Route Reconstruction from Traffic Flow via Representative Trajectories

Bram Custers
Wouter Meulemans
b.a.custers@tue.nl
w.meulemans@tue.nl
TU Eindhoven

Bettina Speckmann
Kevin Verbeek
b.speckmann@tue.nl
k.a.b.verbeek@tue.nl
TU Eindhoven

ABSTRACT

Understanding human mobility patterns is an important aspect of traffic analysis and urban planning. Trajectory data provide detailed views on specific routes, but typically do not capture all traffic. On the other hand, loop detectors built into the road network capture all traffic flow at specific locations, but provide no information on the individual routes. Given a set of loop-detector measurements as well as a (small) set of representative trajectories, our goal is to investigate how one can effectively combine these two partial data sources to create a more complete picture of the underlying mobility patterns. Specifically, we want to reconstruct a realistic set of routes from the loop-detector data, using the given trajectories as representatives of typical behavior.

We model the loop-detector data as a network flow field that needs to be covered by the reconstructed routes and we capture the realism of the routes via the strong Fréchet distance to the representative trajectories. We prove that several forms of the resulting algorithmic problem are NP-hard. Hence we explore heuristic approaches which decompose the flow well while following the representative trajectories to varying degrees. We propose an iterative *Fréchet Routes* (FR) heuristic which generates candidates routes with bounded Fréchet distance to the representative trajectories. We also describe a variant of multi-commodity min-cost flow (MCMCF) which is only loosely coupled to the trajectories.

We perform an extensive experimental evaluation of our two proposed approaches in comparison to a global min-cost flow (GMCF), which is essentially agnostic to the representative trajectories. To make meaningful claims in terms of quality, we derive a ground truth by map-matching real-world trajectories. We find that GMCF explains the flow best, but produces a large number of often non-sensical routes (significantly more than the ground truth). MCMCF produces a large number of mostly realistic routes which explain the flow reasonably well. In contrast, FR produces much smaller sets of realistic routes which still explain the flow well, at the cost of a higher running time. Finally, we report on the results of a case study which combines real-world loop detector data and representative trajectories for the region around The Hague, the Netherlands.

CCS CONCEPTS

• **Theory of computation** → **Computational geometry**.

KEYWORDS

Floating Car Data, Loop Detector Data, Fréchet distance, data fusion, flow decomposition

ACM Reference Format:

Bram Custers, Wouter Meulemans, Bettina Speckmann, and Kevin Verbeek. 2021. Route Reconstruction from Traffic Flow via Representative Trajectories. In *29th International Conference on Advances in Geographic Information Systems (SIGSPATIAL '21)*, November 2–5, 2021, Beijing, China. ACM, New York, NY, USA, 12 pages. <https://doi.org/10.1145/3474717.3483650>

1 INTRODUCTION

Understanding human mobility patterns is an important aspect of traffic analysis and urban planning. To analyze mobility, we need to answer questions about various aspects of (vehicle) traffic, such as “How busy is this road at different times of the day?”, “Between which locations do people travel most frequently?” or “Which routes do people use to travel through the road network?” To answer such questions we can use various heterogeneous data sources, which generally fall into one of two categories: checkpoint data and tracking data.

Checkpoint data originate from measurements by static devices such as loop detectors or traffic cameras, placed on fixed locations throughout the road network. They provide a comprehensive view of the amount of traffic flow at that particular location, but inherently no information on how people navigate through the network. Tracking data, on the other hand – predominantly captured through GPS in smart phones and navigation systems – provide a detailed view of individual behavior in the form of trajectories. However, trajectory data does not describe the general traffic flow, as not all vehicles are tracked or tracked by the same system. Furthermore, the (often significant) detail in trajectories also raises privacy concerns, so trajectory data are frequently segmented and anonymized before analysis.

Both categories thus give only a partial view on the full dynamics of human mobility. Given their complementarity it is natural to investigate possibilities for data fusion, that is, data enrichment that leverages the strength of both. Given loop-detector measurements as well as a (small) set of representative trajectories, we investigate how one can effectively combine these two partial data sources to create a more complete picture of the underlying mobility patterns. Specifically, we want to reconstruct a realistic set of routes from the loop-detector data, using the given trajectories as representatives of typical behavior.

This work is licensed under a Creative Commons Attribution International 4.0 License.
SIGSPATIAL '21, November 2–5, 2021, Beijing, China
© 2021 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-8664-7/21/11.
<https://doi.org/10.1145/3474717.3483650>

Contributions and organization. After a brief review of related work, we formally model our problem in Section 2, while also introducing the necessary concepts and notation. We arrive at a formal problem statement which models the loop-detector data as a time-independent network flow that needs to be covered by the reconstructed routes; we capture the realism of the routes via the strong Fréchet distance to the representative trajectories. In Section 3 we prove that several forms of the resulting algorithmic problem are NP-hard even in restricted settings. Hence we explore heuristic approaches which decompose the flow well while following the representative trajectories to varying degrees. In Section 4, we propose an iterative *Fréchet Routes* (FR) heuristic which generates candidate routes which have bounded Fréchet distance to the representative trajectories. In the same section we also describe a variant of multi-commodity min-cost flow (MCMCF) which is only loosely coupled to the trajectories.

In Section 5 we report on an experimental evaluation of these proposed approaches in comparison to a global min-cost flow baseline (GMCF) which is essentially agnostic to the representative trajectories. To make meaningful claims in terms of quality, we derive a ground truth by map matching real-world trajectories. We find that GMCF explains the flow best, but produces a large number of often nonsensical routes (significantly more than the ground truth). MCMCF produces a large number of mostly realistic routes which explain the flow reasonably well. In contrast, FR produces significantly smaller sets of realistic routes which still explain the flow well, albeit at the cost of a higher running time. In Section 6 we report on the results of a case study which combines real-world loop-detector data and representative trajectories.

Our approaches work for any type of data source that can induce a flow field and are not restricted to loop-detector data. We reflect on our results and discuss avenues for further research in Section 7.

Related work. Our problem is closely related to flow decomposition, where the goal is to decompose an aggregated flow into paths, optimizing a given objective function. Any flow can be decomposed into at most $O(E)$ paths and cycles, where E is the number of edges in the graph [1]. However, given a set of paths that decompose the flow, it is NP-hard to determine the correct integral coefficients for each path [16]. Minimizing the number of paths in a decomposition is also NP-hard [22], and thus various approximation algorithms have been developed [11]. We require that the reconstructed routes are similar to one of the representative trajectories, but we do not require that the flow is explained completely.

We are aware of only a single geometric approach to reconstruct flow from checkpoint data. Duckham *et al.* [10] use the Earth Mover’s distance to estimate the movement of couriers from checkpoint data. Given the limited data, the results are quite accurate, but a full reconstruction of the movement is clearly out of reach.

Reconstructing a route (in a network) given a GPS trajectory is referred to as map matching [2, 12, 14, 24], see also the survey by Quddus *et al.* [21]. The goal is to find a route in a given network, accounting for potential misalignment between GPS measurements and the network, noise inherent in GPS systems, and inaccuracies in the road network. There are a wide variety of available algorithms; various solutions are based on hidden Markov models, a strategy which was pioneered by Newson and Krumm [19]. The

map-matching algorithm by Alt *et al.* [2] is particularly relevant to our work (see Section 4), as it decides in quadratic time whether a graph admits a path with Fréchet distance at most ϵ to an input trajectory. Generally, map-matching techniques are not designed to explain flow data, but rather to correct measurement errors in a single trajectory. Moreover, if we insist on simple routes, map matching is NP-hard for various measures, including the Fréchet distance [17] and Hausdorff distance [5]. If the road network is a perfect grid, routes with bounded (but not necessarily minimal) Fréchet and Hausdorff distance can be found efficiently [5].

The problem we study resembles traffic generation [13]. The main difference is that we prefer a succinct set of routes, which is not the case for traffic generation approaches in general.

2 MODELING

Our input has three components: (1) a road network, given as a graph $\mathcal{G}$; (2) a set of representative trajectories $\mathcal{T}$, each encoded by a sequence of measurements; and (3) loop-detector data, which are traffic-volume measurements expressed as the number of cars at a specific time at a specific location. Our goal is to combine these heterogeneous data sources to create a more complete picture of the underlying mobility patterns. Specifically, we want to reconstruct a realistic set of routes from the loop-detector data, using the trajectories as representatives of typical behavior. Here we discuss the modeling decisions we made to finally arrive at a formal problem statement which is amenable to algorithmic treatment. While doing so we also introduce our notation.

Road network. A road network is specified as a graph, $\mathcal{G} = (V, E)$ with n vertices and m edges. We assume that each vertex $v \in V$ has a position in $\mathbb{R}^2$ and that edges are straight segments between their endpoints. The graph is directed and thus bi-directional roads are represented by two separate edges which lie on top of each other.

Trajectories. A trajectory is a curve in space-time that represents a route taken by a vehicle. A trajectory T is represented as a sequence $\langle p_1, \dots, p_n \rangle$ of measurements p_i , where p_i is a tuple (x_i, y_i, t_i) containing a coordinates $(x_i, y_i) \in \mathbb{R}^2$ and a timestamp t_i . Trajectories obtained from vehicles are often *map-matched* to a road network $\mathcal{G}$: a reasonable driven route in $\mathcal{G}$ is reconstructed for the trajectory (the definition of “reasonable” depends on the modeling decision made by the respective algorithm). The map-matched trajectory can then be expressed by a *route* $P = \langle e_1, \dots, e_k \rangle$: a sequence of edges that encode the traversed path in the road network $\mathcal{G}$. We say that a route is *simple* if it visits every vertex in $\mathcal{G}$ at most once. For ease of notation, we introduce the function $M(P, e)$ that indicates how often the edge $e \in E$ is traversed in the route P . Note that $M(P, e)$ is 0 or 1 if P is simple (necessary but not sufficient), but may take on higher values if P is not simple.

2.1 Modeling loop-detector data as time-independent complete flows

Loop-detector data is gathered by counting the number of vehicles passing an induction loop in the road network, aggregated over a fixed time interval. These time intervals generally range from minutes to hours and thus can give an accurate view on the traffic volume over time at that particular location.

Mathematically speaking, loop-detector data can be interpreted as an (incomplete) **flow** on the road network: we assign the aggregate loop-detector data to the edge in the road network where the detector is located and interpret the data as the volume of traffic through this edge. As not every edge in a road network has a loop detector, the flow data is a priori incomplete. Reconstructing a complete flow is challenging due to the inherent uncertainty surrounding the exact driven routes between detector locations [9]. We focus initially on **complete flow** information, and briefly consider incomplete flows in the case study.

Loop-detector data inherently depend on time and hence a priori imply a time-dependent flow. However, time-dependent flows pose several data, modeling, and complexity problems. First of all, we need additional data to model the time needed to traverse the network. We need to know the travel times for edges to be able to reason about realistic routes in the network for the flow. The realism of the reconstructed routes then heavily depends on the accuracy of these values and on time itself. Furthermore, time-dependent flows naturally require the use of time-dependent representative trajectories, in which case the set of representative trajectories will generally be (too) sparse. In addition, as we show in Section 3, time-independent formulations of our problem are already computationally hard, which suggests that the time-dependent problem will be even harder to compute. Hence we model the loop-detector data via **time-independent** flow, which one can interpret as a “long-time average” of the loop-detector measurements.

We formalize the flow as follows: given a network $\mathcal{G} = (V, E)$, a *flow* (f, S, T) on $\mathcal{G}$ is specified by a flow function $f: E \rightarrow \mathbb{R}_{\geq 0}$ mapping each edge of $\mathcal{G}$ to its flow value. In addition, there are sources $S \subseteq V$ and sinks $T \subseteq V$ for the flow. Commonly, a flow function must satisfy the flow-conservation property: for each vertex that is neither a source nor a sink, the sum over all incoming flow is equal to the sum over all outgoing flow. However, errors in actual data may cause violations of the flow-conservation property. Furthermore, traffic volume generally does not specify sources or sinks. We hence introduce the notion of a *flow field* to describe the measured traffic volume, which is a function $\phi: E \rightarrow \mathbb{N}$ (not necessarily satisfying the flow-conservation property).

2.2 Reconstructed routes

Our goal is to compute a multiset $\overline{\mathcal{P}}$ of realistic routes that explains a flow field ϕ well. We represent the multiset $\overline{\mathcal{P}}$ by a base set of routes $\mathcal{P}$ (a *basis* $\mathcal{P}$ for short), along with associated frequency counts. It seems natural to assume that these counts should have integer values. However, with that restriction, even computing the correct counts for a specific basis to explain a given flow is NP-hard [16]. We therefore relax the counts $c: \mathcal{P} \rightarrow \mathbb{R}_{\geq 0}$ to be fractional *coefficients*. This relaxation allows us to efficiently compute the coefficients for a specific basis and flow. We say that the real-valued multiset of routes $\overline{\mathcal{P}} = (\mathcal{P}, c)$ is a *reconstruction* of the flow field ϕ .

We need to quantify how well a reconstruction $(\mathcal{P}, c)$ explains the input flow field ϕ . To this end, we derive a flow $(f_{(\mathcal{P}, c)}, S_{\mathcal{P}}, T_{\mathcal{P}})$ from $(\mathcal{P}, c)$ as follows:

$$\forall e \in E: \quad f_{(\mathcal{P}, c)}(e) = \sum_{P \in \mathcal{P}} M(P, e) c(P)$$

where $S_{\mathcal{P}}$ and $T_{\mathcal{P}}$ contain the start and end vertices of the routes in $\mathcal{P}$, respectively. The error in the loop-detector measurements can be positive or negative, suggesting a measure based on the absolute difference between $(f_{(\mathcal{P}, c)}, S_{\mathcal{P}}, T_{\mathcal{P}})$ and ϕ per edge. In line with traffic-analysis literature [7, 8], we compute the *flow deviation* $\Delta(\mathcal{P}, c, \phi)$ as the sum of squared differences between $(f_{(\mathcal{P}, c)}, S_{\mathcal{P}}, T_{\mathcal{P}})$ and ϕ over all edges.

$$\Delta(\mathcal{P}, c, \phi) = \sum_{e \in E} (\phi(e) - f_{(\mathcal{P}, c)}(e))^2$$

For a fixed basis $\mathcal{P}$, the coefficients c that minimize the flow deviation can be computed efficiently with standard techniques [6, 15].

2.3 Realistic routes

We are given a set $\mathcal{T}$ of trajectories that represent typical behavior of vehicles in the network. Our aim is to compute a realistic reconstruction of the flow field based on $\mathcal{T}$. We measure the realism of a route in the basis $\mathcal{P}$ via its distance to the closest trajectory in $\mathcal{T}$.

There are many possible similarity measures for polylines such as the Fréchet distance [3], the Hausdorff distance, and Dynamic Time Warping [4]. In our setting we might encounter a large difference (either way) in spatial resolution between the road network and the representative trajectories, since the sampling densities of vehicle trajectories differ greatly between providers and sampling technologies used. In the presence of such large discrepancy in sampling density, discrete measures such as Dynamic Time Warping and the discrete versions of the Fréchet distance and Hausdorff distance are known to perform poorly, since measurements have to be matched to vertices of the road network.

The Hausdorff distance and the weak Fréchet distance do not capture the order of points and edges in trajectories and are hence less suitable for our purpose. Hence we choose the strong Fréchet distance to measure the realism of our reconstructed routes: it naturally captures the variability in the paths while encouraging that the general direction of reconstructed routes and representative trajectories are similar.

The (strong) *Fréchet distance* $d_F(P, Q)$ between two curves $P, Q: [0, 1] \rightarrow \mathbb{R}^2$ is defined as

$$d_F(P, Q) = \inf_{\alpha, \beta} \sup_{t \in [0, 1]} \|P(\alpha(t)) - Q(\beta(t))\|,$$

where α and β are reparameterizations of P and Q , respectively. The functions α, β must be strictly monotonically increasing, with $\alpha(0) = \beta(0) = 0$ and $\alpha(1) = \beta(1) = 1$. To determine if two curves lie at Fréchet distance at most ε , we use the so-called *free-space diagram* [3]. This diagram represents matching locations on curves P and Q that are within ε Euclidean distance of each other. Two curves have Fréchet distance at most ε if the diagram admits a strictly monotone path. This problem can be solved in $O(n^2)$ time where n is the total complexity of the two curves [3].

We say that a route is **realistic** if it lies within a prespecified Fréchet distance ε of the closest representative trajectory. The parameter ε controls the realism of our reconstruction. We say that a reconstruction $(\mathcal{P}, c)$ is realistic if all routes $P \in \mathcal{P}$ are realistic.

The routes taken by vehicles tend to be simple, since humans generally take shortest paths to their destinations. Hence we would prefer to reconstruct simple routes only. However, as we show in

Section 3, even just minimizing the flow deviation is NP-hard for simple paths. Our heuristic approaches hence prefer simple routes but do not exclude non-simple ones.

2.4 Formal problem statement

Our complete input is the road network $\mathcal{G} = (V, E)$, the set of representative trajectories $\mathcal{T}$, the flow field ϕ induced by the loop-detector data, and realism parameter $\varepsilon > 0$. We want to find a realistic reconstruction $(\mathcal{P}, c)$ such that the flow deviation $\Delta(\mathcal{P}, c, \phi)$ is minimized. Following Occam's Razor, we are looking for a concise explanation of the flow given the representatives, that is, we prefer reconstructions with small cardinality $|\mathcal{P}|$.

3 COMPUTATIONAL COMPLEXITY

In this section we explore the computational complexity of our problem. First of all, we restrict the reconstructed routes to be simple. In this setting, even computing just a single realistic route is NP-hard [17]. By extension, computing a realistic simple reconstruction is hard as well. Hence, we next consider a restricted variant of the problem, where the reconstructed routes do not have to be realistic and even share start and end point. Specifically, we require that all routes in the reconstruction are simple and start at a vertex s and end at a vertex t . We refer to such a reconstruction as an (s, t) -reconstruction. However, we show that even this simplified problem is NP-hard. For this result, we consider two variants of the deviation function: the sum of squared differences as defined in the previous section, but also the sum of absolute differences. In the following two theorems we refer to these as *squared* and *absolute* deviation, respectively (proofs can be found in Appendix A).

THEOREM 3.1. *Given a road network $\mathcal{G}$ with source s , sink t and an associated flow field ϕ , it is NP-hard to compute an (s, t) -reconstruction with only simple paths that minimizes the absolute deviation to ϕ .*

THEOREM 3.2. *Given a road network $\mathcal{G}$ with source s , sink t and an associated flow field ϕ , it is NP-hard to compute an (s, t) -reconstruction with only simple paths that minimizes the squared deviation to ϕ .*

Consequently, we weaken the requirements even further and study relaxed (s, t) -reconstructions which may contain non-simple routes. We sketch a simple algorithm which approaches a relaxed (s, t) -reconstruction with optimal flow deviation. First, we find the min-cost flow under the flow deviation [23]. Then we construct a so-called pathflows-cycleflows decomposition of the resulting flow by employing a regular flow decomposition algorithm [1]. Now we can construct non-simple routes from the cycles by routing a shortest (s, t) -path via each cycle. By sending a very small amount of flow along these routes and spinning around the cycles many times, we can construct non-simple route flows that come arbitrarily close to the cycle flows they were constructed from. Hence, they also come arbitrarily close to the optimal deviation. The resulting routes are highly non-simple and generally nonsensical.

We now return to the problem as stated in Section 2.4: we reintroduce the requirement that the reconstructed routes must be realistic, while still allowing non-simple routes and dropping the (s, t) -requirement. In this setting, minimizing the flow deviation becomes trivial if ε is large enough: we can use single edges as routes in our reconstruction, trivially covering the entire flow field.

Algorithm 1: FRECHETROUTES($\mathcal{G}, \phi, \mathcal{T}, \varepsilon, i_{\max}$)

Data: Road network $\mathcal{G}$, flow field ϕ , representative trajectories $\mathcal{T}$, threshold ε , number of iterations $i_{\max}$

Result: Reconstruction $(\mathcal{P}, c)$

Initialize residual flow field $\phi_r \leftarrow \phi$; basis $\mathcal{P} \leftarrow \emptyset$;
coefficients $c \leftarrow \emptyset$;

repeat

for $T \in \mathcal{T}$ **do**

$\mathcal{P} \leftarrow \mathcal{P} \cup \text{GENERATEBASISROUTES}(T, \phi_r, \mathcal{G}, \varepsilon)$;

$(\phi_r, c) \leftarrow \text{COMPUTECOEFFICIENTS}(\mathcal{P}, \phi)$;

$(\mathcal{P}, c) \leftarrow \text{PRUNE}(\mathcal{P}, c)$;

until $i_{\max}$ iterations have been performed;

return $(\mathcal{P}, c)$

For smaller values of ε , as of yet we cannot establish the computational complexity. We observe that, already for simple paths, the problem under the absolute and squared deviations is always convex; the difficulty stems from the exponential number of candidate paths to consider for the reconstruction. Furthermore, by Carathéodory's Theorem, we know that the optimal reconstruction contains at most $|E|$ paths. Thus, the difficulty of the problem lies in efficiently searching through the solution space for the best routes. Allowing non-simple routes grows the solution space, but may potentially make it possible to search the space more efficiently. However, we see no reason why this would be the case if the routes must also be realistic. We therefore conjecture that the problem is also NP-hard for non-simple realistic reconstructions.

4 ROUTE RECONSTRUCTION ALGORITHMS

Here we present heuristic approaches that decompose the flow well while following the representative trajectories to varying degrees. In Section 4.1 we describe our iterative Fréchet Routes (FR) heuristics which creates only realistic routes, i.e., routes that have bounded Fréchet distance to a representative trajectory. We discuss two variants (WFR and EFR) which differ in their approach to constructing routes. In Section 4.2 we describe a multi-commodity min-cost flow approach (MCMCF) which is loosely coupled to the representative trajectories, and a global min-cost flow baseline (GMCF) which is essentially agnostic to the representative trajectories.

4.1 Fréchet Routes

Recall that our goal is to find a realistic basis $\mathcal{P}$ and coefficients c such that the flow deviation is minimized. Our Fréchet Routes (FR) heuristic decouples finding the basis and deciding on the coefficients c for a given basis. We grow the basis iteratively, aiming to improve the deviation after each iteration (see Algorithm 1). Since the solution space is infinite, the main challenge is to find a basis that is small enough for efficient computation but comprehensive enough to result in a small deviation from the flow field.

In one iteration of our heuristic we add new routes to the basis for each representative trajectory independently (`GENERATEBASISROUTES`) and evaluate the resulting basis. For a given basis we can compute the coefficients c that minimize the flow deviation efficiently

with standard techniques [6, 15]. We prune the basis by eliminating duplicate routes and routes with coefficient zero. Furthermore, we compute the *residual* flow field $\phi_r: E \rightarrow \mathbb{R}$, defined as $\phi_r(e) = \phi(e) - \sum_{P \in \mathcal{P}} M(P, e)c_P$ for all edges $e \in E$. The residual flow field guides our search for new basis elements.

Generating basis routes. Given the road network $\mathcal{G}$, a single representative trajectory T , the residual flow field ϕ_r (or the flow field ϕ in the first iteration), and threshold ε on the Fréchet distance, we generate basis routes for T as follows. The residual flow field stems from our current reconstruction $(\mathcal{P}, c)$. This reconstruction has an associated deviation $\Delta(\mathcal{P}, c, \phi)$ (this deviation is the flow within ϕ initially). If we extend the basis $\mathcal{P}$ with a path P and an associated positive coefficient c_P , the deviation changes by:

$$\begin{aligned} & \Delta(\mathcal{P} \oplus P, c \oplus c_P, \phi) - \Delta(\mathcal{P}, c, \phi) \\ = & \sum_{e \in P} (\phi_r(e) - M(P, e)c_P)^2 - \phi_r(e)^2 \\ = & -c_P \sum_{e \in P} M(P, e)(2\phi_r(e) - M(P, e)c_P) \end{aligned}$$

For routes that visit an edge at most once (e.g., simple routes), the above simplifies to $-c_P \sum_{e \in P} (2\phi_r(e) - c_P)$. Negative values reduce deviation, so we are particularly interested in capturing edges in routes which have high residual flow $\sum_{e \in P} \phi_r(e)$. Below we describe two different approaches to do so: Edge-inclusion Fréchet Routes (EFR), which selects k new routes per trajectory, and Weighted Fréchet Routes (WFR), which select one new route per trajectory. Both are adaptations of Fréchet map-matching as described by Alt *et al.* [2].

Fréchet map-matching. Here we briefly sketch the algorithm by Alt *et al.*, additional details can be found in Appendix B. The input is a trajectory $T = \langle p_1, \dots, p_l \rangle$, a road network $\mathcal{G}$, and a threshold ε . The output is a route in $\mathcal{G}$ with Fréchet distance at most ε to T . The algorithm uses a *free-space manifold*: a generalization of the free-space diagram that represents matching locations within distance ε between T and locations in $\mathcal{G}$. A monotone path in this manifold corresponds to a route in $\mathcal{G}$ that is within ε Fréchet distance of T . We can determine the existence of such a path by applying a mixture of Dijkstra and a sweepline. Intervals of free space at vertices are *white intervals*, and the algorithm determines their subintervals that are *reachable* with a monotone path. The sweepline processes such reachable intervals in order of their lower endpoint, and follows edges e to the target vertex via monotone paths to find new reachable intervals.

Edge-inclusion Fréchet Routes (EFR). We modify the map-matching algorithm to find a route that must include a specific edge $e = (u, v)$ with high residual flow. In fact, we are attempting to find k routes which include the k edges with the highest residual flow and which each have positive residual flow. To find a route within Fréchet distance ε of the representative trajectory T which contains edge e , we need to find two path in the free-space manifold $\mathcal{F}$: a path from the start to a white interval at u and a path from a white interval at v to the end. Moreover, the concatenation of these two paths with e needs to be monotone in $\mathcal{F}$. To do so, we consider each white interval at u , decide if it is reachable from the start, and if so, continue from all possible white intervals at v .

EFR stops its search in the free-space manifold as soon as the begin/endpoint of the reconstructed route lies within ε distance of the start/end of T . This might ignore flow on edges in the ε -vicinity

of the start and end of T . Hence, we greedily add suitable edges to the ends of the route, taking care not to introduce cycles and to not decrease the average amount of residual flow per edge in the route.

Weighted Fréchet Routes (WFR). EFR uses the residual flow only to indicate the top k routes. We further modify the map-matching algorithm to find routes that generally include edges with high residual flow, that is, high weight. To do so, we maintain a sorted list of weights with each white interval. Let τ be the parameter of trajectory T (intuitively, the height values of the free space manifold $\mathcal{F}$). Each entry in the list is a tuple (τ, ψ) such that there is a path through $\mathcal{F}$ with weight at least ψ (the value of ψ depends on the execution order of Dijkstra's algorithm and is hence only a lower bound). We prune tuples (τ, ψ) whenever there is another tuple (τ', ψ') where $\tau' < \tau$ and $\psi' \geq \psi$. We maintain the weights as we execute the map matching algorithm. In principle, we can construct a high-weight route at the end of the algorithm. However, note that the road network $\mathcal{G}$ can contain cycles which result in cycles between white intervals. Hence we construct a high-weight route explicitly via back-tracking, using each tuple at most once. Note that the resulting route may still contain cycles; we break only cycles in the dependency of white intervals.

4.2 Min-cost flow

We now describe two heuristics that are based on min-cost flow and relax the realism constraint of Fréchet Routes. On a high level, both heuristics follow the same approach: we first solve the min-cost flow problem for the flow field (guided by the representatives to a certain degree) and then we heuristically compute a reconstruction from the resulting flow. Note that our cost function is the (quadratic) deviation of the reconstructed flow from the flow field, which differs from the standard (linear) cost function for min-cost flows.

Multi-commodity min-cost flow (MCMCF). For each representative trajectory T we construct a subgraph $G(T)$ of $\mathcal{G}$ with all vertices and edges within distance ε of T . Vertices within distance ε from the start or end of T can act as sources or sinks of a flow in $G(T)$. Each representative trajectory hence induces a single (min-cost) flow problem. By overlapping the graphs $G(T)$ for all $T \in \mathcal{T}$, we construct a *multi-commodity min-cost flow* problem on $\mathcal{G}$, where each trajectory T has an associated commodity. We can solve the resulting MCMCF using standard software packages (see Section 5).

Global min-cost flow (GMCF). We retain only the sources and sinks of MCMCF and otherwise impose no restriction on the flow. This results in a min-cost flow problem over the entire road network $\mathcal{G}$, which is essentially agnostic to the representative trajectories.

Heuristic path reconstruction. The result of either min-cost flow approach is an edge flow per commodity or over the complete road network. Our goal is to approximate these flows via a reconstruction that may use non-simple paths. For each commodity (or the complete network) we first compute a “path flows-cycle flows” decomposition [1], which is equivalent to the edge flows. We can directly add the resulting path flows to our basis. The cycle flows, however, generally are not correct source-sink paths. We observe that cycle flows which are disjoint from all path flows cannot be close to any of the representative trajectories; we hence exclude them from the basis. All other cycle flows we greedily merge with one of the path-flows which overlap it at one or more vertices. If

the path flow is higher than the cycle flow, then we reduce it to be equal to the cycle flow. If the path flow is lower than the cycle flow, then we traverse the cycle multiple times, using the path flow, to create a consistent (non-simple) route, rounding where necessary.

5 EXPERIMENTAL EVALUATION

We evaluate and compare the various heuristics of the previous section using real-world trajectories. Particularly, we investigate the extensions and parameters of our Fréchet-Routes methods and compare Fréchet Routes to the min-cost-flow-based methods.

We implemented all algorithms¹ in C++ using Boost and MoveTK². For flow problems and determining coefficients c , we use IBM ILOG CPLEX 12.9. We ran all experiments single-threaded on Ubuntu 18.04, on an Intel(R) Xeon(R) Gold 5118 CPU @ 2.30GHz.

Data. To evaluate how well representative trajectories assist in route reconstruction beyond the given sample, we require a ground truth: all driven routes $\mathcal{P}^*$ that together define the flow field ϕ . We use as network $\mathcal{G}$ the roads surrounding The Hague (the Netherlands) extracted from OpenStreetMap [20]; see Fig. 1. As the complete trajectory set $\mathcal{T}^*$, we use 11 445 real-world trajectories provided by HERE Technologies³ in the same area. We map-match $\mathcal{T}^*$ to $\mathcal{G}$ to obtain $\mathcal{P}^*$. To avoid bias, we use [25] instead of [2], as the latter relies on the Fréchet distance and is the basis for our Fréchet Routes. We derive the flow field ϕ for $\mathcal{G}$ by counting the number of occurrences of each edge in $\mathcal{P}^*$. The representative trajectories $\mathcal{T}$ are sampled from $\mathcal{T}^*$, using $\alpha > 0$ such that $|\mathcal{T}| = \lceil \alpha |\mathcal{T}^*| \rceil$.

Measures. To evaluate our heuristics, we use the measures listed below that are generally based on Section 2. Whereas some measures can be used to assess any result, the two measures marked with “GT” rely on having a ground truth. That is, the latter type can measure performance beyond the provided representatives.

flow deviation: how well do the reconstructed routes represent the input flow? We measure the sum of squared differences between input and reconstructed flow, over all edges.

realism: how realistic is the reconstruction? We average the Fréchet distance from each reconstructed route to its closest representative trajectory in $\mathcal{T}$, weighted by the coefficients.

realism (GT): how realistic are the reconstructed routes in relation to the ground truth? This is identical to realism, but we use the closest trajectory in the complete input $\mathcal{T}^*$.

completeness (GT): how well is the ground truth captured by the result? We average the Fréchet distance from each route in the ground truth $\mathcal{P}^*$ to the closest reconstructed route.

cardinality: the number of reconstructed routes.

running time: total computation time (wall clock).

For all measures, lower values indicate better performance. We note that finding a subset of $\mathcal{T}$ with optimal coverage is NP-hard, via a reduction from dominating set for unit-disk graphs [18]. Generally, the data does not admit a solution with perfect coverage or fidelity, as the trajectories are not aligned to the road network. To indicate the distortion inherent in the data due to map matching, we visualize the distortion between $\mathcal{T}^*$ to $\mathcal{P}^*$ in Fig. 2.

Figure 1: Road network of The Hague used in our experiments, 60 277 vertices and 100 654 edges, with color indicating the flow field induced by $\mathcal{P}^*$; gray edges do not contain flow.

Figure 2: Histogram of Fréchet distances between all $T \in \mathcal{T}^*$ and their routes in $\mathcal{P}^*$. The rightmost bar aggregates higher values, which is approximately 3% of the trajectories.

5.1 Fréchet Routes

Here we investigate our Fréchet-Routes algorithm, in terms of candidate-generation methods and its parameters. Throughout this section, we keep ϵ fixed at 100m and run each trial with seven random samples and average the results. Throughout this investigation, we do not consider our realism measures, since FR and its extensions guarantee realism by construction.

Candidate generation. We aim to investigate EFR and WEFR to generate basis routes. Specifically, we run four variants: with weighted routes (WFR), with edge-inclusion (EFR), with both (WEFR) and without either extension (FR). We use $\alpha = 0.05$, $i_{\max} = 8$ and $k = 2$. Fig. 3 summarizes the results. Compared to FR, the extensions have a mild positive effect on completeness and a strong positive effect on deviation, but increase cardinality. Whereas EFR is slightly worse in deviation, completeness and running time than WFR, it performs better in cardinality. Interestingly, WEFR seems to effectively combine these, resulting in deviation and complexity between WFR and EFR and completeness is actually slightly better

¹Source available at <https://github.com/tue-alga/RouteReconstruction>

²<https://movetk.win.tue.nl>

³<https://www.here.com>

Figure 3: Results using variants of Fréchet Routes, from left to right: FR (purple), WFR (red), EFR (orange), WEFR (blue).

Figure 4: Results for WEFR per iteration, with $k = 2$ (blue) and $k = 10$ (green).

Figure 5: Results of WEFR for varying sampling rate α .

than either variant in isolation, suggesting a complementary nature here in obtaining large variation (WFR) and trying to address specific edges with deviation (EFR). The drawback is the increase in running time, since we effectively run both methods in parallel.

Iterations and edge inclusion. For WEFR we investigate the effect of $i_{\max}$, the number of iterations, and k , the number of paths generated by edge inclusion. We run our algorithm with $\alpha = 0.05$ and $k \in \{2, 10\}$ for $i_{\max} = 8$ iterations, recording the result after each iteration. Fig. 4 illustrates the results. The time spent per iteration remains roughly similar, even though cardinality tends to increase. Iterating has a very mild positive effect on completeness and flow deviation. Cardinality increases quickly in the first iterations, but the later ones do not necessarily increase cardinality, as alternative routes are generated that can replace earlier candidates. Compared to $k = 2$, $k = 10$ yields higher cardinality and running time, but improves deviation. The somewhat unstable coverage can be attributed to the use of map matching to obtain a ground truth.

Sampling rate. We now vary α using values in $\{0.05, 0.1, 0.15, 0.2, 0.25\}$. This mimics different degrees of coverage of the representative trajectories relative to the overall traffic. Generally, we may expect that solution quality increases as we have more representative trajectories. Based on the above, we use $k = 2$ but reduce $i_{\max}$ to 5,

since a larger sample implies that more candidates are generated per iteration. The results are shown in Fig. 5. As expected, increasing sampling rate yields better deviation and completeness at the expense of longer computation times and higher cardinality. We observe that completeness seems to not vary much for α between 0.1 and 0.25: there may be outlying behavior in the full dataset which is not readily captured by other traffic, but this may also be in part due to the inherent error in the ground truth (see Fig 2).

5.2 Comparing different methods

We now compare WEFR to the MCF-based (min-cost-flow) methods: GMCF and MCMCF. As before, we use $\epsilon = 100m$ and $\alpha = 0.05$ to get representative trajectories. We repeated the analysis below for seven random samples, but the outcome was similar every time; we hence focus our exposition on one such random sample here.

Table 1 lists measures of the (full) solution for each of the three methods. MCMCF and WEFR are similar in deviation, with MCMCF performing about 10% better than WEFR. Considering cardinality, WEFR achieves this deviation with but a fraction of the routes that MCMCF uses. Interestingly, we see that MCMCF uses roughly the same (but slightly more) routes than in the ground truth. The question is though, how realistic all such routes are. GMCF achieves the lowest deviation, but is also least constrained in terms of realism.

Table 1: Deviation and cardinality for the three different methods on the full solution, as well as on subsets thereof.

		WEFR	MCMCF	GMCF
Full	Deviation ($\times 10^7$)	2.13	1.95	1.07
	Cardinality	926	11553	19336
	Running time (min)	46.6	31.1	43.5
Restricted	Deviation ($\times 10^7$)	2.13	3.35	15.7
	Cardinality	465	638	28
Adjusted	Deviation ($\times 10^7$)	2.13	2.12	14.1
	Cardinality	465	638	28
Reduced	Deviation ($\times 10^7$)	2.13	2.16	
	Cardinality	465	465	

Running times are in the same order of magnitude, with WEFR being slowest and MCMCF being fastest.

In Fig. 6 we relate the coefficient of each reconstructed route to its realism (GT); we use the ground-truth variant which gives a slight advantage to the MCF-based methods, as WEFR constructs realistic routes by definition. We observe that the MCF-based methods include many unrealistic routes, even when measuring with respect to $\mathcal{T}^*$. Also, there are one or two routes with considerable coefficient, and many routes with low coefficients for this dataset.

We thus restrict the solutions. First, we filter to include only reconstructed routes that are deemed realistic with respect to the ground truth, using a threshold of 100m. Motivated by the very high cardinality of the MCF solutions, we also filter to include only routes that have a coefficient of at least 1: that is, we focus on those routes that actually contribute to the flow deviation minimization.

Table 1 lists statistics for these *restricted* solutions. WEFR retains roughly half of its routes (only based on the coefficient). Yet, the deviation is hardly affected. For MCMCF, the cardinality is reduced to roughly 5%, though this increases deviation to be over 50% worse than WEFR. This suggests that MCMCF produces fairly realistic routes, among many unrealistic ones, yet they are not immediately selected to represent the flow field. GMCF reduces to few routes, with excessive deviation: it does not explain traffic flow well with realistic routes and we do not consider it further.

Fig. 6 illustrates also the coefficient-realism relation for WEFR and MCMCF for the restricted solutions. Here, it is evident that WEFR and MCMCF have somewhat similar structure, whereas nearly all routes in the GMCF disappear. Of note is that one of the two high-coefficient routes was filtered out for MCMCF.

By removing routes from the solution, the computed coefficients are no longer optimal for flow deviation. For better understanding the solution, we adjust the coefficients by recomputing them for the restricted set. For these *adjusted* (and restricted) solutions, MCMCF recovers its loss in deviation to be comparable with WEFR; GMCF barely recovers (see Table 1). In Fig. 6 we see that the solution structure resembles the WEFR situation even more.

Figure 6: Relation between realism (GT) in meters on the horizontal axes and the coefficient of each reconstructed route in the solutions of WEFR, MCMCF and GMCF. Here $\epsilon = 100\text{m}$ and $\alpha = 0.05$. (Top) Routes in the full solution. (Bottom) Routes for WEFR and MCMCF after restricting the routes and for MCMCF also after adjusting coefficients.

To assess to what degree the higher cardinality of MCMCF influences the above, we reduce its solution further: we select the 465 highest-coefficient routes, and again adjust the coefficients. For this *reduced* set, we see that the deviation is increased again slightly, to be only slightly worse than WEFR (Table 1). This reinforces the conclusion that MCMCF is able to find realistic routes, but that these are hidden between the unrealistic ones. Though filtering as above is possible, it provides computational overhead compared to using WEFR which directly guarantees realism.

6 CASE STUDY

We present a case study using real-world loop-detector data, obtained from the Dutch National Traffic Dataportal⁴. Using this loop-detector data and the full trajectory set $\mathcal{T}^*$ described in the previous section, we apply the different approaches to reconstruct routes from these loop detectors. Fig. 7 illustrates the locations of the loop detectors and their amount of flow for a single hour during the day the trajectory dataset was recorded. To acquire a (partial) flow field out of the detector data, we assign their flow to the nearest edge in the road network, selecting the maximum flow when multiple detectors match to the same edge.

Incomplete flow. As Fig. 7 illustrates, the loop detectors do not cover all edges in the road-network and it hence gives an incomplete flow. Hence, we adapt the deviation measure to include only edges that have a loop detector and we adapt the non-negative least squares accordingly. This does not fundamentally alter the described approaches. One consequence of this approach is that there are (sub)routes in the network that do not affect flow deviation, and thus may lead to unnatural routes which include small cycles and detours. EFR does not suffer from this problem as the underlying algorithm [2] tends to create short paths. For WFR we discourage the use of edges without a loop detector by giving them a small negative residual value. For MCMCF one could consider to also include some form of minimization of the flow on edges not having a loop detector; however, preliminary tests suggest that this is not effective; we leave improving realism for MCMCF on incomplete flows to future work. Finally, we do not modify GMCF as it does not exhibit this problem and, as we shall see, such an approach may even exacerbate the issues of this method.

Analysis. Fig. 7 shows the reconstructed routes as the derived flow field for all algorithms, and Table 2 shows relevant quality measures. It is clear in Fig. 7 that GMCF very greedily constructs routes: the routes are mostly short disjoint pieces that cover loop detector flow. For WEFR and MCMCF we see longer routes, extending beyond the immediate vicinity of the loop detectors: this we can attribute to the

stricter realism requirements which cause *entire* (mostly) realistic routes to be selected. This is further supported by the deviation and realism of the different approaches (Table 2): GMCF performs best regarding deviation, but at the expense of low realism. WEFR and MCMCF perform better here, where WEFR balances the realism and deviation best of the two. This can be attributed to the more targeted search of WEFR for high weight routes.

Contrasting our previous experiments with complete flows, WEFR now actually achieves better flow deviation than MCMCF, in spite of MCMCF having higher cardinality. The drawback is that WEFR scales less well with the higher number of representative trajectories, taking about five times as long as MCMCF – but reducing the number of iterations may partially alleviate this drawback.

7 DISCUSSION

Our work shows that the studied problem is challenging even in “simple” forms, but our evaluation shows promising results. As such, it leads to various avenues for further research.

Representatives. Our experiments show that more representatives can improve the performance of route reconstruction. Yet, this also comes at a computational cost. However, this is mostly as it diversifies behavior: adding very similar representatives will not improve the results drastically, if at all. As our methods do not require that the representatives are actually part of the traffic generating the flow data, we could preprocess representatives using clustering and central trajectories to reduce computation time of our algorithms. Furthermore, we could use the information from such clustering methods, or from map-matching accuracy, to vary the threshold ϵ per representative, to relate realism to the uncertainty in the data. We leave to future work to investigate how such techniques affect efficiency and quality.

Reconstruction. MCMCF does not guarantee a bound on the Fréchet distance, and we observed that its very large basis contains many routes that exceeded the given realism parameter. We showed that we can post-process the routes to only include realistic ones to make the solution more realistic, but this increases computation time and flow deviation. An interesting direction of research could be to incorporate this directly into the route-reconstruction phase, to avoid or at least reduce this overhead.

As demonstrated, our techniques can be adapted to handle cases where not all edges have associated flow. We observed a need to incorporate information on the roads without flow data, to avoid unnatural cycles and detours in the reconstructed routes. Requiring strict simplicity may eliminate cycles, yet it is unlikely to fully address the problem and makes the problem significantly harder. Hence, we may want to consider other models to further restrict what defines a realistic route, beyond the Fréchet distance used here.

Time-varying flow data. We considered only flow data that is static: an edge has a single value associated with it (if any) representing the amount of traffic in a certain time interval. However, traffic changes over time and hence flow data is time-varying: can we extend our techniques to cope well for such data? Challenges lie in the increased computational complexity, but also on how to incorporate temporal aspects of the representative trajectories, which may not even align in time with the flow data.

Table 2: Quality measures for the case study.

	WEFR	MCMCF	GMCF
Deviation ($\times 10^7$)	3.89	4.03	3.07
Cardinality	109	646	271
Realism (m)	82.4	98.2	503.7
Running time (min)	153.2	26.9	1.3

⁴<https://ndw.nu/en/>

Figure 7: Case study: (Top) real-world loop-detector measurements and representative trajectories. (Bottom) Results of WEFR, MCMCF, and GMCF, visualized as the induced flow field derived from the routes and their coefficients. Here, again $\epsilon = 100m$.

ACKNOWLEDGMENTS

B. Custers supported by HERE Technologies and NWO (628.011.005).

REFERENCES

- [1] R. K. Ahuja. *Network Flows: Theory, Algorithms, and Applications*. Pearson Education, 1993.
- [2] H. Alt, A. Efrat, G. Rote, and C. Wenk. Matching planar maps. *J. of Alg.*, 49(2):262–283, 2003.
- [3] H. Alt and M. Godau. Computing the Fréchet distance between two polygonal curves. *Int. J. of Comput. Geom. & Appl.*, 5(01n02):75–91, 1995.
- [4] D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *Proc. KDD Workshop*, vol. 10(16), pages 359–370, 1994.
- [5] Q. W. Bouts, I. Kostitsyna, M. J. van Kreveld, W. Meulemans, W. Sonke, and K. Verbeek. Mapping polygons to the grid with small Hausdorff and Fréchet distance. In *Proc. European Symp. on Alg., LIPIcs* 57, pages 22:1–16, 2016.
- [6] R. Bro and S. De Jong. A fast non-negativity-constrained least squares algorithm. *J. of Chemometrics*, 11(5):393–401, 1997.
- [7] P. Cao, T. Miwa, T. Yamamoto, and T. Morikawa. Bilevel generalized least squares estimation of dynamic origin–destination matrix for urban network with probe vehicle data. *Transp. Research Record*, 2333(1):66–73, 2013.
- [8] E. Cascetta. Estimation of trip matrices from traffic counts and survey data: a generalized least squares estimator. *Transp. Research Part B: Methodological*, 18(4-5):289–299, 1984.
- [9] E. Castillo, A. J. Conejo, J. M. Menéndez, and P. Jiménez. The observability problem in traffic network models. *Computer-Aided Civil and Infrastructure Eng.*, 23(3):208–222, 2008.
- [10] M. Duckham, M. J. van Kreveld, R. Purves, B. Speckmann, Y. Tao, K. Verbeek, and J. Wood. Modeling checkpoint-based movement with the earth mover’s distance. In *Proc. GIScience*, pages 225–239, 2016.
- [11] T. Hartman, A. Hassidim, H. Kaplan, D. Raz, and M. Segalov. How to split a flow? In *Proc. IEEE INFOCOM*, pages 828–836, 2012.
- [12] J.-H. Haunert and B. Budig. An algorithm for map matching given incomplete road data. In *Proc. ACM SIGSPATIAL GIS*, pages 510–513, 2012.
- [13] H. Hu and D.-L. Lee Gamma: A framework for moving object simulation In *Proc. SSTD*, pages 37–65, 2005.
- [14] G. R. Jagadeesh and T. Srikanthan. Online map-matching of noisy and sparse location data with hidden Markov and route choice models. *IEEE Trans. on Intel. Transp. Sys.*, 18(9):2423–2434, 2017.
- [15] D. Kim, S. Sra, and I. S. Dhillon. Tackling box-constrained optimization via a new projected quasi-Newton approach. *SIAM JOSC*, 32(6):3548–3563, 2010.
- [16] K. Kloster, P. Kuinke, M. P. O’Brien, F. Reidl, F. S. Villaamil, B. D. Sullivan, and A. van der Poel. A practical FPT algorithm for flow decomposition and transcript assembly. In *Proc. Workshop on Alg. Eng. and Exp.*, pages 75–86, 2018.
- [17] M. Löffler and W. Meulemans. Discretized approaches to schematization. In *Proc. Canadian Conf. on Comput. Geom.*, 2017.
- [18] S. Masuyama, T. Ibaraki, and T. Hasegawa. The computational complexity of the m-center problems on the plane. *IEICE Trans.*, 64(2):57–64, 1981.
- [19] P. Newson and J. Krumm. Hidden Markov map matching through noise and sparseness. In *Proc. ACM SIGSPATIAL GIS*, pages 336–343, 2009.
- [20] OpenStreetMap contributors. Planet dump retrieved from <https://planet.osm.org>. <https://www.openstreetmap.org>, 2020.
- [21] M. A. Quddus, W. Y. Ochieng, and R. B. Noland. Current map-matching algorithms for transport applications: State-of-the art and future research directions. *Transp. Research Part C: Emerging Technologies*, 15(5):312–328, 2007.
- [22] B. Vatinlin, F. Chauvet, P. Chrétienne, and P. Mahey. Simple bounds and greedy algorithms for decomposing a flow into a minimal set of paths. *European J. of Oper. Research*, 185(3):1390–1401, 2008.
- [23] L. A. Végh. A strongly polynomial algorithm for a class of minimum-cost flow problems with separable convex objectives. *SICOMP*, 45(5):1729–1761, 2016.
- [24] C. Wenk, R. Salas, and D. Pfoser. Addressing the need for map-matching speed: Localizing global curve-matching algorithms. In *Proc. Int. Conf. on Sci. & Stat. Database Management*, pages 379–388, 2006.
- [25] C. Yang and G. Gidofalvi. Fast map matching, an algorithm integrating hidden Markov model with precomputation. *Int. J. of GIS*, 32(3):547–570, 2018.

A OMITTED PROOFS

THEOREM 3.1. *Given a road network $\mathcal{G}$ with source s , sink t and an associated flow field ϕ , it is NP-hard to compute an (s, t) -reconstruction with only simple paths that minimizes the absolute deviation to ϕ .*

PROOF. We show that it is NP-hard to determine whether there exists an (s, t) -reconstruction with simple paths such that the absolute deviation $\Delta_{\text{abs}}(\mathcal{P}, c, \phi) \leq \delta$ for some $\delta \geq 0$.

Our proof uses a reduction from the longest-path problem: given a graph $\mathcal{G}' = (V', E')$, a source s' and sink t' in V' , and a threshold $L > 0$, decide whether $\mathcal{G}'$ admits a simple path of at least L edges from s' to t' . We turn this into an instance of our problem as follows. We augment the network to $\mathcal{G}$ by adding a new vertex s connected by a new path of $L - 1$ edges to s' . For the sink, we use the same vertex, $t = t'$. The flow field ϕ has value 1 for all edges in E' , and value 0 otherwise. Finally, we set the deviation threshold at $\delta = |E'| - 1$. Our claim is that this instance admits an (s, t) -reconstruction with absolute deviation at most δ , if and only if $\mathcal{G}'$ admits a simple path of length at least L from s' to t' .

Assume that $\mathcal{G}'$ admits a simple path of length at least L from s' to t' . Let P' denote this path, and P the route in $\mathcal{G}$ consisting of the path from s to s' concatenated with P' . Consider the reconstruction consisting only of P with coefficient 1. The result is an (s, t) -reconstruction with only simple paths by construction. For every edge e originally from P' we have that $|\phi(e) - \sum_{P \in \mathcal{P}} M(P, e)c(P)| = 0$; for every other edge this value is 1. Thus, the absolute deviation $\Delta_{\text{abs}}(\mathcal{P}, c, \phi)$ is the number of edges along the path from s to s' plus the number number of edges in $\mathcal{G}'$ not covered by P' . Hence, this is at most $(L - 1) + (|E'| - L) = |E'| - 1 = \delta$.

Now, assume that an (s, t) -reconstruction $(\mathcal{P}, c)$ exists with absolute deviation at most δ . Specifically, we assume that $\Delta_{\text{abs}}(\mathcal{P} \setminus \{P\}, c, \phi) > \Delta_{\text{abs}}(\mathcal{P}, c, \phi)$ for all routes $P \in \mathcal{P}$. In other words, removing any route increases the deviation. Observe that a reconstruction without any routes would achieve deviation $|E'| > \delta$, thus the reconstruction must contain at least one route. Consider a route $P \in \mathcal{P}$. As P must start at s and end at t , and the flow field at all edges between s and s' is zero, we know that removing P from the solution locally decreases the deviation by $(L - 1) \cdot c(P)$. Since removing a path must increase deviation and the deviation at one edge can increase by at most $c(P)$, P must contain at least L edges originating from $\mathcal{G}'$. Hence, the subpath of P starting at s' and ending at t' has at least L edges. As a result, we conclude that $\mathcal{G}'$ has a simple path of length at least L .

Finally note that the construction can easily be performed in polynomial time, so the stated problem is NP-hard. $\square$

THEOREM 3.2. *Given a road network $\mathcal{G}$ with source s , sink t and an associated flow field ϕ , it is NP-hard to compute an (s, t) -reconstruction with only simple paths that minimizes the squared deviation to ϕ .*

PROOF. We again use a reduction from the longest path problem. Let the instance of the longest path problem consist of a graph $\mathcal{G}' = (V', E')$ along with source s' , sink t' , and threshold L . We augment $\mathcal{G}'$ to obtain $\mathcal{G} = (V, E)$ as follows. We first add a new path of $L - 1$ edges from the new source vertex s to s' , and we refer to this set of new edges as E_1 . Furthermore, we add a new path of $L - 1$ edges from s' to t' , and we refer to this set of edges as E_2 . Hence we have $E = E' \cup E_1 \cup E_2$. For the sink, we use the

same vertex $t = t'$. For the flow field ϕ we set $\phi(e) = 0$ if $e \in E_1$, $\phi(e) = 1$ if $e \in E'$, and $\phi(e) = 2$ if $e \in E_2$. We now claim that the minimum deviation for the instance formed by $\mathcal{G}$ and ϕ is $|E|$ if the longest simple path in $\mathcal{G}'$ has length at most $L - 1$, and the minimum deviation is strictly smaller than $|E|$ if there exists a path of length at least L in $\mathcal{G}'$. This directly implies that the stated reconstruction problem is NP-hard.

First assume that the longest simple path in $\mathcal{G}'$ has length at most $L - 1$. Consider the (s, t) -reconstruction $(\mathcal{P}, c)$ consisting of a single path $P^* \in \mathcal{P}$ with $P^* = E_1 \cup E_2$ and $c(P^*) = 1$. Note that the deviation of this reconstruction is exactly $|E|$. Now consider any simple path P between s and t and add it to $\mathcal{P}$ with $c(P) = 0$ (this does not really change the decomposition). By definition, the derivative of the deviation with respect to the coefficient $c(P)$ is given by:

$$\begin{aligned} \frac{\partial \Delta(\mathcal{P}, c, f)}{\partial c(P)} &= \sum_{e \in E} -2M(P, e)(f(e) - \sum_{P_i \in \mathcal{P}} M(P_i, e)c(P_i)) \\ &= 2 \sum_{e \in E} M(P, e)(M(P^*, e) - f(e)) \\ &= 2(|P \cap E_1| - |P \cap E_2| - |P \cap E'|) \end{aligned}$$

The last step in the above is due to $M(P^*, e) - f(e)$ being 1 for $e \in E_1$ and -1 otherwise. Note that $|P \cap E_1| = L - 1$ by construction. As any simple path P either passes through $\mathcal{G}'$ or is equal to P^* , precisely one of the other terms is zero. If $P = P^*$, then $|P \cap E_2| = L - 1$ and thus the derivative is zero. Otherwise, P passes through $\mathcal{G}'$ and thus $|P \cap E'| \leq L - 1$. Hence, the derivative with respect to $c(P)$ is always non-negative. Since $c(P)$ cannot become smaller than zero, we conclude that $(\mathcal{P}, c)$ is a local minimum for the deviation function. Since both the constraints and the deviation function are convex, this local minimum must also be the global minimum.

Now assume that there exists a simple path P' in $\mathcal{G}'$ with length at least L . As above, let $P^* = E_1 \cup E_2$, and let $P = E_1 \cup P'$. Consider the (s, t) -reconstruction $(\mathcal{P}, c)$ with $\mathcal{P} = \{P^*, P\}$, $c(P^*) = 1$, and $c(P) = \epsilon$. Note that the edges in E_2 contribute exactly $|E_2|$ to the deviation. For an edge $e \in E_1$, the deviation is $(1 + \epsilon)^2$, and for an edge $e \in P'$ the deviation is $(1 - \epsilon)^2$. In total, the edges in E' contribute at most $|E'| - L + L(1 - \epsilon)^2$ to the deviation. We obtain the following total deviation over all edges:

$$\begin{aligned} \Delta(\mathcal{P}, c, f) &\leq |E_2| + |E'| - L + L(1 - \epsilon)^2 + (L - 1)(1 + \epsilon)^2 \\ &= |E| - 2L + 1 + L(1 - \epsilon)^2 + (L - 1)(1 + \epsilon)^2 \\ &= |E| - 2L + 1 + 2(L - 1)(1 + \epsilon^2) + (1 - \epsilon)^2 \\ &= |E| - 1 + 2(L - 1)\epsilon^2 + (1 - \epsilon)^2 \\ &= |E| + (2L - 2)\epsilon^2 - 2\epsilon + \epsilon^2 \\ &= |E| + \epsilon((2L - 1)\epsilon - 2) \end{aligned}$$

Finally, by choosing $\epsilon = 1/(2L - 1)$, we obtain that $\Delta(\mathcal{P}, c, f) = |E| - 1/(2L - 1) < |E|$, which concludes the proof. $\square$

B FRÉCHET MAP-MATCHING

Alt *et al.* [2] present an algorithm to map-match a trajectory $T = \langle p_1, \dots, p_l \rangle$ to a road network $\mathcal{G}$: the result is a route in $\mathcal{G}$ such that its Fréchet distance to T is at most ϵ . This decision version can

subsequently be used to find the route that minimizes the Fréchet distance to T , but we need only this decision algorithm.

Let the allowed Fréchet distance be fixed to ε , and let T be parameterized on $[1, l]$ such that for $\tau \in [i, i+1]$, $T(\tau) = p_i + (\tau - i)(p_{i+1} - p_i)$. Furthermore, we assume for simplicity here that $\mathcal{G}$ is the subset of the overall network that is fully covered by the Minkowski sum of a disk of radius ε with the trajectory T .

Trajectory T , network $\mathcal{G}$ and ε define a free-space manifold $\mathcal{F}$. This manifold defined as the locations in the direct product space $(\tau, r) \in T \times \mathcal{G}$ such that $d(T(\tau), r) \leq \varepsilon$. A monotone path through the free space on this free-space manifold from some point with $\tau = 1$ to some point with $\tau = l$ then matches to a route in $\mathcal{G}$ with Fréchet distance at most ε . Note that the monotonicity avoids moving backwards along an edge, but an edge may be visited multiple times.

First, for each vertex v of $\mathcal{G}$, the algorithm computes a 1-dimensional free-space diagram $FD(v): [1, l] \rightarrow \{0, 1\}$. In this diagram, intervals that map to value 1 are called *white intervals*. The white intervals mark the parameter values of τ on T for which v is within Euclidean distance ε and thus a potential match.

Then, for each edge (u, v) in $\mathcal{G}$, the algorithm computes the left-right pointers for all white intervals I of $FD(u)$. These left-right pointers mark a range of $FD(v)$ that can be reached from I by a monotone path in the free space of the connecting 2-dimensional free-space strip defined by (u, v) and T . Due to convexity any point in the free space of $FD(v)$ between these extremal left-right pointers

can be reached, but note that this range may encompass multiple white intervals at v .

To determine whether a route exists within Fréchet distance ε , the algorithm now applies a sweep-line algorithm over the parameter space defined by τ , starting at 1. For every vertex v , it maintains a list $C(v)$ of ranges in $FD(v)$ that can be reached by a monotone path in the free-space manifold defined by $\mathcal{G}$ and T . That is, the intersection of $C(v)$ with the white intervals of $FD(v)$ define reachable intervals. In particular, this list contains the ranges such that the last part of the matching path ends at or goes through the sweep-line value in the dimension of τ .

The events that the sweep line handles are when the start of a reachable white interval I in some $C(u)$ is reached that was previously not discovered yet. This event is handled by updating all $C(v)$ for all v such that (u, v) in E , taking into account the new intervals that can be reached from I . These can be efficiently retrieved via the precomputed left-right pointers.

The sweep line stops as soon as a white interval is added to some $C(v)$ that contains parameter value l or no white intervals are available anymore. The former implies the existence of a route within Fréchet distance ε , whereas the latter implies that such a route does not exist. To reconstruct the route, the algorithm keeps track of predecessor vertices whenever the white intervals in $C(v)$ are updated when handling a white interval I . In total, the algorithm runs in $O(l(V + E))$ time, assuming efficient bookkeeping.