x + h]$ is then

$$\psi_{ij}(x) = \chi_{ij}^{\leq}(x + h) - \chi_{ij}^{<}(x). \tag{6}$$

Our goal is now to segment f so that, for each segment S_{ij}, we have $\max_x \psi_{ij}(x)/\lambda_{ij} \geq \rho$. Let B be the start-stop matrix. To test the value B_{ij}, we could compute an explicit representation of $\chi_{ij}^{\leq}$ and an explicit representation of $\chi_{ij}^{<}$ shifted by h to the left, and then take their difference (see Figure 14). This takes $O((j - i)\log(j - i))$ time. Since there are $O(n^2)$ cells, the total amount of time required to compute the start-stop matrix is $O(n^3 \log n)$.

It is, however, not necessary to recompute the functions from scratch for each cell. Increasing j by one corresponds to raising the functions $\chi_{ij}^{\leq}$ and $\chi_{ij}^{<}$ by w_{j+1} for all values $x \geq v_{j+1}$. Therefore, we have

$$\psi_{i(j+1)}(x) = \begin{cases} \psi_{ij}(x) + w_{j+1} & \text{if } x \in [v_{j+1} - h, v_{j+1}], \text{ and} \\ \psi_{ij}(x) & \text{otherwise.} \end{cases} \tag{7}$$

We now store ψ_{ij} as a data structure that allows us to query the maximum of ψ_{ij} on any given interval and can be updated efficiently to represent $\psi_{i(j+1)}$. The data structure we use is an augmented segment tree that supports both operations in $O(\log n)$ time.[6]

Data structure to compute the start-stop matrix. We associate each piece $s_j = (v_j, w_j)$ of f with an interval $I_j = [v_j - h, v_j]$ of *weight* w_j. By Equation (7), it now follows that the value of $\psi_{ij}(x)$ is equal to the total weight of the intervals from $I_1, \ldots, I_j$ that contain x. Hence, we can represent ψ_{ij} using the set of intervals $I_1, \ldots, I_j$. We now describe an augmented segment tree T that stores these weighted intervals.

Let $u_1, \ldots, u_m$ be the endpoints of all intervals in $I_1, \ldots, I_n$ in sorted order. Internally, a segment tree T stores the elementary intervals $[u_i, u_{i+1}]$ in this order in the leaves of a balanced binary tree [de Berg et al. 1997]. The internal nodes store values that allow searching on u-value. Since we know the endpoints of the intervals of ψ_{ij} for all i and j in advance, we can initialize T with the set of $O(n)$ endpoints and all weights set

[6]The "segments" stored in this standard data structure as described in de Berg et al. [1997] are not to be confused with the segments of the segmentation of a trajectory.

to zero. Therefore, no rebalancing has to be done when adding or removing an interval stored in the tree; only the values stored in the nodes that relate to weights will change.

Each node ν in a segment tree has an associated range r_ν and an associated set $\mathcal{I}_\nu$ of intervals. The range r_ν is the union of the elementary intervals stored in (the leaves of) the subtree rooted at ν, and $\mathcal{I}_\nu$ is a subset of intervals stored in the tree. An interval I occurs in $\mathcal{I}_\nu$ if and only if I contains r_ν but not the range of ν's parent node [de Berg et al. 1997]. We now augment T such that each node ν stores the total weight A_ν of the intervals associated with ν.

So for a tree T representing the function ψ_{ij}, we can obtain the value of ψ_{ij} at x by searching for the elementary interval containing x and summing over the A-values on the search path.

A second augmentation provides us with a way to determine the maximum of ψ_{ij} in a given interval in $O(\log n)$ time. This is done by storing a value B_ν at every node ν that is the maximum sum of all A-values on a path from ν to a leaf in the subtree rooted at ν.

When querying T for the maximum of the function ψ_{ij} on a given interval $[a, b]$, we walk along the two paths from the root to the elementary intervals containing a and b and maintain the maximum of the B-values stored in the roots of the subtrees between the two paths. This takes time linear in the number of nodes visited, and hence we can find the maximum of ψ_{ij} on $[a, b]$ in $O(\log n)$ time.

When inserting an interval $I_j = [v_j - h, v_j]$ with weight w_j, updating A- and B-values can be done in $O(\log n)$ time. The A-value needs to be increased by w_j in the $O(\log n)$ nodes ν for which $I_j \in \mathcal{I}_\nu$. This operation is standard. The B-values have to be updated only in the nodes along the path from the root to the nodes where the A-values have been modified. Since those nodes lie along the two paths from the root to the elementary intervals storing $v_j - h$ and v_j, this can be done in $O(\log n)$ time overall.

LEMMA 6.1. *The start-stop matrix can be computed in $O(n^2 \log n)$ time.*

PROOF. We fill in the matrix B by testing the validity of subsequences of S. A single column of B corresponds to testing the validity of $S_{ii}, S_{i(i+1)}, \ldots, S_{in}$. This can be done in the given order (bottom-up in a column) in $O(n \log n)$ time by using the data structure described previously.

Assume that we have a tree T representing ψ_{ij} and that we have determined whether S_{ij} is valid. Then we insert I_{j+1} with weight w_{j+1} in the augmented tree and perform a query to determine $\max_x \psi_{i(j+1)}(x)$. We also compute $\lambda_{i(j+1)}$ from λ_{ij} by adding w_{j+1}. Now the test $\max_x \psi_{i(j+1)}(x)/\lambda_{i(j+1)} \geq \rho$ determines whether or not $S_{i(j+1)}$ is valid. □

We then use the algorithm from Theorem 2.1, and conclude:

THEOREM 6.2. *Given a piecewise-constant function f with n breakpoints, a threshold value $h > 0$, and a ratio $\rho \in [0, 1]$, we can compute a minimal discrete segmentation for the condition that, on a fraction of length at least ρ of a segment, the difference between the maximum and minimum function value is at most h, in $O(n^2 \log n)$ time.*

6.1.2. Continuous Case. For continuous segmentation, we are allowed to cut f at any two points $a, b \in [0, 1]$. So we need to determine the forbidden space in the start-stop diagram. The breakpoints of the function f decompose the start-stop diagram into a grid. We now prove that the forbidden space within a cell $C = [\underline{a}, \overline{a}] \times [\underline{b}, \overline{b}]$ of this grid can be described by four linear inequalities of the following form, where $\psi_1^*, \ldots, \psi_4^*$ are

constant values specific to the cell:

$$(b-a)\rho > \begin{cases} \psi_1^* + (\overline{a}-a) + (b-\underline{b}) \\ \psi_2^* + (\overline{a}-a) \\ \psi_3^* + (b-\underline{b}) \\ \psi_4^* \end{cases}. \tag{8}$$

LEMMA 6.3. *For any cell $C = [\underline{a}, \overline{a}] \times [\underline{b}, \overline{b}]$ of the start-stop diagram, there exist values of $\psi_1^*, \ldots, \psi_4^*$ such that the forbidden space in C is the intersection of four half-planes with the cell, described by the inequalities in Equation* (8).

PROOF. We use an approach similar to that used in the previous section. Let

$$\chi^{\leq}(x,a,b) = \big|\{t \mid t \in [a,b] \wedge f(t) \leq x\}\big|$$

denote the length of the portion of $[a,b]$ on which the value of f is at most x, let $\chi^{<}$ be defined analogously, and let

$$\psi(x,a,b) = \chi^{\leq}(x+h,a,b) - \chi^{<}(x,a,b). \tag{9}$$

For any point $(a,b) \in [\underline{a}, \overline{a}] \times [\underline{b}, \overline{b}]$, the corresponding segment $[a,b]$ is invalid if and only if $\max_x \psi(x,a,b) < (b-a)\rho$; that is, for all possible intervals $I = [x, x+h]$, the fraction of $[a,b]$ on which the function value lies in I is smaller than ρ.

Note that the candidate segment corresponding to (a,b) contains the segment defined by $(\overline{a}, \underline{b})$. Let ψ_C be the function ψ restricted to the cell C. We can rewrite ψ_C with respect to $\psi(x, \overline{a}, \underline{b})$ as follows. Within the cell, decreasing a by some value Δ_a corresponds to raising the function $\chi^{\leq}$ by Δ_a for all input values that are greater than or equal to $f(a)$; the function $\chi^{<}$ is affected in a similar manner. By the definition of ψ in Equation (9), this implies that ψ_C increases by Δ_a only on the interval $I_a = [f(a)-h,\ f(a)]$. Similarly, increasing b by Δ_b results in increasing ψ_C by Δ_b on $I_b = [f(b)-h,\ f(b)]$. Letting $\Delta_a = \overline{a} - a$ and $\Delta_b = b - \underline{b}$, we can rewrite ψ_C as follows:

$$\psi_C(x,a,b) = \begin{cases} \psi(x,\overline{a},\underline{b}) + \Delta_a + \Delta_b & \text{if } x \in I_a \cap I_b, \\ \psi(x,\overline{a},\underline{b}) + \Delta_a & \text{if } x \in I_a \setminus I_b, \\ \psi(x,\overline{a},\underline{b}) + \Delta_b & \text{if } x \in I_b \setminus I_a, \\ \psi(x,\overline{a},\underline{b}) & \text{otherwise.} \end{cases} \tag{10}$$

Therefore,

$$\max_x \psi_C(x,a,b) = \max \begin{cases} \max_{x \in I_a \cap I_b} \psi(x,\overline{a},\underline{b}) + \overline{a} - a + b - \underline{b} \\ \max_{x \in I_a \setminus I_b} \psi(x,\overline{a},\underline{b}) + \overline{a} - a \\ \max_{x \in I_b \setminus I_a} \psi(x,\overline{a},\underline{b}) + b - \underline{b} \\ \max_{x \notin (I_a \cup I_b)} \psi(x,\overline{a},\underline{b}), \end{cases} \tag{11}$$

so the condition $\max_x \psi(x,a,b) < (b-a)\rho$ is equivalent to the conjunction of four linear inequalities in a and b; hence, the forbidden space within C is the intersection of the resulting four half-planes. The four maxima on the right-hand side of Equation (11) are real numbers that depend only on a and b; if we refer to them as $\psi_1^*, \ldots, \psi_4^*$, respectively, we obtain conditions of the form in Equation (8). □

LEMMA 6.4. *The start-stop diagram can be computed in $O(n^2 \log n)$ time.*

PROOF. As in the discrete case, we can represent the function $\psi_{ij} = \psi(\cdot, \overline{a}, \underline{b})$ in a data structure, where $(\overline{a}, \underline{b})$ is the lower right corner of the current cell. Again, we maintain this data structure while traversing the cells of the grid bottom up within a column

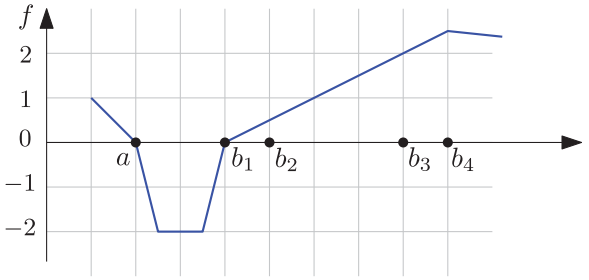


Fig. 15. A piecewise-linear attribute function that leads to tunnel cells.

and reinitialize it for every column. We use exactly the same data structure as before. By Lemma 6.3, the forbidden space in a cell is described by Equation (8). We now need four queries to compute the values of $\psi_1^*, \ldots, \psi_4^*$, since these are the maxima of the function ψ_{ij} on the intervals in Equation (10). Hence, we get the free space of a cell C. This means we can compute the start-stop diagram in $O(n^2 \log n)$ time. □

Clearly, Lemma 6.3 implies that the forbidden space within each cell of the start-stop diagram is verticonvex and has constant complexity, so we can invoke Theorem 3.2. We conclude:

THEOREM 6.5. *Given a piecewise-constant function f with n breakpoints, a threshold value $h > 0$, and a ratio $\rho \in [0, 1]$, we can compute a minimal segmentation for the condition that on a fraction of at least ρ of a segment, the difference between the maximum and minimum function value is at most h in $O(n^2 \log n + kn^2)$ time, where k is the size of a minimal segmentation.*

Piecewise-linear functions. If the attribute function f is piecewise linear, then the grid induced by the breakpoints of f may contain (many) tunnel cells. Assume that we want to segment the attribute function f depicted in Figure 15 on the outlier-tolerant criterion with $h = 2 + \varepsilon$, for some $0 \leq \varepsilon \leq 0.1$ and $\rho = 2/3$; that is, the difference between any two values is not more than approximately 2 but we can disregard 1/3 of the segment. For this criterion, the candidate segments $[a, b_2]$ and $[a, b_3]$ are valid, as well as any candidate segment $[a, b]$ for $b_1 < b < b_2$. However, there exist candidate segments $[a, b]$ for $b_2 < b < b_3$ and for $b_3 < b < b_4$ that are not valid. Since b_1, b_2, b_3, and b_4 lie on the same piece of the function f, this implies that the vertical line at a intersects the forbidden space in this cell twice, once below b_3 and once above b_3.

It is now easy to see that we can create more than one tunnel cell per row: we simply use the previous construction with points $a^- = a - \delta$ and $a^+ = a + \delta$, for some arbitrarily small δ, as the starting point of the segments. Since a is a breakpoint, these points lie on different pieces of f, and hence we get two tunnel cells. In both cases, the endpoints of the segments lie on the piece $[b_1, b_4]$, so the cells lie in the same row.

This example suggests that solving the problem for piecewise-linear attribute functions efficiently calls for a different approach.

6.2. Segmentation on the Standard Deviation Criterion

Another important nonmonotone criterion that we consider involves the standard deviation of an attribute function. In this section, we show how to compute a minimal segmentation of a piecewise-constant function f where each segment has standard deviation not exceeding a given threshold value. Let $\mu(a, b) = \int_a^b f(y)\,\mathrm{d}y/(b - a)$ denote

the mean value on a candidate segment $[a, b]$. The standard deviation $\sigma(a, b)$ is given by

$$\sigma(a,b) = \sqrt{\frac{\int_a^b (f(x) - \mu(a,b))^2 \, \mathrm{d}x}{b-a}}.$$

6.2.1. Discrete Case. In the discrete segmentation problem, we are allowed to partition f only where its value changes. Recall that f is given as a sequence $S = s_1, \ldots, s_n$ of pairs $s_i = (v_i, w_i)$, where v_i is the value of piece i and w_i is its weight.

LEMMA 6.6. *The start-stop matrix can be computed in $O(n^2)$ time.*

PROOF. To compute the start-stop matrix B, we need to test whether the standard deviation of a segment is below the allowed threshold. We could compute this in time linear in the length of a segment. However, we can also maintain the mean μ_{ij} and standard deviation σ_{ij} of a weighted sequence S_{ij}. We can then compute the mean $\mu_{i(j+1)}$ and the standard deviation $\sigma_{i(j+1)}$ for $S_{i(j+1)}$ in constant time from μ_{ij} and σ_{ij}. This implies that we can fill B in constant time per cell and thus quadratic time in total. □

Once we have B, we can compute a minimal segmentation in $O(n^2)$ time (Theorem 2.1). Hence:

THEOREM 6.7. *For any value $h > 0$, a minimal discrete segmentation where each segment has standard deviation at most h can be computed in $O(n^2)$ time.*

6.2.2. Continuous Case. In the continuous case, a and b can have any real value in I. Setting the standard deviation $\sigma(a, b)$ equal to the constant threshold value h, we obtain the functional description of the boundaries of the forbidden space in the start-stop diagram:

$$\sigma(a,b) = h \iff \int_a^b (f(x) - \mu(a,b))^2 \, \mathrm{d}x = (b-a) \cdot h^2.$$

Further algebraic manipulations, using the fact that f is piecewise constant, give a cubic expression in a and b. Hence, the boundaries of the forbidden space within each cell of the start-stop diagram are piecewise-cubic curves. This allows us to prove the following lemma:

LEMMA 6.8. *Every cell of the start-stop diagram is verticonvex.*

PROOF. On a vertical line, the start point of the candidate segment is fixed. Let it be $\tilde{a}$. Inside a single cell, the stop point has a single constant function value $f(b) = c$. Therefore, varying b implies including more or less of c in the values whose standard deviation is considered.

Imagine b is at its low end of the cell, and let it increase to its high end. Then the mean $\mu(\tilde{a}, b)$ tends monotonically toward c. There are four cases to distinguish:

(1) Initially $\sigma(\tilde{a}, b) < h$ and $|\mu(\tilde{a}, b) - c| < h$. Then the whole intersection of the cell and the vertical line is allowed.
(2) Initially $\sigma(\tilde{a}, b) > h$ and $|\mu(\tilde{a}, b) - c| > h$. Then the lower end of the intersection is forbidden, but possibly, at some value of b it becomes allowed.
(3) Initially $\sigma(\tilde{a}, b) < h$ and $|\mu(\tilde{a}, b) - c| > h$. Then the lower end of the intersection is allowed, but possibly, it becomes forbidden and possibly later allowed again.
(4) Initially $\sigma(\tilde{a}, b) > h$ and $|\mu(\tilde{a}, b) - c| < h$. Then the lower end of the intersection is forbidden, but possibly, at some value of b it becomes allowed.

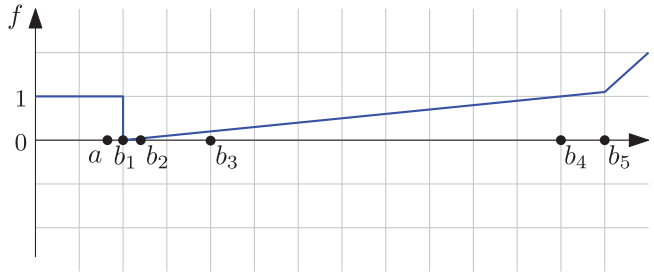


Fig. 16. A piecewise-linear attribute function that leads to tunnel cells.

In other words, we have the property that if for some b the candidate segment becomes allowed, then it stays allowed. This is true for the following reason. At the value $\tilde{b}$ of b when $[\tilde{a}, b]$ becomes allowed, we have $\sigma(\tilde{a}, \tilde{b}) = h$ and $|\mu(\tilde{a}, \tilde{b}) - c| < h$. For increasing b, the average $\mu(\tilde{a}, b)$ will tend monotonically toward c. So for larger b we have $|\mu(\tilde{a}, b) - c| < h$, and therefore $\sigma(\tilde{a}, b) < h$. This proves the lemma. □

LEMMA 6.9. *The start-stop diagram can be computed in $O(n^2)$ time.*

PROOF. The forbidden space in each cell has constant complexity, and given the description of σ for cell C, we can compute σ for neighbors of C in $O(1)$ time. □

Using Theorem 3.2, we then obtain:

THEOREM 6.10. *Given a piecewise-constant function f with n breakpoints and a threshold value $h > 0$, we can compute a minimal segmentation for the criterion that the standard deviation of a segment is at most h in $O(kn^2)$ time, where k is the size of a minimal segmentation.*

Piecewise-linear functions. As with the outlier-tolerant criterion, if f is piecewise linear, then the grid decomposition of the start-stop diagram induced by the breakpoints of f may contain (many) tunnel cells. Consider, for example, the function f depicted in Figure 16. For both the candidate segments $[a, b_2]$ and $[a, b_4]$, the mean is close to 0.5 and the standard deviation is close to 0.25. However, there exists a value $b_2 < b < b_4$, such that the mean and the standard deviation of $[a, b]$ are smaller than 0.25. For example, in Figure 16, the standard deviation at b_3 is below 0.2. So if we choose $h = 0.25$, the cell $[0, b_1] \times [b_1, b_5]$ is a tunnel cell: the vertical line at a intersects the forbidden space twice, once below b_3 and once above it. We can extend this example to construct two tunnel cells as before. Therefore, we do not expect that there exists a straightforward extension of our algorithm to this case.

7. LINE SIMPLIFICATION

Even though we designed our framework for trajectory segmentation, the approach is also applicable to different problems. In this section, we discuss its use for certain polygonal line simplification problems. Early line simplification algorithms try to reduce the number of vertices of a polygonal line while keeping its global shape by selecting a subset of the vertices [Douglas and Peucker 1911; Imai and Iri 1988]. A different approach to the problem is taken by line simplification methods that allow the vertices of the simplified polyline to lie anywhere [Guibas et al. 1993] and only require the shape to resemble the input curve. We propose a version of line simplification that compromises these two extremes: we restrict the vertices of the simplified polyline to lie on the input curve, but do not insist that they coincide with input vertices. A single edge of a simplified polyline is called a *shortcut* and it can replace a piece of the

original polyline, namely, the piece between the endpoints of the shortcut (which lie on the original polyline). We cast line simplification problems into our framework: a simplification is a staircase in the start-stop diagram and we can study the properties of its cells.

The most common criterion for line simplification is the Hausdorff distance. An edge of a simplification with its endpoints on the input polyline is valid if the Hausdorff distance between that edge and the polyline piece it replaces is at most a prespecified value. It is easy to verify that the resulting cells in the start-stop diagram can be tunnel cells, and there can be many in a single row. Hence, our approach does not give a polynomial-time algorithm for the simplification problem using the Hausdorff distance.

Next we consider the dilation criterion, where every edge of the simplification can be at most a given factor shorter than the polyline piece it replaces. Fix a starting point and an edge that contains the endpoint, and parameterize the position on the end edge. The function that gives the dilation expressed in this parameter is the ratio of a linear function and a hyperbolic function that does not grow faster than the linear function. It can easily be verified that the dilation measure gives verticonvex cells in the start-stop diagram.

Another criterion that we can handle easily is requiring that shortcuts cannot be shorter than a given threshold. This criterion also gives verticonvex cells in the start-stop diagram and we obtain the analogous result. However, the conjunction of the two criteria, dilation and nonshortness, does not give verticonvex cells, which is to be expected after the discussion of Section 4.

Now suppose that not only a polyline P but also a convex polygonal obstacle is given, and assume that they do not intersect. We are interested in the minimum-link simplification of P that also does not intersect the obstacle. Cells in the start-stop diagram are verticonvex, and we can apply our techniques to obtain a polynomial-time algorithm. If the obstacle has m vertices, the complexity of the free space in each cell is $O(m)$, and hence the running time of the algorithm will be $O(n^2m + n^2k)$.

Similarly, assume that a polyline in 3-dimensional space is given, together with a convex polyhedral obstacle. Again, cells are verticonvex, and we obtain the same running time as in the planar case due to the following:

LEMMA 7.1. *The complexity of a cell in the start-stop diagram is $O(m)$.*

PROOF. For any cell c, let e and e' be the edges on which the starting and ending points lie, respectively. For any point p on e, consider the triangle $\triangle(p, e')$ that p forms with edge e', and consider how it intersects with the obstacle $\mathcal{O}$; see Figure 17. Because $\mathcal{O}$ is convex, there are up to two rays from p inside $\triangle(p, e')$ to e' that are tangent to $\mathcal{O}$, and generally these tangencies involve edges of $\mathcal{O}$. Now consider p moving over e from one endpoint to the other, and consider one such tangency. As long as the edge of $\mathcal{O}$ remains the same, we get one curve in the cell c bounding the free space. The edge of $\mathcal{O}$ can change in two ways to a different edge, causing a different curve bounding c: (1) when the tangent encounters a vertex and (2) when the tangent aligns with a face of $\mathcal{O}$. It is easy to see that any vertex or face of $\mathcal{O}$ can cause at most one such event during the move of p along e. □

If the obstacle is not convex, then we can get tunnel cells and we do not obtain a polynomial-time algorithm. If we have more than one obstacle, then we can get tunnel cells as well. We summarize our results in the following theorem.

THEOREM 7.2. *Given a polygonal line P with n vertices, we can compute a minimum-link simplification P' of P where all k vertices of P' lie on P, and such that*

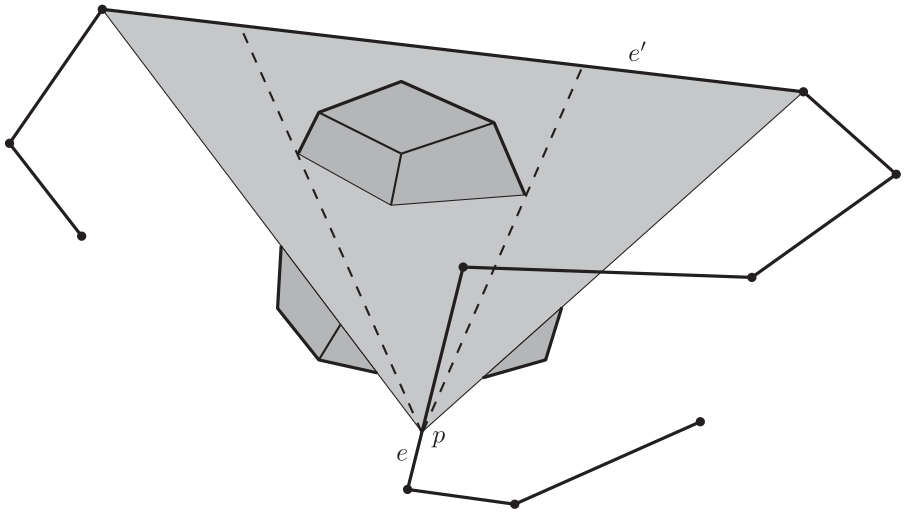


Fig. 17. The triangle formed by p and e' intersecting an obstacle, and the tangents from p to e'.

—*the dilation of each edge of P' is at most a given value δ, in $O(n^2k)$ time, if P lies in the plane;*
—*the length of each edge of P' is at least a given length γ, in $O(n^2k)$ time, if P lies in $\mathbb{R}^d$ (for any constant d); or*
—*the simplification P' does not intersect a given convex polyhedral obstacle $\mathcal{O}$, in $O(n^2m + n^2k)$ time, where m is the number of vertices in $\mathcal{O}$, if P and $\mathcal{O}$ lie in $\mathbb{R}^2$ or $\mathbb{R}^3$.*

8. CONCLUSIONS

In this article, we analyzed the problem of segmenting a trajectory for nonmonotone criteria. For monotone criteria, near-linear-time algorithms are known [Buchin et al. 2011]. We showed also that for certain nonmonotone criteria, polynomial-time solutions exist. In particular, our approach uses the start-stop diagram and we identify properties of this diagram that make the problem tractable. Furthermore, we proved that the abstract segmentation problem based on this diagram is NP-hard.

For two concrete nonmonotone criteria, we presented efficient algorithms to compute the start-stop diagram. We also showed that the resulting diagrams indeed have the properties that allow for efficient segmentation. As a result, we can compute an optimal segmentation of a trajectory based on the outlier-tolerant criterion in $O(n^2 \log n + kn^2)$ time, and on the standard deviation criterion in $O(kn^2)$ time, where n is the number of vertices of the input trajectory and k is the number of segments in an optimal solution. These two criteria are relevant in practice, since they are more robust against noise than related monotone criteria.

The question for a complete characterization of the start-stop diagram s that allow for efficient segmentation is still open. We answered this question partially, but more can perhaps be said. Furthermore, the analyses of the running times for the outlier-tolerant criterion and the standard deviation criterion are based on the assumption that the attribute function is piecewise constant. While many second-order attributes, such as speed and heading, are piecewise constant due to the linear interpolation of the trajectory, it would also be interesting to see whether similar results can be obtained for, for example, piecewise-linear functions. The examples given in Sections 6.1 and 6.2

seem to indicate that the methods developed in this article do not immediately apply if the attribute functions are piecewise linear.

Finally, our two-step approach of first computing the start-stop diagram explicitly and then solving the segmentation problem inherently requires at least $\Omega(n^2)$ time and space. An intriguing open problem is whether a subquadratic solution may be possible.

ACKNOWLEDGMENTS

The authors would like to thank Michael Kerber for interesting discussions about this article.

REFERENCES

H. Alt and M. Godau. 1995. Computing the Fréchet distance between two polygonal curves. *Int. Journal of Computer and Geometry Applications* 5 (1995), 75–91.

A. Anagnostopoulos, M. Vlachos, M. Hadjieleftheriou, E. J. Keogh, and P. S. Yu. 2006. Global distance-based segmentation of trajectories. In *Proc. 12th Conference on Knowledge Discovery and Data Mining*. 34–43.

P. Bovet and S. Benhamou. 1988. Spatial analysis of animals' movements using a correlated random walk model. *Journal of Theoretical Biology* 131, 4 (1988), 419–433. DOI:http://dx.doi.org/DOI:10.1016/S0022-5193(88)80038-9

M. Buchin, A. Driemel, M. J. van Kreveld, and V. Sacristan. 2011. Segmenting trajectories: A framework and algorithms using spatiotemporal criteria. *Journal of Spatial Information Science* 3, 1 (2011), 33–63.

C. Calenge, S. Dray, and M. Royer-Carenzi. 2009. The concept of animals' trajectories from a data analysis perspective. *Ecological Informatics* 4, 1 (2009), 34–41. DOI:http://dx.doi.org/10.1016/j.ecoinf.2008.10.002

W. S. Chan and F. Chin. 1992. Approximation of polygonal curves with minimum number of line segments. In *Proc. Algorithms and Computation (LNCS 650)*. 378–387.

P. Chundi and D. J. Rosenkrantz. 2009. Segmentation of time series data. In *Encyclopedia of Data Warehousing and Mining*, John Wang (Ed.). IGI Global, 1753–1758.

S. Dasgupta, C. Papadimitriou, and U. Vazirani. 2008. *Algorithms*. McGraw-Hill.

M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. 1997. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, Berlin.

S. Dodge, R. Weibel, and E. Forootan. 2009. Revealing the physics of movement: Comparing the similarity of movement characteristics of different types of moving objects. *Computers, Environment and Urban Systems* 33, 6 (2009), 419–434.

David H. Douglas and Thomas K. Peucker. 1911. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization* 10, 2 (1911), 112–122. DOI:http://dx.doi.org/10.3138/FM57-6770-U75U-7727

K. S. Fu and J. K. Mui. 1981. A survey on image segmentation. *Pattern Recognition* 13, 1 (1981), 3–16. DOI:http://dx.doi.org/10.1016/0031-3203(81)90028-5

L. J. Guibas, J. Hershberger, J. S. B. Mitchell, and J. Snoeyink. 1993. Approximating polygons and subdivisions with minimum link paths. *International Journal of Computer and Geometry Applications* 3, 4 (1993), 383–415.

E. Gurarie, R. D. Andrews, and K. L. Laidre. 2009. A novel method for identifying behavioural changes in animal movement data. *Ecology Letters* 12, 5 (2009), 395–408. DOI:http://dx.doi.org/10.1111/j.1461-0248.2009.01293.x

H. Imai and M. Iri. 1988. Polygonal approximation of curve-formulations and algorithms. *Computational Morphology: A Computational Geometric Approach to the Analysis of Form* (1988), 71–86.

X. Li, X. Li, D. Tang, and X. Xu. 2010. Deriving features of traffic flow around an intersection from trajectories of vehicles. In *Proc. 18th International Conference on Geoinformatics*. IEEE, 1–5.

R. Mann, A. D. Jepson, and T. El-Maraghi. 2002. Trajectory segmentation using dynamic programming. In *Proc. 16th International Conference on Pattern Recognition (ICPR'02)*, Vol. 1. 331–334.

R. Nathan, W. M. Getz, E. Revilla, M. Holyoak, R. Kadmon, D. Saltz, and P. E. Smouse. 2008. A movement ecology paradigm for unifying organismal movement research. *Proceedings of the National Academy of Sciences* 105 (2008), 19052–19059.

U. Ramer. 1972. An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics and Image Processing* 1, 3 (1972), 244–256. DOI:http://dx.doi.org/10.1016/S0146-664X(72)80017-0

A. Stohl. 1998. Computation, accuracy and applications of trajectories – A review and bibliography. *Atmospheric Environment* 32, 6 (1998), 947–966. DOI:http://dx.doi.org/10.1016/S1352-2310(97)00457-3

E. Terzi and P. Tsaparas. 2006. Efficient algorithms for sequence segmentation. In *Proc. 6th SIAM International Conference on Data Mining*. 314–325.

B. van Moorter, D. R. Visscher, C. L. Jerde, J. L. Frair, and E. H. Merrill. 2010. Identifying movement states from location data using cluster analysis. *Journal of Wildlife Management* 74, 3 (2010), 588–594. DOI:http://dx.doi.org/10.2193/2009-155

X. Zhou, S. Shekhar, P. Mohan, S. Liess, and P. K. Snyder. 2011. Discovering interesting sub-paths in spatiotemporal datasets: A summary of results. In *Proc. 19th ACM SIGSPATIAL Conference on Advances in Geographic Information Systems*. 44–53.

Received May 2013; revised March 2014; accepted July 2014