INSTITUTE OF
COMPUTER SCIENCE
FREIE UNIVERSITÄT BERLIN

Bachelor Thesis

Lexicographic Fréchet Matchings with Degenerate Inputs

Anton Irmfried Norbert Begehr
Matriculation-Nr: 5013449
a.begehr@fu-berlin.de

Supervisor: Prof. Dr. Günter Rote

Berlin, July 12, 2018

Abstract

The classical Fréchet distance is the minimal connecting distance needed to monotonically traverse two paths completely. Lexicographic Fréchet matchings utilizes steepest descent and applies the Fréchet distance recursively (i.e. divide and conquer) to minimize the time spent at distance greater than a threshold. We take a deep dive into degenerate inputs where different matchings appear equal when viewed at a distance. The problem of choosing one of multiple paths through critical events at the same distance is investigated. Multiple critical events are compared using derivatives and second derivatives.

Statutory Declaration

I declare that I have developed and written the enclosed Bachelors Thesis completely by myself, and have not used sources or means without declaration in the text. Any thoughts from others or literal quotations are clearly marked. The Bachelors Thesis was not used in the same or in a similar version to achieve an academic grading or is being published elsewhere.

Anton Begehr
Berlin, July 12, 2018

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Problem Metaphor (Dog Walking) . . . 1

2 Preliminaries **2**
- 2.1 Polygonal Chains . . . 2
- 2.2 Parametrisation . . . 2
- 2.3 Height Function . . . 3

3 Classical Fréchet Distance **3**
- 3.1 Traversal of P and Q . . . 4
- 3.2 Free-Space Diagram . . . 5
- 3.3 Decision Problem . . . 5
- 3.4 Classical Critical Events . . . 6
- 3.5 Computing the Classical Fréchet Distance . . . 7

4 Lexicographic Fréchet Matchings **7**
- 4.1 Steepest Descent . . . 7
- 4.2 New Lexicographic Type of Critical Event . . . 8
- 4.3 Lexicographic Fréchet Matching Algorithm . . . 8

5 Lexicographic Fréchet Distance with Degenerate Inputs **10**
- 5.1 Problem . . . 10
- 5.2 Examples . . . 11
  - 5.2.1 Minimal Example: Traverse Both . . . 11
  - 5.2.2 Minimal Example: Traverse Either . . . 12
  - 5.2.3 Minimal Example: Traverse One . . . 12
  - 5.2.4 More Complex Example 1 . . . 13
  - 5.2.5 More Complex Example 2 . . . 14
- 5.3 Possible Paths Through Critical Events . . . 14
  - 5.3.1 Generate Traversability Graph . . . 14
  - 5.3.2 Can C_{k1} reach C_{k2}? – Reachability Intervals . . . 15
  - 5.3.3 Can C_{k1} reach C_{k2}? – Minimum x-/y-Coordinates . . . 17
- 5.4 Traversal Cross-Section $f(t)$ and Profile $\hat{f}(s)$ . . . 19
  - 5.4.1 Hyperbolas . . . 19
  - 5.4.2 Cross-Section $f(t)$ . . . 19
  - 5.4.3 Profile $\hat{f}$ . . . 20
- 5.5 Multiple Ascents to and Descents from ϵ_0 . . . 22
  - 5.5.1 Simplifications and First Derivative . . . 22

5.5.2 Second Derivative 25
5.5.3 Example: First Derivative 26
5.5.4 Example: Second Derivative 27

6 Concluding Perspective **31**
6.1 Third, Fourth, Fifth, etc. Derivative 32
6.2 Equal Descents at ϵ_0, Unequal Descents Below 33

A Appendices **35**
A.1 Fréchet Web App 35
A.1.1 Development 35
A.1.2 Live Version and Links 35
A.1.3 Source Control 35
A.1.4 Architecture 35
A.1.5 Features 36

1 Introduction

1.1 Motivation

The classical Fréchet distance is a measure of dissimilarity between two curves. In comparison to the root-mean-square error (RMSE), the Fréchet distance has a longer runtime, but therefore it takes into account the position and ordering of points. This is important for a wide range of use-cases: Comparing molecule chains like proteins, handwriting recognition, route comparison, computer vision, and WiFi access point planning being just a few of the real world use-cases of the Fréchet distance.

Lexicographical Fréchet matchings extend the capabilities of the classical Fréchet distance by not just finding one global Fréchet distance for two input paths, it also applies the algorithm for all local sub-traversals to attain a lexicographically optimized traversal.

The goal of this paper is to introduce both the classical Fréchet distance and lexicographical Fréchet matchings; and ultimately to investigate degenerate inputs for lexicographical Fréchet matchings, as revealed by Rote in section 7 of his 2014 paper "Lexicographic Fréchet Matchings"[2].

1.2 Problem Metaphor (Dog Walking)

The Fréchet distance is often visually explained using the "dog walking"-metaphor, which is described in the following paragraphs.

Imagine two curves:

- Curve C_P tracing the walking curve of a man
- Curve C_Q tracing the walking curve of the man's dog

The Fréchet Distance then is the shortest possible length of a leash, the man would need to keep ahold of his dog. Both the man and his dog have to traverse their entire paths and neither are allowed to walk backwards.

Both the man and his dog can independently vary their respective speeds. With the conditions described above, at every point in time, either both the man and his dog are walking forwards along their respective paths or one is idle while the other is walking along his path.

To make calculating the Fréchet distance more feasible by simplification, the arbitrary curves C_P on which the man is walking, and C_Q on which his dog is walking, are each approximated by a connected chain of line segments, a polygonal chain. The arbitrary curve C_P is approximated by the polygonal chain P. The arbitrary curve C_Q is approximated by the polygonal chain Q.

2 Preliminaries

2.1 Polygonal Chains

Given are two polygonal chains P and Q. A polygonal chain is defined by a list of points, sorted by the order of the chain, ascending from beginning to end. The first point of the list is the start of the polygonal curve and the last point of the list is the end of the polygonal curve. This order implies the direction of the directional polygonal curve.

P and Q are polygonal chains consisting of n and m points respectively.

$$P_{points} = [P_1, P_2, \ldots, P_n]$$
$$Q_{points} = [Q_1, Q_2, \ldots, Q_m]$$

Points of P and Q are expressed as vectors:

$$P_i = \begin{pmatrix} x_{Pi} \\ y_{Pi} \end{pmatrix}, Q_j = \begin{pmatrix} x_{Qj} \\ y_{Qj} \end{pmatrix}$$

For simplification we assume all points P_i of P and Q_j of Q lie in the xy-plane, but the results generalize easily to arbitrary dimensions[2].

2.2 Parametrisation

We will use an arc-length parametrisation to define the polygonal curves P and Q going forward. We are using the arc-length parametrisation, instead of an unit-length parameter, to achieve visualizations with proportional scaled cells in the further course of this examination.

First we calculate the Euclidean distances between the points of P and Q in the order they appear:

$$d_{Pi} = \|P_{i+1} - P_i\|, 1 \leq i \leq n-1$$
$$d_{Qj} = \|Q_{j+1} - Q_j\|, 1 \leq j \leq m-1$$

We also calculate the distances L_{Pi} and L_{Qi}, that define the distance from beginning of P and Q to the point P_i and Q_j on the P and Q respectively:

$$L_{Pi} = \sum_{k=1}^{i} d_{Pk}, 1 \leq i \leq n-1$$

$$L_{Qi} = \sum_{k=1}^{j} d_{Qk}, 1 \leq j \leq m-1$$

The total lengths of P and Q are then defined as $L_P = L_{Pn-1}$ and $L_Q = L_{Qm-1}$.

We apply an arc-length parametrisation on P and Q using parameters $x \in [0, L_P]$ and $y \in [0, L_Q]$ respectively. These parametrisation can be expressed as follows:

$$P(x) = \begin{cases} P_1 + \frac{x}{d_{P1}}(P_2 - P_1) & 0 \leq x \leq L_{P1} \\ P_2 + \frac{x - L_{P1}}{d_{P2}}(P_3 - P_2) & L_{P1} < x \leq L_{P2} \\ \dots & \\ P_{n-2} + \frac{x - L_{Pn-2}}{d_{Pn-1}}(P_{n-1} - P_{n-2}) & L_{Pn-2} < x \leq L_{Pn-1} \end{cases}$$

$$Q(y) = \begin{cases} Q_1 + \frac{y}{d_{Q1}}(Q_2 - Q_1) & 0 \leq y \leq L_{Q1} \\ Q_2 + \frac{y - L_{Q1}}{d_{Q2}}(Q_3 - Q_2) & L_{Q1} < x \leq L_{Q2} \\ \dots & \\ Q_{m-2} + \frac{x - L_{Qm-2}}{d_{Qm-1}}(Q_{m-1} - Q_{m-2}) & L_{Qm-2} < x \leq L_{Qm-1} \end{cases}$$

2.3 Height Function

To calculate all possible distances between points on P and Q, we will define the height function δ that maps each point pair R consisting of one point on P and one point on Q to the Euclidean distance between the points.

$$R = [0, L_P] \times [0, L_Q]$$
$$(x, y) \in R$$
$$\delta(x, y) := \|P(x) - Q(y)\|$$

3 Classical Fréchet Distance

This section introduces the Fréchet distance and gives an overview of the main findings in the 1995 paper "Computing the Fréchet distance between two polygonal curves" by Alt and Godau[1] which later will be essential for computing the lexicographic Fréchet distance.

3.1 Traversal of P and Q

Let us revisit the "dog walking"-metaphor from section 1.2 of the man and his dog walking along or in other words traversing their paths. The Fréchet distance is a dissimilarity measure that considers the traversal of both polygonal chains P and Q with following conditions:

(A) Both P and Q are traversed completely.

(B) Both P and Q are traversed monotonically ascending in the direction of P and Q respectively. Meaning, a traverser on either P or Q can stand idle and go forwards, but may never go backwards.

We traverse both P and Q together. This means that at every point in time one of the following cases holds true:

- Both the traverser on P and the traverser on Q are moving.
- The traverser on P is moving and the traverser on Q is standing idle.
- The traverser on Q is moving and the traverser on P is standing idle.
- Neither the traverser on P, nor the traverser on Q is moving.

The last case is identical to the end of both traversals. Both traverses have reached the end of their polygonal chains.

We define a traversal of both P and Q together as a joint parametrisation $(\alpha(t), \beta(t))$. This joint parametrisation represents a continuous curve in the parameter area R. The curve represented by $(\alpha(t), \beta(t))$ is a monotonic curve, because of above condition (B) which states that both P and Q are traversed only forwards and never backwards.

Revisiting the "dog walking"-metaphor from Section 1.2, the Fréchet distance is the length of the shortest leash with which it is possible for the man and his dog to traverse their paths completely by only travelling forward. In other words, the Fréchet distance is defined as the largest distance between the traverser on path P and the traverser on path Q of the traversal with the smallest maximum distance between the respective traversers.

Using the joint parametrisation $(\alpha(t), \beta(t))$ as representative for a traversal, next the distance function $f(t)$ for the joint parametrisation needs to be defined to determine the Fréchet distance. The height function defined in Section 2.3 will be used to define the distance function $f(t)$ for a joint parametrisation $(\alpha(t), \beta(t))$:

$$f(t) = \delta(\alpha(t), \beta(t)) := \|P(\alpha(t)) - Q(\beta(t))\|$$

Now, the Fréchet distance of P and Q is the largest value of $f(t)$ for the joint parameterisations $(\alpha(t), \beta(t))$ with the smallest maximum of all possible joint parameterisations.

3.2 Free-Space Diagram

The free-space diagram F_ϵ is the set of all points of R that lie below the given height ϵ:

$$(x, y) \in R$$

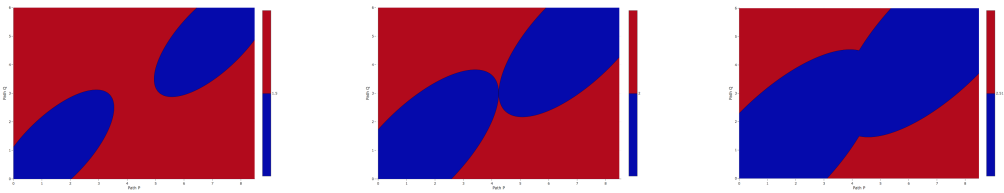
$$F_\epsilon = \{(x, y) \mid \|P(x) - Q(y)\| \leq \epsilon\}$$

The fundamental insight of Alt and Godau[1] is that the Fréchet distance of P and Q is the smallest ϵ for which a monotonic path exists from $(0, 0)$ to (L_P, L_Q) in F_ϵ.[2]

3.3 Decision Problem

The decision problem asks, if a height function δ can be traversed monotonically while not exceeding the height of a given ϵ. In their paper, Alt and Godau[1] define the decision algorithm which solves the decision problem by using the free-space diagram F_ϵ.

In the following example we will see three free-space diagrams F_ϵ for the same height function with different ϵ. ϵ_F denotes the Fréchet distance.



(a) $\epsilon < \epsilon_F$ (b) $\epsilon = \epsilon_F$ (c) $\epsilon > \epsilon_F$

Figure 1: Three free-space diagrams with same height function δ[1]

The following cases are depicted in Figure 1 above:

(a) ϵ is smaller than the Fréchet distance ϵ_F. The decision algorithm returns false.

[1]View the example here: `https://abegehr.github.io/frechet/?p=(1_2)(4_5)(7_2)&q=(1_3)(7_3)`

(b) ϵ is equal to the Fréchet distance ϵ_F. The decision algorithm returns true.

(c) ϵ is larger than the Fréchet distance ϵ_F. The decision algorithm returns true.

3.4 Classical Critical Events

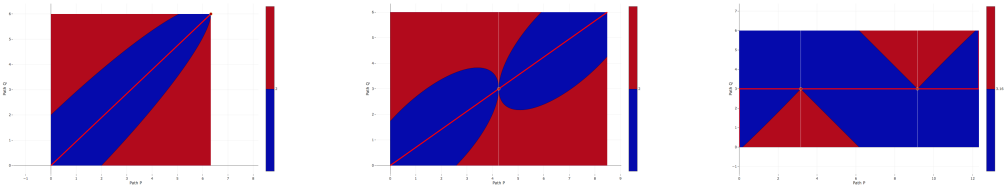
Alt and Godau also identified that the critical height ϵ_F where a critical pathway in the free-space diagram closes occurs in one of three cases. These are the three types of classical critical events:

(a) ϵ_F at $(0, 0) \in F_\epsilon$ or $(L_P, L_Q) \in F_\epsilon$

(b) A new passage with height ϵ_F opens on the border between two cells.

(c) A new horizontal or vertical passage with height ϵ_F opens on borders with cells in between.

In Figure 2 example free-space diagrams for each type of classical critical event are shown.



(a) type a[2] (b) type b[3] (c) type c[4]

Figure 2: Examples for the three types of classical critical events

Critical events will be denoted with C_k, where $C_k.A$ is the starting-point and $C_k.B$ is the end-point. For classical critical events of type a or b the starting-point and end-point are the same.

[2]Classical critical event of type a at height ϵ_F is at the upper-right corner (L_P, L_Q). View the example here: `https://abegehr.github.io/frechet/?p=(1_3)(7_5)&q=(1_3)(7_3)`

[3]Classical critical event of type b at height ϵ_F is at vertical border connecting the left to the right cell. View the example here: `https://abegehr.github.io/frechet/?p=(1_2)(4_5)(7_2)&q=(1_3)(7_3)`

[4]Classical critical event of type c at height ϵ_F is at vertical border connecting the left to the right cell over the middle cell. View the example here: `https://abegehr.github.io/frechet/?p=(4_3)(7_4)(1_4)(4_3)&q=(1_3)(7_3)`

3.5 Computing the Classical Fréchet Distance

We now have the tools to compute the classical Fréchet distance. To compute the classical Fréchet distance, we can apply *Algorithm 2* of Alt and Godau's paper.[1] The algorithm first determines all classical critical events of type a, b, and c, including their critical ϵ. Then using a binary search and the decision algorithm *Algorithm 2* finds the smallest critical ϵ for which the height function δ is traversable.

4 Lexicographic Fréchet Matchings

In his 2014 paper "Lexicographic Fréchet Matchings", Rote extended upon the findings from Alt and Godau in their paper "Computing the Fréchet distance between two polygonal curves" to establish an algorithm that produces a lexicographic Fréchet matching.

Taking a lexicographic approach roughly means that we want to minimize the time $T(s)$ during which the height exceeds a threshold s.[2] We want to spend as little time as possible at the current height and want to descend as quickly as possible.

4.1 Steepest Descent

To achieve a lexicographically optimized traversal, Rote makes use of the steepest descent. Therefore he developed a steepest descent algorithm.

Because we chose our input to be two paths consisting of straight line segments, the level set in a cell consists of concentric ellipses. Rote defines two lines for every cell, l and l':

(l) The points where the ellipses have vertical tangents and δ has a linear horizontal gradient lie on line l through the common center.[2]

(l') The points where the ellipses have horizontal tangents and δ has a linear vertical gradient lie on line l through the common center.[2]

Depending on whether we are descending from a point on one of the lines or from in between one of the quadrants they span up, the direction of the steepest descent is affected. Consult Rote's paper "Lexicographic Fréchet Matchings"[2] (Figure 5) for more detail. It is important to note at this point that the steepest descent for a point is unique.

4.2 New Lexicographic Type of Critical Event

Rote also identifies a new lexicographic type of critical event which does not influence the global Fréchet distance ϵ_F, but is essential for achieving a lexicographically correct traversal.

The new lexicographic type of critical event occurs while descending on a steepest descent path. It occurs when we would horizontally or vertically surpass a critical opening of the height at which we currently are.

Figure 3 shows a minimal example of these new lexicographic types of critical events.

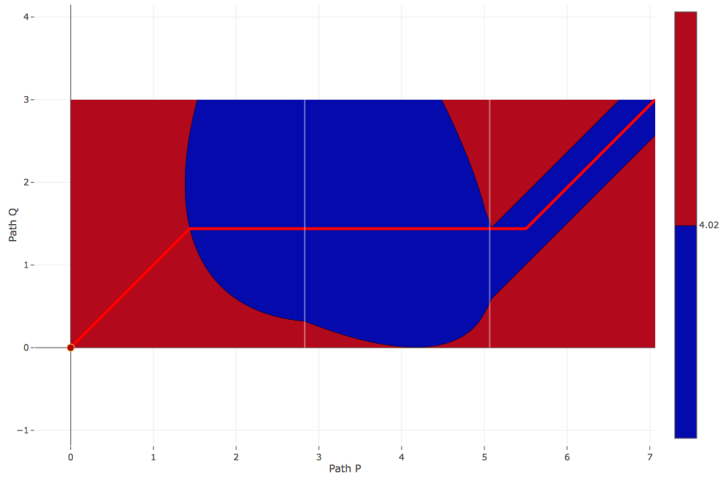


Figure 3: Example of new lexicographic type of critical event[5]

4.3 Lexicographic Fréchet Matching Algorithm

We now have the tools needed to understand Rote's algorithm for lexicographic Fréchet matchings[2] under *general inputs* (not yet regarding degenerate inputs, as we'll discuss in section 5). In contrary to the algorithm for computing the classical Fréchet distance, the algorithm for lexicographic Fréchet matching utilizes recursion to result in a local traversal that is lexicographically optimized. The algorithm as described here is aligned to the implementation of the online visualization.

[5]New lexicographic critical event of type a below height ϵ_F connecting the middle of the left cell to the vertical border between the middle and the right cell. View the example here: `https://abegehr.github.io/frechet/?p=(3_2)(5_4)(3_3)(5_3)&q=(2_7)(5_7)`

Algorithm 1 Requirements for Lexicographic Fréchet Matching Algorithm

function DecisionAlgorithm(δ, A, B, ϵ)
▷ Implementation of the decision algorithm that decides if δ is traversable from A to B with ϵ.
    return *true* or $false$

function SteepestDescent($\delta, A, B, \epsilon_A, \epsilon_B$)
▷ Attempts a steepest descent as far as possible for the higher of A and B, or both in case they reside at equal height. Attempt the steepest descent to A' and B' until A' or B' reaches a cell border or a line l or l', they would pass each other vertically or horizontally, or they meet each other. Descend to an equal height if possible.
    return $A', B', \epsilon_{A'}, \epsilon_{B'}$

function ReverseDescent($\delta, A, B, \epsilon_A, \epsilon_B$)
▷ Reverses steepest descent until a classical critical event of type b or c, or a new lexicographic critical event is found. Also returns the critical event c.
    return $A', B', \epsilon_{A'}, \epsilon_{B'}, c$

function ClassicalFréchet($\delta, A, B, \epsilon_A, \epsilon_B$)
▷ Using a binary search and the decision algorithm, this function finds the lowest traversable classical critical event that is needed to traverse δ from A to B. This is the same as the algorithm for the classical Fréchet distance.
    return ϵ_F, c

Algorithm 2 Lexicographic Fréchet Matching Algorithm

```
1: function TraverseRecursive(δ, A, B, ε_A, ε_B)
2:     if A = B or B.x ≤ A.x or B.y ≤ A.y then
3:                                          ▷ Traversal done. Connect A and B.
4:         return Connect(A, B)
5:     ε ← max(ε_A, ε_B)
6:     if DecisionAlgorithm(δ, A, B, ε) then
7:         A', B', ε_A', ε_B' ← SteepestDescent(δ, A, B, ε_A, ε_B)
8:         ε' ← max(ε_A', ε_B')
9:         if DecisionAlgorithm(δ, A', B', ε') then
10:                                         ▷ Traversable after steepest descent.
11:            return TraverseRecursive(δ, A', B', ε_A', ε_B')
12:        else        ▷ Not traversable. The descent passes a critical event.
13:            while A ≠ A' or B ≠ B' do
14:                A', B', ε_A', ε_B', c ← ReverseDescent(δ, A', B', ε_A', ε_B')
15:                ε' ← max(ε_A', ε_B')
16:                if DecisionAlgorithm(δ, A', B', ε') then
17:                                         ▷ Traversable with the critical event c.
18:                    return TraverseRecursive(δ, A', c.B, ε_A', ε_c) + c +
   TraverseRecursive(δ, c.A, B', ε_c, ε_B')
19:    else                ▷ A critical event with ε_c > ε is needed to traverse.
20:        ε_F, c ← ClassicalFréchet(δ, A, B, ε_A, ε_B)
21:        return TraverseRecursive(δ, A', c.B, ε_A', ε_c) + c + Tra-
   verseRecursive(δ, c.A, B', ε_c, ε_B')
```

5 Lexicographic Fréchet Distance with Degenerate Inputs

5.1 Problem

As Rote describes in section 7 of his paper "Lexicographic Fréchet Matchings" and as we hinted to in section 4.3 of this paper, multiple critical events can occur for the same ϵ.

Multiple critical events for the same ϵ turn out to be problematic, because as can be seen in algorithm 2, we assume the functions defined in algorithm 1 return only one critical event which then will be traversed. This assumption does not hold in all cases.

This section aims to answer the question of how to handle multiple critical events with the same critical ϵ by visualizing the problem and postulating

an algorithm that decides which combination of critical events result in a lexicographic traversal. The basis of this section is section 7 of Rote's paper "Lexicographic Fréchet Matchings" [2].

5.2 Examples

In this subsection, four examples will be discussed, for which multiple critical events with the same critical ϵ require a decision on which of them should be traversed to result in a lexicographic traversal. Instead of providing in-depth detail on solutions, this subsection aims to illustrate the problem through examples.

5.2.1 Minimal Example: Traverse Both

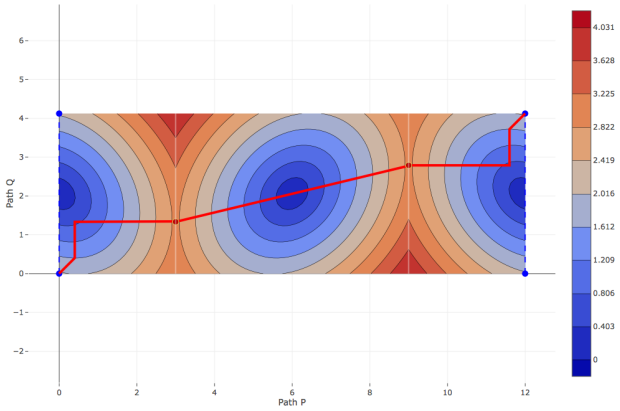


Figure 4: One path over two critical events

Figure 4 shows a minimal example with two critical events with the same ϵ. There are two classical critical events of type b at $\epsilon \approx 2.91$ at the two vertical cell-borders. Both critical events need to be traversed.

View the example from figure 4 here: `https://abegehr.github.io/frechet/?p=(5_5)(8_5)(2_5)(5_5)&q=(5.5_3)(4.5_7)`

5.2.2 Minimal Example: Traverse Either

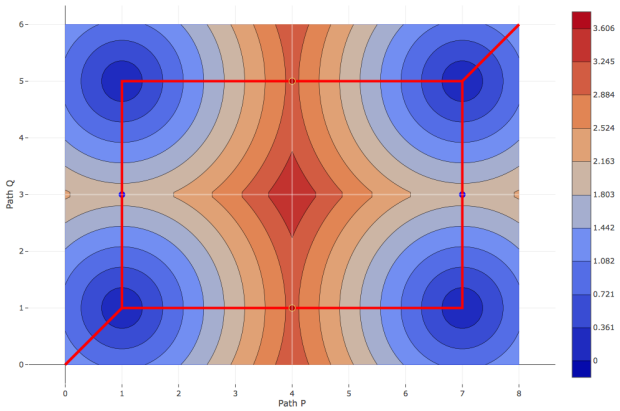


Figure 5: Two equivalent paths

Figure 5 shows a minimal example of multiple critical events with the same ϵ. There are four critical events to consider. Two classical critical events of type b with equal $\epsilon = 3$ at the two vertical cell-borders. And two classical critical events of type b with equal $\epsilon = 2$ at the two horizontal cell-borders. The decision problem shows that $\epsilon = 3$ is needed to traverse this height function δ. Either one of the critical events at $\epsilon = 3$ needs to be traversed.

View the example from figure 5 here: `https://abegehr.github.io/frechet/?p=(4_5)(8_5)(4_5)&q=(5_4)(5_7)(5_4)`

5.2.3 Minimal Example: Traverse One

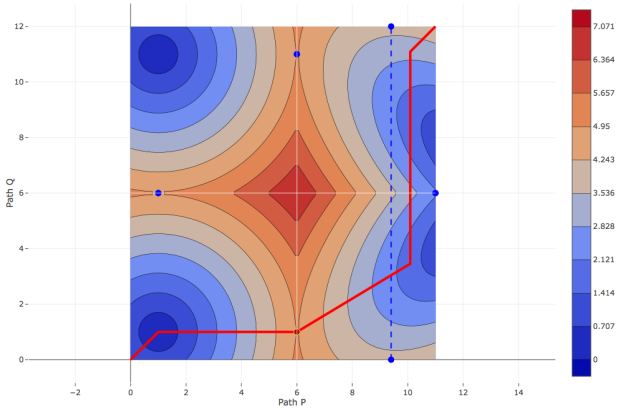


Figure 6: Two inequivalent paths

Figure 6 shows three classical critical events of type b at $\epsilon = 5$ to consider: $C_1(6|1)$, $C_2(1|6)$, and $C_3(6|11)$. There are two possible paths through these three critical events: traversing just C_1 or traversing C_2 and C_3. Since the ascends to and descends from C_1, C_2, and C_3 are similar, we choose the path over only C_1.

View the example from figure 6 here: `https://abegehr.github.io/frechet/?p=(3_2)(3_8)(6_4)&q=(2_3)(8_3)(2_3)`

5.2.4 More Complex Example 1

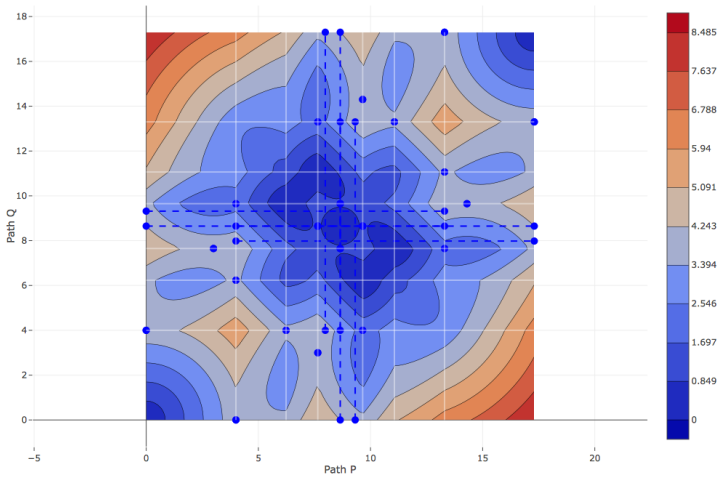


Figure 7: More complex example with eight critical events

Figure 7 shows a more complex example of multiple critical events with the same ϵ. There are eight classical critical events of type b with $\epsilon_0 = 4$: $C_1(4, 0)$, $C_2(0, 4)$, $C_3(7.65, 3)$, $C_4(3, 7.65)$, $C_5(14.30, 9.65)$, $C_6(9.65, 14.3)$, $C_7(17.3, 13.3)$, and $C_8(13.3, 17.3)$. The coordinates are rounded to two decimal places.

View the example from figure 7 to see the decision on which of the eight critical events to traverse here: `https://abegehr.github.io/frechet/?p=(0_0)(4_0)(3_2)(4_3)(2_3)(3_4)(2_6)(6_6)&q=(0_0)(0_4)(2_3)(3_4)(3_2)(4_3)(6_2)(6_6)`

5.2.5 More Complex Example 2

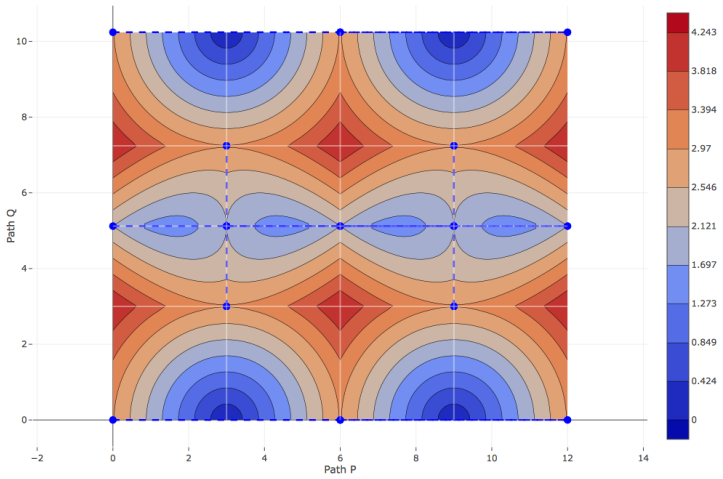


Figure 8: More complex example with sixteen critical events

Figure 8 shows a more complex example of multiple critical events with the same ϵ. There are sixteen classical critical events: ten of type b and six of type b. Five critical events are on the bottom side and five are on the top side of δ. Six critical events are on the center row of cells. There are two classical critical events of type c that lye on vertical cell borders.

View the example from figure 8 to see the decision on which of the sixteen critical events to traverse here: `https://abegehr.github.io/frechet/?p=(3_4)(6_4)(3_4)(6_4)(3_4)&q=(6_4)(6_7)(4.5_5.5)(6_7)(6_4)`

5.3 Possible Paths Through Critical Events

We have seen in the examples that in some cases we need to traverse a path through multiple of the critical events at equal height ϵ_0, while in other cases a path through one critical event is sufficient. To decide which path to traverse, first, we need to find all possible paths through the critical events.

5.3.1 Generate Traversability Graph

The possible paths through multiple critical events can be represented as a graph. The start node represents the lower left corner A and the end node represents the upper right corner B of the current δ subsection. Each critical event $C_k \in C$ of height ϵ_0 is a node. Edges are directional and denote traversability without another critical event of height ϵ_0 in between.

To generate this graph, first, we use the free-space diagram represented by L_{ij}^F and B_{ij}^F and the decision algorithm (see Alt and Godau [1]) to generate the reachable border bounds L_{ij}^R and B_{ij}^R of all cells. Then we use the reachable border bounds and the critical events C at height ϵ_0 to generate for each vertical and horizontal cell-border which subset of critical events C can reach the border. We call the subset of C that reaches the left and bottom border of cell (i, j): L_{ij}^C and B_{ij}^C respectively. After computing each L_{ij}^C and B_{ij}^C, we then connect critical events in L_{ij}^C and B_{ij}^C to reachable critical events starting on the top and right border.

See the algorithm 3 for more detail on how we generate the traversability graph of multiple critical traversals C_k that are at same height ϵ_0. p and q is the number of cells between the starting-point A and the ending-point b.

Algorithm 3 Generate Traversability Graph of multiple critical events at ϵ_0

1: **function** GENERATETRAVERSALGRAPH(A, B, C, ϵ_0)
2: ▷ A is start-point of traversal. B is ending-point of traversal. C holds all critical events C_k.
3: Add A and B as critical events to C.
4: $\forall i, j : L_{ij}^C := \{\} \wedge B_{ij}^C := \{\}$
5: $L^R, B^R \leftarrow$ DECISIONALGORITHM'(A, B, ϵ_0)
6: **for** $C_k \in C$ **do**
7: Add $C_k.B$ (ending-points) to L_{ij}^C or B_{ij}^C where $C_k.B$ lies.
8: **for** $i := 0$ to p **do** determine $L_{i,0}^C$
9: **for** $j := 0$ to q **do** determine $B_{0,j}^C$
10: **for** $i := 0$ to p **do**
11: **for** $j := 0$ to q **do**
12: **for** C_{k2} that starts on right or top border **do**
13: **for** $C_{k1} \in L_{i,j}^C \cup B_{i,j}^C$ **do**
14: **if** C_{k1} can reach C_{k2} **then**
15: Add edge (C_k1, C_k2) to graph.
16: construct $L_{i+1,j}^C$ and $B_{i,j+1}^C$ from L_{ij}^C, B_{ij}^C, $L_{i+1,j}^R$, $B_{i,j+1}^R$
return graph

5.3.2 Can C_{k1} reach C_{k2}? – Reachability Intervals

In the above algorithm 3 on line 14, we seek to determine if the critical event C_{k1} can reach the critical event C_{k2}. It is known that C_{k1} can reach the left or bottom border of the current cell (i, j), because $C_{k1} \in L_{i,j}^C \cup B_{i,j}^C$. It is also know that C_{k2} starts at the top or right border of the current cell.

We discard the potential edge (C_{k1}, C_{k2}) in the trivial case, where C_{k2} is not monotonically reachable from C_{k1}: $\neg(C_{k1}.B.x \leq C_{k2}.A.x \wedge C_{k1}.B.y \leq C_{k2}.A.y)$.

To determine if C_{k2} is reachable from C_{k1}, we first look at if C_{k1} comes from the left or bottom border and if C_{k2} starts on the right or top border. There are four cases to consider:

1. $C_{k1} \in L^C_{i,j}$ enters on left border and C_{k2} starts on right border.
2. $C_{k1} \in L^C_{i,j}$ enters on left border and C_{k2} starts on top border.
3. $C_{k1} \in B^C_{i,j}$ enters on bottom border and C_{k2} starts on right border.
4. $C_{k1} \in B^C_{i,j}$ enters on bottom border and C_{k2} starts on top border.

Reachability intervals I_{ij} of right and top borders can be computed as follows:

1. From left to right: $I^{l\to r}_{ij} = L^R_{i,j} \cap L^R_{i+1,j}$
2. From left to top: If $L^R_{i,j}$ is open: $I^{l\to t}_{ij} = B^R_{i,j+1}$. Otherwise closed.
3. From bottom to right: If $B^R_{i,j}$ is open: $I^{b\to r}_{ij} = L^R_{i+1,j}$. Otherwise closed.
4. From bottom to top: $I^{b\to t}_{ij} = B^R_{i,j} \cap B^R_{i,j+1}$

The superscript denotes the sides of the cell. $I^{b\to t}_{ij}$ for example represents the reachability interval of the top border for traversals coming from the bottom border.

It can be considered to use solely these reachability intervals I_{ij} to decide if C_{k2} can be reached from C_{k1}. One would simply check if C_{k2} lies in the reachability intervals from the side on which C_{k1} lies. This is not sufficient, as the example visualized in figure 9 shows.

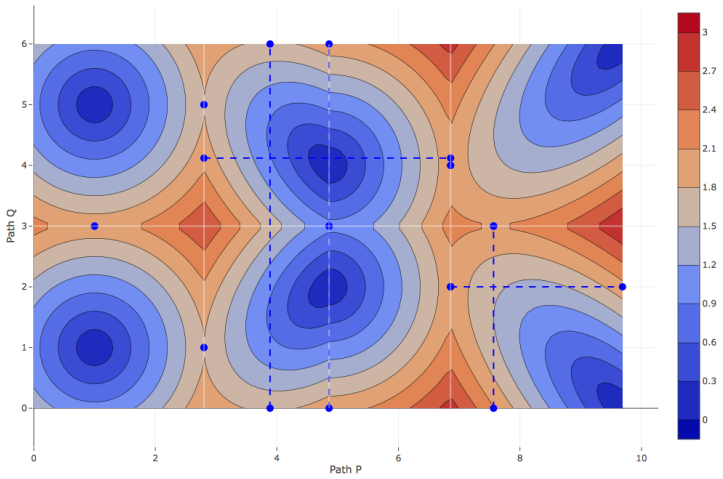


Figure 9: Considering solely reachability intervals is not sufficient[6]

In figure 9 the decision algorithm tells us that the classical Fréchet distance is $\epsilon_F = 2$. There are two critical events of type b with $\epsilon = 2$ that we consider: $C_1(3, 1)$ and $C_2(\sim 6.86, 4)$. If we were to use solely the reachability intervals to check if C_{k2} is reachable from C_{k1} as described above, the generated graph would find an edge between C_1 and C_2, even though there exists no such path, because it is closed by the right border of cell $(0, 1)$: $L^R_{1,1}$. Using solely the reachability intervals, C_2 would still appear reachable from C_1, because $L^R_{2,1}$ starts below $L^R_{1,1}$, since $L^R_{2,1}$ can also be reached from $B^R_{1,1}$, and not just from $L^R_{1,1}$. We can now learn from this counter example and apply better measures.

5.3.3 Can C_{k1} reach C_{k2}? – Minimum x-/y-Coordinates

To solve the problem of reachability intervals for generating a traversability graph of critical events, we need to memorize for each critical event not just that it can reach border L_{ij} or B_{ij}, but also for vertical borders the minimum y-coordinate and for horizontal borders the minimum x-coordinate that it needs to reach the border, as these can differ for different critical events that both reach a border. Then, when we connect a critical event C_{k1} to C_{k2}, we test the reachability intervals and we determine if the C_{k2} lies on or above the minimum x- or y-coordinate that C_{k1} needs to reach the border in question.

We can now define two helper algorithms in more detail. First, the algorithm for determining if C_{k1} can reach C_{k2}. And second, the algorithm that constructs $L^C_{i+1,j}$ and $B^C_{i,j+1}$ from L^C_{ij}, B^C_{ij}, $L^R_{i+1,j}$, $B^R_{i,j+1}$.

[6]View the example here: `https://abegehr.github.io/frechet/?p=(5_5)(5_7.8)(4_6)(4_8)(6_6)&q=(6_6)(3_6)(6_6)`

Algorithm 4 Generate Traversability Graph Helper Functions

function CanReach($C_{k1}, min_x, min_y, C_{k2}, I_{ij}^{l\to r}, I_{ij}^{l\to t}, I_{ij}^{b\to r}, I_{ij}^{b\to t}$)
    ▷ Determines if C_{k1} can reach C_{k2}.
    if C_{k1} is on left border and C_{k2} is on right border **then**
        return $C_{k2}.A.y \in I_{ij}^{l\to r} \wedge C_{k2}.A.y \geq min_y$
    if C_{k1} is on left border and C_{k2} is on top border **then**
        return $C_{k2}.A.x \in I_{ij}^{l\to t} \wedge C_{k2}.A.x \geq min_x$
    if C_{k1} is on bottom border and C_{k2} is on right border **then**
        return $C_{k2}.A.y \in I_{ij}^{b\to r} \wedge C_{k2}.A.y \geq min_y$
    if C_{k1} is on bottom border and C_{k2} is on top border **then**
        return $C_{k2}.A.x \in I_{ij}^{b\to t} \wedge C_{k2}.A.x \geq min_x$

function ConstructCriticalReach(L_{ij}^C, B_{ij}^C, $L_{i+1,j}^R$, $B_{i,j+1}^R$)
    ▷ Constructs $L_{i+1,j}^C$ and $B_{i,j+1}^C$ from L_{ij}^C, B_{ij}^C, $L_{i+1,j}^R$, $B_{i,j+1}^R$
    $L_{i+1,j}^C := \{\}$
    $B_{i,j+1}^C := \{\}$
    if $L_{i+1,j}^R$ is not closed **then**
        for $(C_k, min_x) \in B_{ij}^C$ **do**
            Add $(C_k, L_{i+1,j}^R.min)$ to $L_{i+1,j}^C$
        for $(C_k, min_y) \in L_{ij}^C$ **do**
            if $min_y \leq L_{i+1,j}^R.max$ **then**
                $min_y \leftarrow max(min_y, L_{i+1,j}^R.min)$
                Add (C_k, min_y) to $L_{i+1,j}^C$
    if $B_{i,j+1}^R$ is not closed **then**
        for $(C_k, min_y) \in L_{ij}^C$ **do**
            Add $(C_k, B_{i,j+1}^R.min)$ to $B_{i,j+1}^C$
        for $(C_k, min_x) \in B_{ij}^C$ **do**
            if $min_x \leq B_{i,j+1}^R.max$ **then**
                $min_x \leftarrow max(min_x, B_{i,j+1}^R.min)$
                Add (C_k, min_x) to $B_{i,j+1}^C$
    return $L_{i+1,j}^C, B_{i,j+1}^C$

The two functions defined in algorithm 4 are necessary for algorithm 3 to generate the correct traversability graph for a set of critical events at the same height ϵ_0.

Checking if $C_{k2}.A$ lies in the respective reachability interval seems superfluous, because we are already checking for the minimum x- or y-coordinate.

Nevertheless, also checking the reachability interval I_{ij} does not hurt valid functioning.

5.4 Traversal Cross-Section $f(t)$ and Profile $\hat{f}(s)$

In this section we will define a traversals cross-section $f(t)$ and its profile $\hat{f}(s)$.

5.4.1 Hyperbolas

Due to our input being two paths consisting of connected straight line-segments, every straight cross-section through our height function δ is a continuous composition of hyperbolas. We use the following function to denote these hyperbolas h:

$$h : \epsilon(t) = \sqrt{u^2 + a(t - v)^2}$$

Each hyperbola has three parameters u, a, and v. A hyperbola is always a cross-section though a segment of the height function δ. An equation system is used to determine the three parameters u, a, and v. There are two cases to consider:

1. For a horizontal of vertical cross-section, $a = 1$. Therefore, only two points (t, ϵ) are needed to determine u and v. The two points (t, ϵ) are calculated by sampling two pairs of points from the input line-segments and determining there t and distance ϵ.

2. For a diagonal cross-section three points (t, ϵ) are needed to determine u, a and v. The two points (t, ϵ) are calculated by sampling three pairs of points from the input line-segments and determining there t and distance ϵ.

For simplicity, we define the hyperbolas all as starting at $t = 0$. This can be done by adjusting the parameter v.

5.4.2 Cross-Section $f(t)$

To attain a cross-section $f(t)$ of an entire traversal, we simply create a composite function of the N hyperbolas and move them along the t-axis to the correct position. This yields $f(t)$. t_k is the time interval in which the k^{th} hyperbola is traversed.

$$f(t) = \begin{cases} \sqrt{u_1^2 + a_1(t - v_1)^2} & 0 \leq t \leq t_1 \\ \sqrt{u_2^2 + a_2(t - v_2 - t_1)^2} & t_1 \leq t \leq t_2 \\ \dots & \\ \sqrt{u_N^2 + a_N(t - v_N - t_{N-1})^2} & t_{N-1} \leq t \leq t_N \end{cases}$$

For the example in figure 4, the cross-section $f(t)$ is shown in figure 10. The x-axis shows the time and the y-axis shows the current ϵ. Notice that when the traversal moves on the l-lines the cross-section is linear. This is because the addend u^2 inside the square root equals zero for this case. Therefore, the square and the square root cancel and we are left with the absolute value of a linear function: $|\sqrt{a}(t - v)|$.

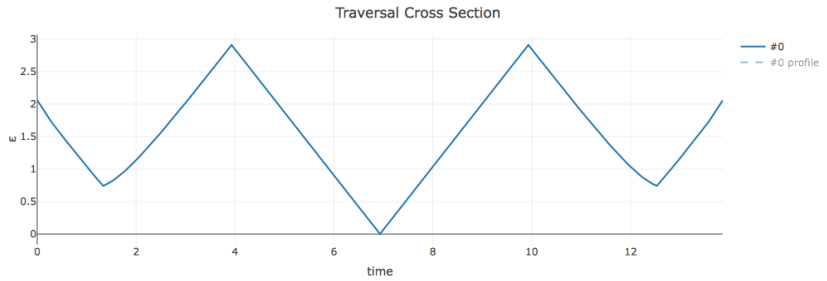


Figure 10: Cross-section $f(t)$ for example from section 5.2.1[7]

5.4.3 Profile $\hat{f}$

In addition to the cross-section $f(t)$, we can also look at the profile $\hat{f}(s)$. Rote defines the profile function $\hat{f}(s)$ as the time that ϵ exceeds a threshold s:

$$\hat{f}(s) = \mu(\{t \mid f(t) \geq s\})$$

where μ denotes the Lebesgue measure.[2]

Figure 11 shows the profile function $\hat{f}(s)$ in addition to the function of the cross-section shown in figure 10 for the example from figure 4.

[7]View the example here: `https://abegehr.github.io/frechet/?p=(4_5)(8_5)(4_5)&q=(5_4)(5_7)(5_4)`

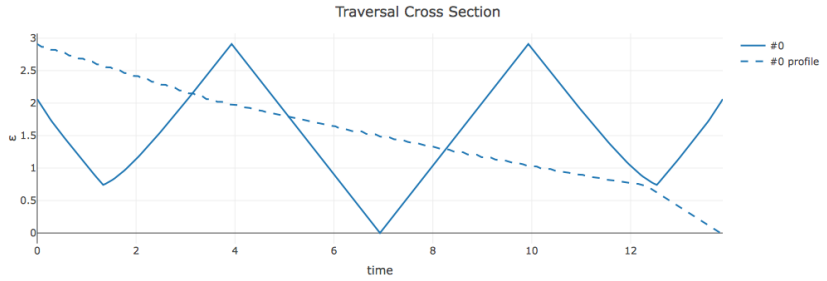


Figure 11: Cross-section $f(t)$ and profile $\hat{f}(s)$ for example from section 5.2.1[7]

The profile function $\hat{f}(s)$ is defined using the cross-section hyperbolas. It is important to note that each hyperbola function $h_j(t)$ is defined so that $0 \leq t \leq t_j - t_{j-1}$; therefore, we can simply sum up the inverses of the hyperbolas in these bounds to calculate the profile in the following two steps:

1. Compute the inverse function of a hyperbolas function $h^{-1}(\epsilon)$. First solve $\epsilon(t)$ for t:

$$\begin{aligned}
\epsilon &= \sqrt{u^2 + a(t-v)^2} \\
\epsilon^2 &= u^2 + a(t-v)^2 \\
\epsilon^2 - u^2 &= a(t-v)^2 \\
\frac{\epsilon^2 - u^2}{a} &= (t-v)^2 \\
\pm\sqrt{\frac{\epsilon^2 - u^2}{a}} &= t - v \\
v \pm \sqrt{\frac{\epsilon^2 - u^2}{a}} &= t
\end{aligned}$$

This results in the following definition for $h^{-1}(\epsilon)$:

$$h_j^{-1} : t_j(\epsilon) = \begin{cases} v_j \pm \sqrt{\frac{\epsilon^2 - u_j^2}{a_j}} & \text{if } h_j(0) \leq \epsilon \leq h_j(t_j - t_{j-1}) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Note that h_j is valid in the interval $0 \leq t \leq t_j - t_{j-1}$; therefore, h_j^{-1} is defined for $h_j(0) \leq \epsilon \leq h_j(t_j - t_{j-1})$. This is important to yield the correct profile function $\hat{f}(s)$. Also notice that each hyperbola h_j has its own three parameters: u_j, a_j, and v_j.

2. Sum up the inverses to yield the profile function $\hat{f}(s)$.

$$\hat{f}(s) = \sum_{j=0}^{N} h_j^{-1}(s)$$

Now we have defined the profile function $\hat{f}(s)$ for a traversal with cross-section $f(t)$ that allows us to calculate the amount of time a traverser traversing this traversal spends above the threshold s.

5.5 Multiple Ascents to and Descents from ϵ_0

To fulfill our goal of a lexicographic traversal, we want to minimize the time during which the height ϵ exceeds a threshold s.[2] Concerning our examples, we want to minimize the time needed to ascend to and descend from our critical height ϵ_0.

If we have multiple critical events at the critical height ϵ_0, we need to decide which to choose while fulfilling our goal of a lexicographic traversal. The profile function $\hat{f}(s)$, introduced in section 5.4, defines the time that a traversals exceeds a threshold s. Our goal of a lexicographic traversal is to minimize this time. This means that for two profile functions $\hat{f}(s)$ and $\hat{g}(s)$ that both are critical at ϵ_0, we need to compare the derivatives.

$$\hat{f}'(s) = \left[\sum_{j=0}^{N} h_j^{-1}(s)\right]'$$

Because of the sum rule, we can apply the derivative to each h_j^{-1}.

$$\hat{f}'(s) = \sum_{j=0}^{N} [h_j^{-1}]'(s)$$

5.5.1 Simplifications and First Derivative

Due to the form of h_j^{-1} (see equation 1), the derivative of h_j^{-1} is not trivial. h_j^{-1} is a composite function and has a $\pm$. We circumvent this problem by only looking at hyperbolas that ascend to and descend from the critical height ϵ_0.

For a critical event C_k at ϵ_0, we will denote the ascent hyperbola with $h_{k\uparrow}$ and the descent hyperbola with $h_{k\downarrow}$.

We will only consider critical events with one ascent and one descent hyperbola, analogous to classical critical events of type b. Classical critical events of type c can easily be modeled by multiple critical events of type

b. Classical critical events of type a and new type of lexicographic critical events consist of only one ascent or descent hyperbola.

Because of simplicity we move the hyperbolas so that $t = 0$ for the critical height ϵ: $h(0) = \epsilon_0$.

The hyperbolas are vertically symmetric at $t_v = v$, $h(t_v + t) = h(t_v - t)$; therefore, the inverse hyperbolas are horizontally symmetric at $t_v = v$. This means that to get rid of the $\pm$ in the inverse and get only positive slopes, we can simply use a $+$ when we calculate the derivative:

$$
\begin{aligned}
[h^{-1}]'(\epsilon) &= \left[v + \sqrt{\frac{\epsilon^2 - u^2}{a}}\right]' \\
[h^{-1}]'(\epsilon) &= \left[\sqrt{\frac{\epsilon^2 - u^2}{a}}\right]' \\
[h^{-1}]'(\epsilon) &= \frac{\left[\frac{\epsilon^2 - u^2}{a}\right]'}{2\sqrt{\frac{\epsilon^2 - u^2}{a}}} \\
[h^{-1}]'(\epsilon) &= \frac{[\epsilon^2 - u^2]'}{2a\sqrt{\frac{\epsilon^2 - u^2}{a}}} \\
[h^{-1}]'(\epsilon) &= \frac{2\epsilon}{2a\sqrt{\frac{\epsilon^2 - u^2}{a}}} \\
[h^{-1}]'(\epsilon) &= \frac{\epsilon}{a\sqrt{\frac{\epsilon^2 - u^2}{a}}} \qquad (2)
\end{aligned}
$$

The hyperbolas are defined so that: $h(0) = \epsilon_0$; therefore, $h^{-1}(\epsilon_0) = 0$.

$$
\begin{aligned}
h^{-1}(\epsilon_0) &= v \pm \sqrt{\frac{\epsilon_0^2 - u^2}{a}} \\
0 &= v \pm \sqrt{\frac{\epsilon_0^2 - u^2}{a}} \\
-v &= \pm\sqrt{\frac{\epsilon_0^2 - u^2}{a}} \\
|-v| &= \left|\pm\sqrt{\frac{\epsilon_0^2 - u^2}{a}}\right| \\
|v| &= \sqrt{\frac{\epsilon_0^2 - u^2}{a}} \qquad (3)
\end{aligned}
$$

We can write a simple equation for $[h^{-1}]'(\epsilon_0)$ by using the equations 2 and 3:

$$
\begin{aligned}
[h^{-1}]'(\epsilon_0) &= \frac{\epsilon_0}{a\sqrt{\frac{\epsilon_0^2 - u^2}{a}}} \\
[h^{-1}]'(\epsilon_0) &= \frac{\epsilon_0}{a\,|v|} \qquad (4)
\end{aligned}
$$

Having an equation for $[h^{-1}]'(\epsilon_0)$, we will now determine the slope at $\hat{f}_{\epsilon_0}(\epsilon_0)$ by adding the ascending and descending slopes of all critical events C at critical height ϵ_0:

$$
\begin{aligned}
\hat{f}'_{\epsilon_0}(\epsilon_0) &= \sum_{C_k \in C(\epsilon_0)} [h^{-1}_{k\downarrow}]'(\epsilon_0) + [h^{-1}_{k\uparrow}]'(\epsilon_0) \\
\hat{f}'_{\epsilon_0}(\epsilon_0) &= \sum_{C_k \in C(\epsilon_0)} \frac{\epsilon_0}{a_{k\downarrow}\,|v_{k\downarrow}|} + \frac{\epsilon_0}{a_{k\uparrow}\,|v_{k\uparrow}|}
\end{aligned}
$$

There are three cases to consider:

1. $\hat{f}'_{\epsilon_0}(\epsilon_0) > \hat{g}'_{\epsilon_0}(\epsilon_0)$: Choose traversal for f.
2. $\hat{f}'_{\epsilon_0}(\epsilon_0) < \hat{g}'_{\epsilon_0}(\epsilon_0)$: Choose traversal for g.
3. $\hat{f}'_{\epsilon_0}(\epsilon_0) = \hat{g}'_{\epsilon_0}(\epsilon_0)$: Cannot decide.

Cases one and two are clear. Case three on the other hand, when the slope of both profiles at ϵ_0 is the same, no decision can be made at that point. We would have to look at further derivatives.

5.5.2 Second Derivative

The the calculation of h^{-1}'s the second derivative:

$$
\begin{aligned}
[h^{-1}]''(\epsilon) &= \left[\frac{\epsilon}{a\sqrt{\frac{\epsilon^2-u^2}{a}}}\right]' \\
[h^{-1}]''(\epsilon) &= \frac{\left[\frac{\epsilon}{\sqrt{\frac{\epsilon^2-u^2}{a}}}\right]'}{a} \\
[h^{-1}]''(\epsilon) &= \frac{\frac{[\epsilon]'}{\sqrt{\frac{\epsilon^2-u^2}{a}}} + \epsilon\left[\frac{1}{\sqrt{\frac{\epsilon^2-u^2}{a}}}\right]'}{a} \\
[h^{-1}]''(\epsilon) &= \frac{\frac{1}{\sqrt{\frac{\epsilon^2-u^2}{a}}} - \frac{\epsilon\left[\frac{\epsilon^2-u^2}{a}\right]'}{2\left(\frac{\epsilon^2-u^2}{a}\right)^{\frac{3}{2}}}}{a} \\
[h^{-1}]''(\epsilon) &= \frac{\frac{1}{\sqrt{\frac{\epsilon^2-u^2}{a}}} - \frac{2\epsilon^2}{2a\left(\frac{\epsilon^2-u^2}{a}\right)^{\frac{3}{2}}}}{a} \\
[h^{-1}]''(\epsilon) &= \frac{1}{a\sqrt{\frac{\epsilon^2-u^2}{a}}} - \frac{\epsilon^2}{a^2\left(\frac{\epsilon^2-u^2}{a}\right)^{\frac{3}{2}}} \\
[h^{-1}]''(\epsilon) &= \left(\frac{1}{a\left(\frac{\epsilon^2-u^2}{a}\right)^{\frac{1}{2}}} - \frac{\epsilon^2}{a^2\left(\frac{\epsilon^2-u^2}{a}\right)^{\frac{3}{2}}}\right)\frac{\left(\frac{\epsilon^2-u^2}{a}\right)^{\frac{3}{2}}}{\left(\frac{\epsilon^2-u^2}{a}\right)^{\frac{3}{2}}} \\
[h^{-1}]''(\epsilon) &= \left(\frac{\epsilon^2-u^2}{a^2} - \frac{\epsilon^2}{a^2}\right)\frac{1}{\left(\frac{\epsilon^2-u^2}{a}\right)^{\frac{3}{2}}} \\
[h^{-1}]''(\epsilon) &= \left(\frac{-u^2}{a^2}\right)\frac{1}{\left(\frac{\epsilon^2-u^2}{a}\right)^{\frac{3}{2}}} \\
[h^{-1}]''(\epsilon) &= -\frac{u^2}{a^2\left(\frac{\epsilon^2-u^2}{a}\right)^{\frac{3}{2}}} \qquad (5)
\end{aligned}
$$

We can write a simple equation for $[h^{-1}]'(\epsilon_0)$ by using the equations 5 and 3:

$$
[h^{-1}]''(\epsilon_0) = -\frac{u^2}{a^2\left|v\right|^3} \qquad (6)
$$

Again, we sum up all $[h^{-1}]''(\epsilon_0)$ and compare the sums of each traversal analogous to the description in section 5.5.1. We choose the traversal with the largest $\sum_{C_k \in C(\epsilon_0)} [h^{-1}_{k\uparrow}]''(\epsilon_0) + [h^{-1}_{k\downarrow}]''(\epsilon_0)$.

5.5.3 Example: First Derivative

Let us consider an example where we can decide from looking at the first derivative which traversal path to choose. Figure 12 shows such an example.

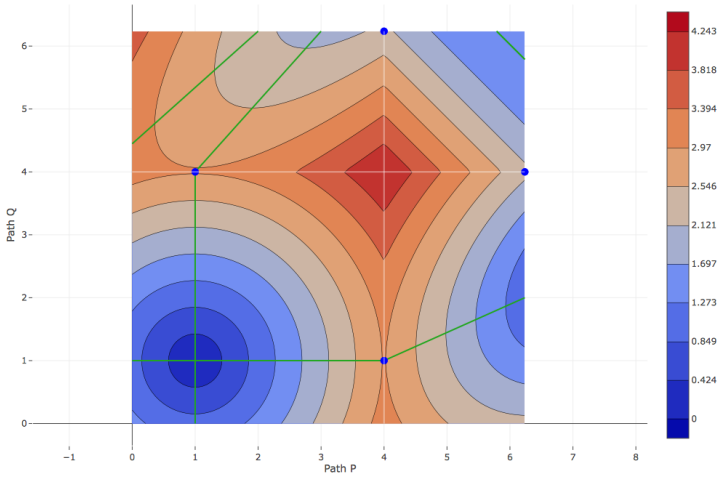


Figure 12: First derivative decides[8]

In the example depicted in figure 12, there are two classical critical events of type b at critical height $\epsilon_0 = 3$ to consider: $C_1(1, 4)$ and $C_2(4, 1)$.

The algorithm 3 that generates a traversability graph for multiple critical events at the same height as defined in section 5.3 returns that there are two possible paths through the critical events C_1 and C_2. Either the graph is traversed from the starting-point to the ending-point over C_1 (traversal path f) or over C_2 (traversal path g).

We determine the hyperbolas representing the ascent to and descent from of C_1 and C_2. Decimal numbers are rounded to three places after the decimal point.

[8]View the example here: `https://abegehr.github.io/frechet/?p=(0_1)(4_1)(2_2)&q=(1_0)(1_4)(3_3)`

$$
\begin{aligned}
h_{f1\uparrow}(t) &= \sqrt{0^2 + 1 \cdot (t - (-3))^2} = |t + 3| \\
h_{f1\downarrow}(t) &\approx \sqrt{0^2 + 0.8 \cdot (t - 3.354)^2} = \left|\sqrt{0.8}(t - 3.354)\right| \\
h_{g1\uparrow}(t) &= \sqrt{0^2 + 1 \cdot (t - (-3))^2} = |t + 3| \\
h_{g1\downarrow}(t) &\approx \sqrt{0^2 + 0.2 \cdot (t - 6.708)^2} = \left|\sqrt{0.2}(t - 6.708)\right|
\end{aligned}
$$

Next, we calculate the specific slopes of the inverse hyperbolas at $\epsilon_0 = 3$ by using equation 4.

$$
\begin{aligned}
h'_{f1\uparrow}(\epsilon_0) &= \frac{3}{1 \cdot |-3|} = 1 \\
h'_{f1\downarrow}(\epsilon_0) &\approx \frac{3}{0.8 \cdot |3.354|} \approx 1.118 \\
h'_{g1\uparrow}(\epsilon_0) &= \frac{3}{1 \cdot |-3|} = 1 \\
h'_{g1\downarrow}(\epsilon_0) &\approx \frac{3}{0.2 \cdot |6.708|} \approx 2.236
\end{aligned}
$$

From the slopes, $\hat{f}'_{\epsilon_0}(\epsilon_0)$ and $\hat{g}'_{\epsilon_0}(\epsilon_0)$ are now calculated.

$$
\begin{aligned}
\hat{f}'_{\epsilon_0}(\epsilon_0) &= h'_{f1\uparrow}(\epsilon_0) + h'_{f1\downarrow}(\epsilon_0) \approx 2.118 \\
\hat{g}'_{\epsilon_0}(\epsilon_0) &= h'_{g1\uparrow}(\epsilon_0) + h'_{g1\downarrow}(\epsilon_0) \approx 3.236
\end{aligned}
$$

The result is clear.

$$
\hat{f}'_{\epsilon_0}(\epsilon_0) \approx 2.118 < \hat{g}'_{\epsilon_0}(\epsilon_0) \approx 3.236
$$

The descent from height ϵ_0 is steeper when taking path g over $C_2(4, 1)$. This can also be seen in figure 12. The contour lines (lines of equal ϵ) are closer together on the right of $C_2(4, 1)$ than they are above $C_1(1, 4)$.

5.5.4 Example: Second Derivative

Figure 13 shows an example where it will not be possible to decide from looking at the first derivative which path to traverse.

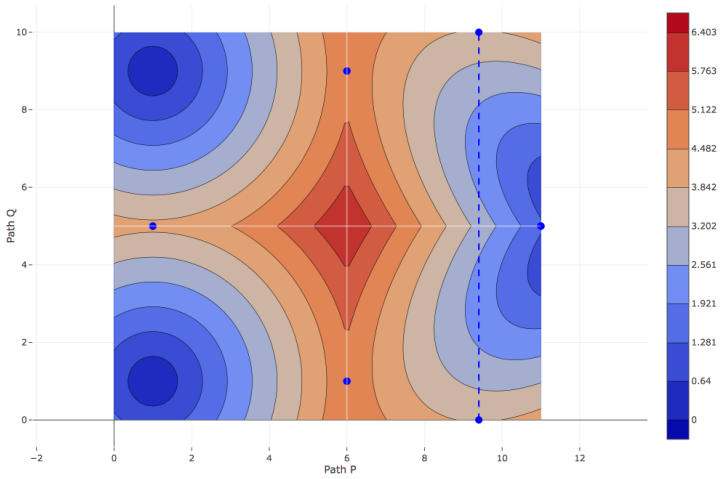


Figure 13: Second derivative decides[9]

In the example depicted in figure 13, there are two classical critical events of type b at critical height $\epsilon_0 = 5$ to consider: $C_1(6, 1)$ and $C_2(6, 9)$.

The algorithm 3 that generates a traversability graph for multiple critical events at the same height as defined in section 5.3 returns that there are two possible paths through the critical events C_1 and C_2. Either the graph is traversed from the starting-point to the ending-point over C_1 (traversal path f) or over C_2 (traversal path g).

We determine the hyperbolas representing the ascent to and descent from of C_1 and C_2.

$$
\begin{aligned}
h_{f1\uparrow}(t) &= \sqrt{0^2 + 1 \cdot (t - (-5))^2} = |t + 5| \\
h_{f1\downarrow}(t) &= \sqrt{0^2 + 0.64 \cdot (t - 6.25)^2} = |0.8 \cdot (t - 6.25)| \\
h_{g1\uparrow}(t) &= \sqrt{0^2 + 1 \cdot (t - (-5))^2} = |t + 5| \\
h_{g1\downarrow}(t) &= \sqrt{3^2 + 1 \cdot (t - 4)^2} = \sqrt{9 + (t - 4)^2}
\end{aligned}
$$

Next, we calculate the specific slopes of the inverse hyperbolas at $\epsilon_0 = 5$ by using equation 4.

[9]View the example here: `https://abegehr.github.io/frechet/?p=(3_2)(3_8)(6_4)&q=(2_3)(7_3)(2_3)`

$$h'_{f1\uparrow}(\epsilon_0) = \frac{5}{1 \cdot |-5|} = 1$$
$$h'_{f1\downarrow}(\epsilon_0) = \frac{5}{0.64 \cdot |6.25|} = 1.25$$
$$h'_{g1\uparrow}(\epsilon_0) = \frac{5}{1 \cdot |-5|} = 1$$
$$h'_{g1\downarrow}(\epsilon_0) = \frac{5}{1 \cdot |4|} = 1.25$$

From the slopes, $\hat{f}'_{\epsilon_0}(\epsilon_0)$ and $\hat{g}'_{\epsilon_0}(\epsilon_0)$ are now calculated.

$$\hat{f}'_{\epsilon_0}(\epsilon_0) = h'_{f1\uparrow}(\epsilon_0) + h'_{f1\downarrow}(\epsilon_0) = 2.25$$
$$\hat{g}'_{\epsilon_0}(\epsilon_0) = h'_{g1\uparrow}(\epsilon_0) + h'_{g1\downarrow}(\epsilon_0) = 2.25$$

The result is unclear.

$$\hat{f}'_{\epsilon_0}(\epsilon_0) = 2.25 = \hat{g}'_{\epsilon_0}(\epsilon_0) = 2.25$$

The first derivative did not yield a definitive answer to the question on which path to traverse. We need to look at the second derivative now. We calculate the specific second derivatives of the inverse hyperbolas at $\epsilon_0 = 5$ by using equation 6.

$$h''_{f1\uparrow}(\epsilon_0) = -\frac{0^2}{1^2 \, |-5|^3} = 0$$
$$h''_{f1\downarrow}(\epsilon_0) = -\frac{0^2}{0.64^2 \, |6.25|^3} = 0$$
$$h''_{g1\uparrow}(\epsilon_0) = -\frac{0^2}{1^2 \, |-5|^3} = 0$$
$$h''_{g1\downarrow}(\epsilon_0) = -\frac{3^2}{1^2 \, |-4|^3} \approx -0.141$$

From the specific second derivatives, $\hat{f}''_{\epsilon_0}(\epsilon_0)$ and $\hat{g}''_{\epsilon_0}(\epsilon_0)$ are now calculated.

$$\hat{f}''_{\epsilon_0}(\epsilon_0) = h''_{f1\uparrow}(\epsilon_0) + h''_{f1\downarrow}(\epsilon_0) = 0$$
$$\hat{g}''_{\epsilon_0}(\epsilon_0) = h''_{g1\uparrow}(\epsilon_0) + h''_{g1\downarrow}(\epsilon_0) = -0.141$$

The result is clear.

$$\hat{f}' l_{\epsilon_0}(\epsilon_0) = 0 > \hat{g}''_{\epsilon_0}(\epsilon_0) \approx -0.141$$

While the descent from height ϵ_0 is equally steep for both paths f and g, the steepness of path g's descent flattens as can be seen by viewing the second derivative. Figure 14 shows the two possible traversal paths. The traversal through the bottom is the path g. The traversal through the top is path f.

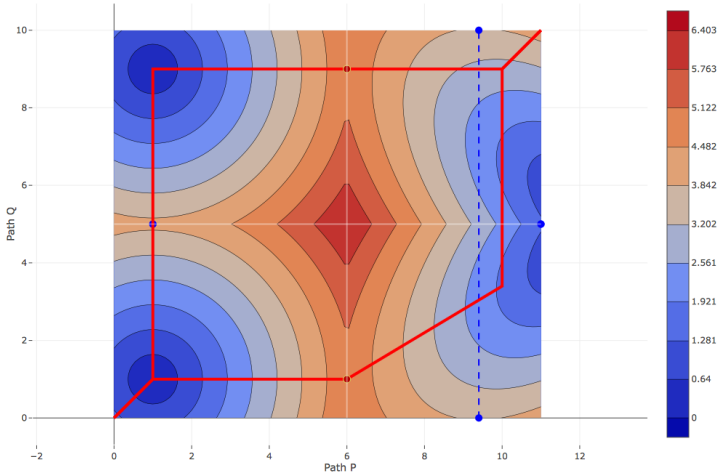


Figure 14: Second derivative decides, traversed[9]

Figure 15 shows the cross-sections and profiles for the traversals f and g. #0 represents g and #1 represents f.

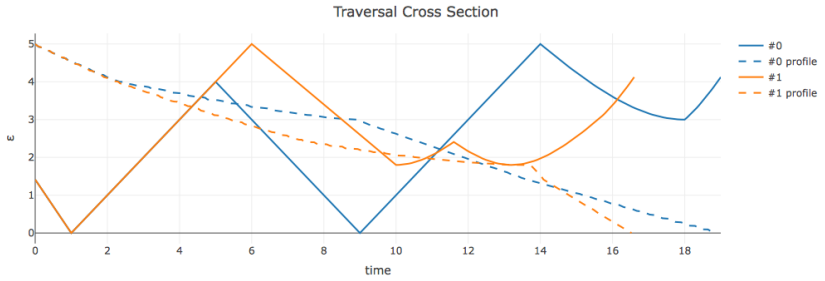


Figure 15: Second derivative decides, traversed[9]

Looking at figure 15, the two critical events with ϵ_0 can be seen. The left one is C_1, the right one is C_2. The slopes at both critical events are equal, but looking at the descent from C_2 one can see that it bends upward and flattens.

6 Concluding Perspective

This Bachelor thesis with the title "Lexicographic Fréchet Matchings with Degenerate Inputs" discussed three main topics from the Fréchet distance spectrum that build on each other.

First, the classical Fréchet distance as researched by Alt and Godau in their 1995 paper "Computing the Fréchet distance between two polygonal curves"[1] is explored. We discussed the free-space diagram, classical critical events of types a, b, and c, and the decision problem.

Second, Rote's 2014 paper "Lexicographic Fréchet Matchings"[2] guides us to extend the Fréchet distance to accomplish a lexicographic matching based on the classical Fréchet distance. Rote introduced the steepest descent, a new lexicographic type of critical events, and of course the algorithm for traversing a height function δ lexicograohically.

Third, we investigate lexicographic Fréchet matchings with degenerate inputs based on section 7 of Rote's paper[2]. The problem of multiple critical events at the same critical height ϵ is defined and visualized by numerous examples. Through the examples we discover that two procedures need to be defined. First we need to know which paths through the critical events are possible and second we need to know which of the paths results in the lexicographically optimal profile to fulfill the goal of a lexicographic traversal.

The problem of degenerate inputs in terms of lexicographic Fréchet matchings is by no mean solved at this point. The next sections showcase mayor questions that still exist.

6.1 Third, Fourth, Fifth, etc. Derivative

Equation 2 defines $[h^{-1}]'(\epsilon)$:

$$[h^{-1}]'(\epsilon) = \frac{\epsilon}{a\sqrt{\frac{\epsilon^2-u^2}{a}}}$$

Due to the square root in the denominator of $[h^{-1}]'(\epsilon)$, $[h^{-1}]'(\epsilon)$ potentially has infinite derivatives.

After taking three derivatives, the values of these three derivatives can be used in an equation system to calculate the three parameters u, a, and v. If we compare two hyperbola inverses, it is enough to compare three derivatives, because if the first three derivatives are the same for two hyperbola inverses, the two hyperbola inverses have the same parameters u, a, and v.

We can extrapolate this to multiple hyperbola inverses: If we are comparing N hyperbola inverses to N other hyperbola inverses, we need to compare at most $3 \cdot N$ derivatives, because with $3 \cdot N$ derivatives an equation system can be established from which the parameters u, a, and v can be calculated for each one of the N hyperbola inverses. Therefore the N hyperbolas would be exactly the same. This procedure is only possible for fs and gs that have the same number N of critical events.

It would be interesting to try to find or construct examples where higher derivatives than the second one are needed. Researching could possibly lead to a proof that only a finite number of derivatives is needed. It might also make sense to look at more derivatives of $h^{-1}(\epsilon)$. See the first five derivatives here:

$$\begin{aligned}
[h^{-1}]'(\epsilon) &= \frac{\epsilon}{a\sqrt{\frac{\epsilon^2-u^2}{a}}} \\
[h^{-1}]''(\epsilon) &= -\frac{u^2}{a^2\left(\frac{\epsilon^2-u^2}{a}\right)^{\frac{3}{2}}} \\
[h^{-1}]'''(\epsilon) &= \frac{3u^2\epsilon}{a^3\left(\frac{\epsilon^2-u^2}{a}\right)^{\frac{5}{2}}} \\
[h^{-1}]''''(\epsilon) &= \frac{3u^2\left(-u^2-4\epsilon^2\right)}{a^4\left(\frac{\epsilon^2-u^2}{a}\right)^{\frac{7}{2}}} \\
[h^{-1}]'''''(\epsilon) &= \frac{15u^2\epsilon\left(3u^2+4\epsilon^2\right)}{a^5\left(\frac{\epsilon^2-u^2}{a}\right)^{\frac{9}{2}}}
\end{aligned}$$

6.2 Equal Descents at ϵ_0, Unequal Descents Below

In section 5.2.2 we saw an example where two paths were exactly equal. The example shown in figure 16 is constructed by slightly mutating the example from section 5.2.2.

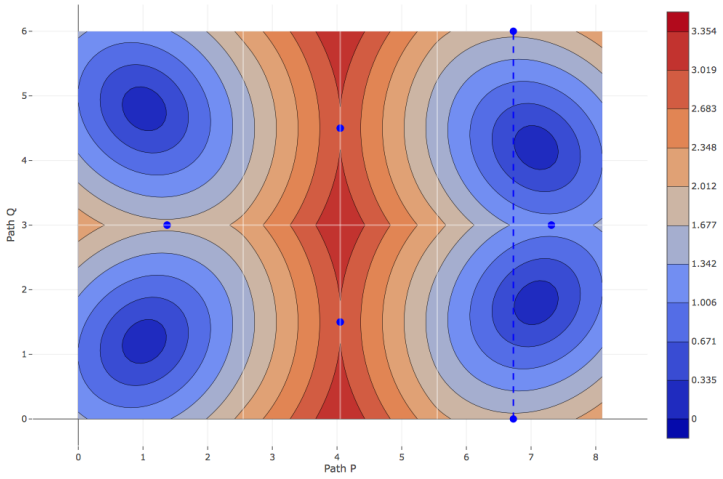


Figure 16: Equal descents at ϵ_0, unequal descents below[10]

The critical height $\epsilon_0 = 3$. There are two critical events: $C_1(4.03, 1.5)$ and $C_1(4.03, 4.5)$. Coordinates are rounded to two decimal places.

The are two paths through the height map δ. One over the classical critical event of type b $C_1(4.03, 1.5)$ and one over the classical critical event of type b $C_1(4.03, 4.5)$.

The hyperbolas representing the ascents to and the descents from C_1 and C_2 are the same. The contain equal parameters. Therefore both paths are traversed (see figure 17).

[10]View the example here: `https://abegehr.github.io/frechet/?p=(4_5)(6.5_5.5)(8_5.5)(6.5_5.5)(4_6)&q=(5_4)(5_7)(5_4)`

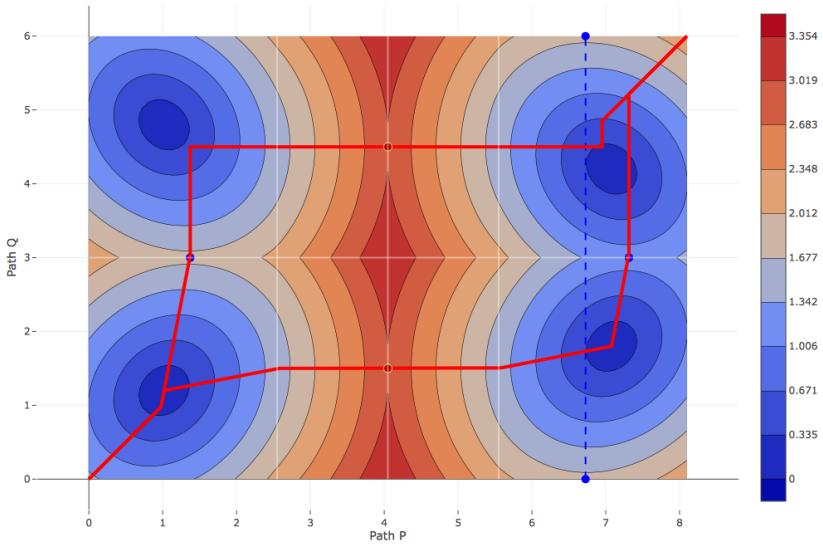


Figure 17: Equal descents at ϵ_0, unequal descents below, traversed[10]

At height $\epsilon_0 = 3$ the profiles of both paths are equal, but at $\epsilon_1 \approx 1.765$ the profile of the lower path over C_1 turns out to have been the better choice. Figure 18 shows the cross-sections and profiles of both traversals.

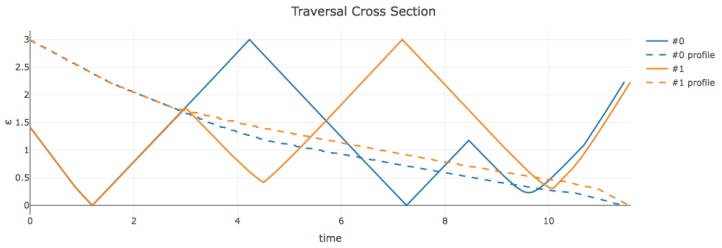


Figure 18: Cross-sections and profiles[10]

The traversal #0 depicts the cross-section and profile for the lower traversal over C_1. The traversal #1 depicts the cross-section and profile for the upper traversal over C_2. It can be seen clearly by looking at the cross-sections and profiles of the two traversals that traversing the path over $C_1(4.03, 1.5)$ yields the lexicographically optimal traversal.

The problem is that when the algorithm decides if C_1 or C_2 should be traversed, the algorithm does not yet know what happens below $\epsilon_0 = 3$. The algorithm would need to traverse both paths and decide at $\epsilon_1 = 1.765$ which one of both to discard and which one to keep traversing.

A Appendices

A.1 Fréchet Web App

The Fréchet web app and its development is not the focus of this paper; nevertheless, some background information about the visualization software and the implementation of the algorithm may be useful.

A.1.1 Development

Initially the algorithm without support for degenerate inputs was implemented in python as part of a software project.

For this Bachelors thesis, the algorithm was then extended towards supporting degenerate inputs by solving multiple critical events at ϵ_0 using the traversability graph and the first derivative of the hyperbolas inverses.

A web frontend was added to make visualizing intuitive and easy.

A.1.2 Live Version and Links

The web app is accessible at: `https://abegehr.github.io/frechet/`. While reading this paper it is highly advised to use the links to the Fréchet web app to view examples that are appended throughout the text. The web app has features that make understanding examples easier that viewing a static picture.

A.1.3 Source Control

The corresponding GitHub repository is reachable at `https://github.com/abegehr/frechet`. The current version of the source code, the commit history, as well as issues can be viewed there. It is also possible to fork the repository and edit the source code to add features.

A.1.4 Architecture

The Frèchet web app consists of a python backend and a React frontend.

The python backend executes the algorithms on a free Heroku dyno. One endpoint that receives two input paths and returns the visualization data is opened by the backend API. The API is reachable at the url: `https://frechet-server.herokuapp.com`.

The React frontend allows the user to input two paths consisting of straight line-segments, communicates with the python backend, and visu-

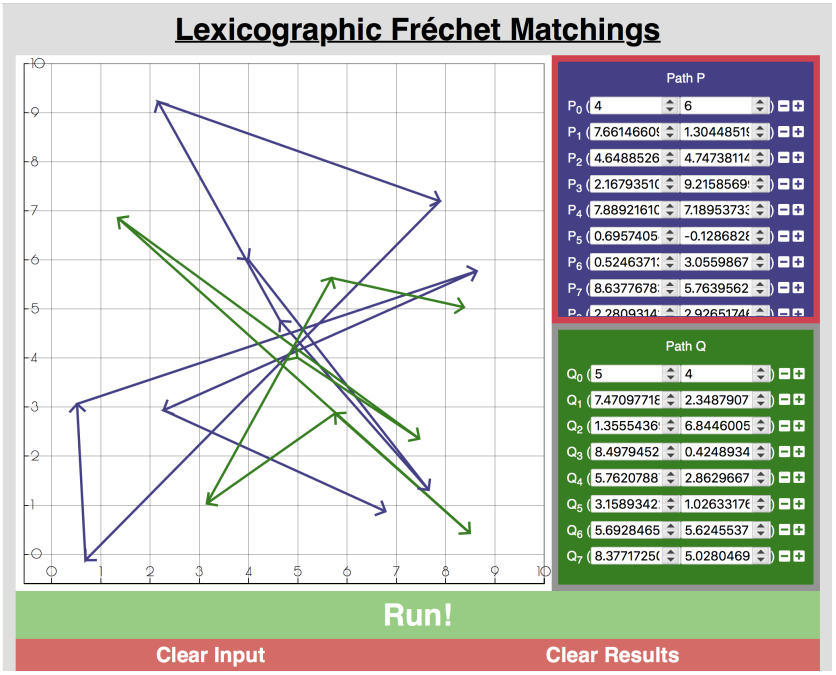
alizes the results computed by the python backend. The frontend is hosted on GitHub static pages.

Initially I wanted to host the Fréchet web app on my Userpage provided by the zedat service of the Freie Universität Berlin, but versioning problems and insufficient support made this unfeasible.

A.1.5 Features

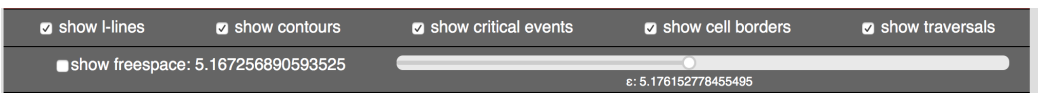
To execute a run of the lexicographic Fréchet matchings algorithm, first visit the web app at `https://abegehr.github.io/frechet/`. Then continue as follows:

1. Input two paths through clicking or by entering coordinates.

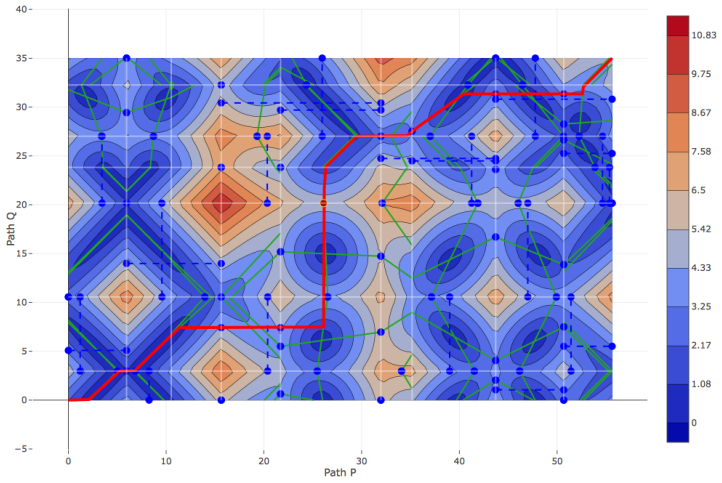


2. Press "Run". The frontend contacts the python backend where the algorithms are executed. When the computations finish, the results are returned to the frontend and displayed in three charts.

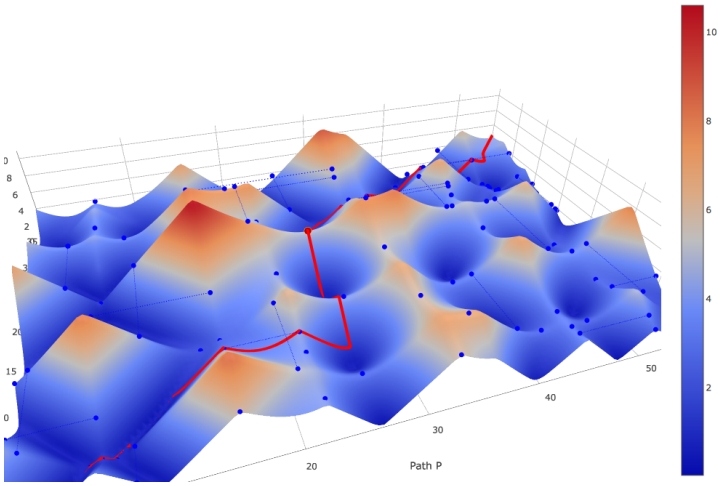
3. If necessary select or deselect options like showing the free-space diagram,



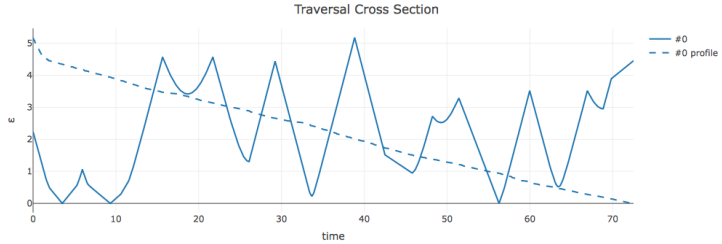
4. View the height diagram δ as a contourmap with critical events and traversals overlaid.



5. View the height diagram δ as a 3D model with critical events and traversals overlaid.



6. View the cross-sections and profiles of traversals.



References

[1] Helmut Alt and Michael Godau. "Computing the Fréchet distance between two polygonal curves". In: *International Journal of Computational Geometry & Applications* 5.01n02 (1995), pp. 75–91.

[2] Günter Rote. "Lexicographic Fréchet Matchings". In: *30th European Workshop on Computational Geometry*. EuroCG. 2014.