## Learn global and optimize local

*A data-driven methodology for last-mile routing*

Ghosh, M.; Kuiper, A.; Mahes, R.; Maragno, D.

**DOI**
10.1016/j.cor.2023.106312

**Publication date**
2023

**Document Version**
Final published version

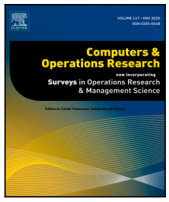
**Published in**
Computers and Operations Research

Computers & Operations Research 159 (2023) 106312

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor



# Learn global and optimize local: A data-driven methodology for last-mile routing

Mayukh Ghosh [a,1], Alex Kuiper [a,*,1], Roshan Mahes [a,b,1], Donato Maragno [a,1]

[a] Amsterdam Business School, University of Amsterdam, 1018 TV Amsterdam, The Netherlands
[b] Korteweg–de Vries Institute for Mathematics, University of Amsterdam, 1098 XG Amsterdam, The Netherlands

ARTICLE INFO

*Keywords:*
Last-mile logistics
Route prediction
Navigation
Traveling salesman problem
Data-driven optimization

ABSTRACT

In last-mile routing, the task of finding a route is often framed as a Traveling Salesman Problem to minimize travel time and associated cost. However, solutions stemming from this approach do not match the realized paths as drivers deviate due to navigational considerations and preferences. To prescribe routes that incorporate this tacit knowledge, a data-driven model is proposed that aligns well with the hierarchical structure of delivery data wherein each stop belongs to a zone — a geographical area. First, on the global level, a zone sequence is established as a result of a minimization over a cost matrix which is a weighted combination of historical information and distances (travel times) between zones. Subsequently, within zones, sequences of stops are determined, such that, integrated with the predetermined zone sequence, a full solution is obtained.

The methodology is particularly promising as it propels itself within the top-tier of submissions to the *Last-Mile Routing Research Challenge* while maintaining an elegant decomposition that ensures a feasible implementation into practice. The concurrence between prescribed and realized routes underpins the adequateness of a hierarchical breakdown of the problem, and the fact that drivers make a series of locally optimal decisions when navigating. Furthermore, experimenting with the balance between historical information and distance exposes that historic information is pivotal in deciding a starting zone of a route. The experiments also reveal that at the end of a route, historical information can best be discarded, making the time it takes to return to the station the primary concern.

## 1. Introduction

Last-mile routing is often the shortest, yet most complex and expensive leg in distribution systems. It accounts for about 28% of the total costs of transportation from the distribution center to the final customer (Goodman, 2005; Gevaers et al., 2011). Last-mile operations also have a large impact on livability, because of traffic congestion and pollution it creates (Taniguchi and Thompson, 2002; Chopra, 2003). This pressure is set to increase because the trend of urbanization continues; around 68% of the global population will reside in urban areas by 2050, according to a report by the United Nations (2018). Currently e-commerce sales are estimated at about $709.78 billion, which is projected to increase as the coronavirus has accelerated a channel shift to e-commerce, see Lipsman and Liu (2020). The same trends are echoed by a recent report of the World Economic Forum (Deloison et al., 2020), which further points out that also the number of same-day deliveries will grow due to customers' ever-increasing expectations, amplifying the importance of last-mile operations and its impact on the ecosystem.

Considering parcel delivery in last-mile operations, the main goal is to ensure that products arrive in good shape. In addition, it should be on time (Gunasekaran et al., 2004), i.e., the time windows that have been communicated will be met. As routes require a multitude of stops to be visited, it is critical that proposed routes are followed, because deviations can cause redeliveries or dissatisfied customers (by not meeting time slots), unforeseen time losses along the route, and additional operational expenses (Boyer et al., 2009; Gevaers et al., 2014).

In spite of the benefits of adhering to a proposed 'optimal' route being clear, in practice — reported by managers in last-mile delivery (e.g., Amazon) — drivers often deviate from proposed routes. This discrepancy between *theoretical* route planning and *real-life* route execution is an important gap in the literature on last-mile routing and has remained an open research question, because of the typical inaccessibility of real-life data. This is underscored by Amazon's vice president for Last Mile Delivery, who identifies the disconnect as:

* Corresponding author.
*E-mail addresses:* m.ghosh@uva.nl (M. Ghosh), a.kuiper@uva.nl (A. Kuiper), a.v.mahes@uva.nl (R. Mahes), d.maragno@uva.nl (D. Maragno).
[1] These authors contributed equally.

https://doi.org/10.1016/j.cor.2023.106312
Received 18 January 2023; Received in revised form 6 June 2023; Accepted 12 June 2023
Available online 15 June 2023

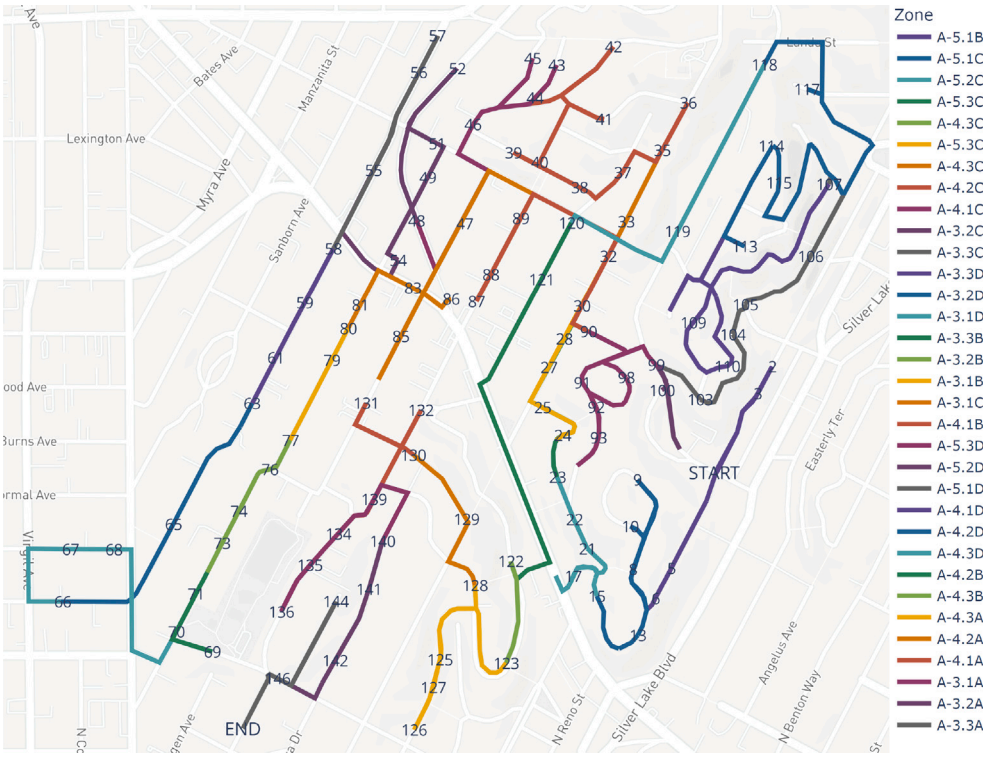



**Fig. 1.** Example of a route from the Amazon dataset, wherein the numerals denote the order of stops visited, and the colors indicate the distinct zones traversed within the route. The image is generated by using OpenStreetMap (2022).

*"Despite the tremendous advances in routing optimization over the last decade, there remains an important gap between periodic route planning and real-time route execution"* (Gantenbein, 2021). Therefore, recently, Amazon in combination with MIT's Center for Transportation & Logistics hosted a challenge (Amazon and MIT Center for Transportation & Logistics, 2021a), wherein by means of offering a vast routing dataset, researchers around the world have been invited to bridge this gap. In the US, Amazon steadily holds the number one position as the largest e-commerce retailer having a market share of 38.0% (Lipsman and Liu, 2020), and inherently is responsible for the majority of traffic associated with last-mile delivery. Therefore, the data associated with Amazon serves as a benchmark for many other organizations facing the bewildering setting of last-mile delivery.

In this work, we answer that call by providing a complete methodology that proposes routes that adequately reflect drivers' tacit knowledge and preferences by learning from historic route data. The foremost purpose of the methodology is to have better predictions about the routes taken by the drivers, allowing more reliable delivery times. Second is to have a better understanding of how human navigation and information at the ground level can be integrated into an optimization framework. Moreover, we emphasize that having an approach that follows the driver's logic can be used as a *tool* to prescribe routes. These routes can be 'better' and more feasible from the driver's perspective as critical elements on the ground level might be absent in routing software or traditional optimization frameworks.

To delve deeper into the problem, we present a historical route extracted from the dataset provided by Amazon, illustrated in Fig. 1. Each route comprises a sequence of stops, where each stop is associated with a specific zone. The zones are visually distinguished using different colors, while the stops are designated by numerical identifiers. Some numerals have been omitted in the figure for clarity. It is important to note that the zones should be understood as neighborhoods, which could be formed, for example, by clustering ZIP codes. However, the exact method of generating these zones for the Amazon data, as in Fig. 1, is not disclosed (Merchán et al., 2022). Ideally, a zone should be designed such that all stops within it are visited before the driver moves to the next zone, as also illustrated by Fig. 1.

Last-mile routing data is challenging to analyze, let alone to learn from; practically, each route is unique. Leveraging historical routes to identify recurring patterns is facilitated by analyzing the data at the zone level. This has several advantages:

(i) Learning at the zone level makes use of aggregated data, which is easier to manage and allows for a focus on higher-level patterns.
(ii) A zone is a (potential) collection of neighboring stops that share the same spatial context.
(iii) Zones are commonly encountered across multiple routes, whereas stops are likely to appear infrequently or not at all, facilitating better learning.
(iv) Learning at the zone level protects the privacy of customers, as personal or sensitive details about individual stops are not used.

The contributed methodology divides the problem into two parts: on the zone level, it aims to infer the sequence of zones after which, within each zone, the sequence of stops is determined. It stems from the idea that first a routing problem is abstractified into zones, for which a sequence is established by solving a *Traveling Salesman Problem* (TSP) (Flood, 1956) formulation that balances historical zone transition information, resembling the tacit knowledge, and distances. So, the methodology learns on a global level, which aligns with the intuition that humans congregate route sequences on a higher level than remembering exact stop sequences. Secondly, the problem in each zone is of relatively low complexity and thus a driver is capable to determine an optimal sequence out of the set of stops that minimizes distance or travel time.

In the next section, we provide the literature review, which outlines valuable elements of our methodology and reports various findings related to human navigation. Section 3 describes the full methodology and is split into learning and prediction. In Section 4, we introduce the dataset on which the methodology will be applied and provide the route prediction metrics. These metrics are aligned with the challenge.

Using the case study data we carry out various experiments and demonstrate the effectiveness of the methodology. Furthermore, we compare the performance to the aforementioned benchmarks as well as to the submissions of the competition. In Section 5, we present our concluding remarks and implications for practice, followed by a discussion and suggestions for future research.

## 2. Literature review

Solving a last-mile routing problem can be translated to a TSP, i.e., a set of stops that should be visited, which start from and end at the station. However, in this research, the goal is to predict a driver's route using historic data. Therefore, in this pursuit, elements of different disciplines are combined. More specifically, we consider classical TSP optimization frameworks, hierarchical structures as an approximation (such as cluster first, routing second), human approaches to solving TSPs, and lastly, learning from route data. In relation to our work, we elaborate further on these themes below.

### 2.1. Traveling salesman problem

The TSP with Euclidean distance is notorious for being hard and is proven to be NP-complete (Papadimitriou, 1977), but for finding the optimal or near-optimal tour there is a rich literature (Lawler, 1985; Applegate et al., 2011). Also, several variations of the standard TSP have been introduced in the literature, to name a few: an open TSP (OTSP) where the driver does not necessarily return to the depot; TSP with time windows to meet (Gendreau et al., 1998); additionally allowing pick-ups such that the capacity of the truck is not exceeded (Gendreau et al., 1999); or the covering Salesman problem (CSP) (Current and Schilling, 1989), where a minimal tour is found such that all stops are in the vicinity of a stop on the tour (e.g., dropping goods at neighbors).

Acquainted to the TSP is its generalization known as *Vehicle Routing Problem (VRP)*; see Dantzig and Ramser (1959) for its formulation and Bräysy and Gendreau (2005a) and Montoya-Torres et al. (2015) for recent surveys on multiple depots VRPs with variants that include time windows, split delivery, heterogeneous fleet, periodic deliveries, and pick-up. Also, various studies have considered the ecological footprint as presented in the literature review given by Lin et al. (2014). Finally, also for the more general VRP, numerous heuristics and metaheuristics have been proposed over the last decades (Bräysy and Gendreau, 2005b; Speranza and Archetti, 2014; Vidal et al., 2013; Purkayastha et al., 2020).

Both TSP and VRP have seen many different integer programming formulations (Orman and Williams, 2007). In our model, we adopt the Miller–Tucker–Zemlin (MTZ) formulation to solve the TSP, introduced for the first time by Miller et al. (1960). The MTZ formulation has the advantage of being intuitive and straightforward to implement and works well when the number of stops (variables) to consider is relatively small, fitting our instances. As a downside, it has been proven to lead to a weak relaxation, leading to higher computation times, see e.g., Campuzano et al. (2020) for a discussion. In our case, as we break down the full problem into smaller ones, this is not an issue, but as desired any other TSP formulation can be adopted.

### 2.2. Hierarchical structure

To reduce the computational complexity, the problem can be broken down into smaller instances, which are computationally much less involved, and connected together. For example, in the seminal work by Karp (1977), the proposed partitioning algorithms are asymptotically optimal heuristics in the case of uniformly distributed stop locations, i.e., the error tends to zero with probability one as the length of a route (i.e., the number of stops) increases. Using the explicit zone structure in the formulation of the TSP is done in Chisman (1975) and is better known as the clustered traveling salesman problem. In this optimization problem, the order of the stops and clusters is determined by adhering to the constraint of first visiting all stops in a cluster before moving to the next.

Both Liao and Liu (2018) and Jiang et al. (2014) decompose the problem with small-scale nodes by relying on clustering algorithms. Each subproblem is subsequently optimized and the center nodes of the subproblems constitute a TSP in itself. Connecting all local tours in the order of the upper layer problem generates approximative solutions with significantly reduced computation times. Such an approach is also found from an empirical point of view, see Vickers et al. (2003).

Moreover, evidence to support the use of a hierarchical structure in practice is found in Graham et al. (2000). They show that solution methods originating from artificial intelligence or operations research algorithms are insufficiently capable to mimic the human approach of solving TSPs. They introduce a hierarchical model by means of a pyramid algorithm on the visual representation. This algorithm is capable of rendering human-alike solutions to the various TSP instances where classical algorithms fail in this task; see also Pizlo et al. (2006) for a more refined algorithm. More importantly, the aforementioned works hint at the use of a hierarchical breakdown because of lower computational complexity and being concurrent with practice.

### 2.3. Human navigation

The TSP lends itself to a broad range of experimental studies as its goal of minimization of the route is easy to understand and visualize. Many experimental studies have been devoted to visual versions of the TSP (MacGregor and Ormerod, 1996; Van Rooij et al., 2003; Vickers et al., 2003). MacGregor and Ormerod (1996) show that humans outperform well-known TSP heuristics and are in small problem sizes capable to be in 1% from optimality.

Humans rely on various tactics to generate near-optimal solutions for the TSP. The potential of these tactics is further demonstrated by the fact that the time needed increases only in a linear fashion compared to the problem size (Pizlo et al., 2006; Dry et al., 2006). One of these tactics is to consider the problem first globally, considering the tour that visits all 'exterior' points, to which interior points are inserted; this resembles the so-called convex hull approach (MacGregor and Ormerod, 1996; Vickers et al., 2003). Argued as the underlying motivation for the convex hull approach is the avoidance of crossings in a tour (Van Rooij et al., 2003). The motivation behind this tactic is the intuition that a cross in a tour is suboptimal, which is even a fact for metric TSPs. In an OTSP, where the solution is not a tour, the starting point can have a profound impact on the performance, see Sengupta et al. (2018). However, they also report that humans are quite capable of selecting a 'right' starting point.

Wiener et al. (2009) find in a series of experiments that, when navigating, a coarse route is stipulated first. This route visits a set of 'regions' after which it is 'optimized' on a detailed level along the way. Furthermore, if the problem size increases, the problem is divided into more regions to make the problem manageable and approachable. Such a global-to-local approach echoes the hierarchical decomposition of first determining the zone sequence on a global level and, subsequently, a series of local problems to determine the sequence in which stops are to be visited in each zone. Additionally, another experiment from Wiener and Mallot (2003) reveals that segmentation into zones affects route planning and navigational behavior as it primes a driver to approach the problem from such a structure. A similar effect can be expected when drivers are presented a route, for example, a solution from routing software as in the Amazon challenge, from which we take the data to demonstrate our methodology (Amazon and MIT Center for Transportation & Logistics, 2021a). Finally, in practice, both Ceikute and Jensen (2013) and Toledo et al. (2013) report that, beyond minimizing travel time or distance, several other factors impact drivers' decisions on which route they take.

### 2.4. Route prediction

Typically, the problem of finding a sequence of stops is positioned in the field of combinatorial optimization, but in this research, the goal of finding an optimal sequence to minimize an objective function does not take center stage. Here, the goal is to find out how route data can be used to predict routes that drivers (will) follow; for an example where learning takes place in order to find the optimal route, see Kool et al. (2019).

The capability to learn and predict (part of) the routes is demonstrated in Krumm (2008). The model is trained from drivers' long-term trip history using GPS data. It uses a Markovian approach, i.e., on the basis of the last road segment, which can also be adjusted to incorporate more history, it predicts the next segment the driver will take. In the same vein, Ye et al. (2015) propose a route prediction method based on a hidden Markov model that can accurately predict an entire route early in the trip. Wang et al. (2015) also employ a Markov model; their algorithm relies on a probability transition matrix that is developed to represent the knowledge of the driver's preferred links and routes. For the VRP, Canoy and Guns (2019) show the potential of using a Markov model in an optimization framework. By constructing a transition probability matrix, based on historical data, and by exploiting the VRP structure, they render solutions that resemble actual route plans much better than relying on a distance metric.

### 2.5. Contribution

Our contribution is a complete methodology to predict routes that drivers follow. It breaks the route prediction problem down into optimization problems on two levels to ensure that elements of human navigation (Sections 2.2 and 2.3) are adequately reflected. The previous works demonstrate that a Markov model is a powerful tool to learn from historical data and to use these in predictions. We adopt such a model by using historical information, but in contrast to Canoy and Guns (2019), we balance it against distances (travel times) to come to a cost matrix to be optimized. So our hierarchical decomposition is such that learning takes place only on the higher level and near-optimal decisions in terms of minimizing distance are taken on the local level.

The approach does not require the derivation of new or complex routing routines, because the problem on the higher level is sufficiently aggregated and the series of problems on the local level are small and sequentially solved. Although the approach seems to follow a *divide-and-conquer* strategy (Karp, 1977; Cesari, 1996), it cannot be executed in parallel as the decision problems on the local level depend on each other sequentially. Fortunately, all local problems can be solved to optimality by using any standard TSP formulations, see Section 2.1. Moreover, by using Amazon's dataset, which contains a vast number of drivers' routes, we find convincing evidence that many of the claims about human navigation in the literature reported for *toy problems* also hold in real-life last-mile delivery. Therefore, this research is one of the first to quantify and develop a methodology that researches and bridges the gap between route optimization in theory and execution in practice.

## 3. Methodology

The methodology that we propose relies on the fourfold elements reviewed in Section 2. We impose a hierarchical structure on two levels: finding the zone sequence on a global level and finding the stop sequence within each zone locally. Such an approach is in line with evidence from human navigation, and more importantly, matches the structure of typical routing data — each stop belongs to a zone. We *learn* on the global level the preference to traverse from one zone to another by adopting a Markov model that is built on counting zone-to-zone transitions in the historical route data. Next, combining this transition matrix with the distances (travel times) between zones we form a cost matrix for the global problem. Relying on standard methods, i.e., the *Miller-Tucker-Zemlin* formulation of the TSP (Miller et al., 1960), we solve the TSP on the global level to determine the zone sequence. So, the objective of this TSP reflects a weighted combination of preferences (learned information) and distance or travel time. Next, a series of TSPs on the local level are solved that solely focus on minimizing the travel time. This mimics the driver's capabilities to find near-optimal solutions in small instances.

Tracking zone-to-zone transitions captures the implicit preferences of drivers. In relation to zones, there are two phases, which form the basis of our approach:

- *Learning.* Extracting zone sequences from historical data enables the counting and collection of zone transitions in a count matrix.
- *Prediction.* For new routes, a combination of a distance and count matrix is used in a TSP formulation which solution provides the zone sequence; after which within zones the stop sequences are iteratively determined by means of a series of OTSPs.

### 3.1. Learning zone preferences

To learn the zone preferences of drivers, the focus lies on obtaining the driving patterns at the zone level. As the data provides the stop sequence per route, we have to convert it to a higher-level sequence of zones. As a driver tends to first finalize a zone before moving to another (Section 2.3), we want to distill the intrinsic sequence of unique zones.

Each route consists of an ordered series of $n+1$ stops, namely $n$ delivery nodes (this number differs per route) and the station. Therefore, a route is represented as a sequence of tuples, $\texttt{route} = ((x_0, y_0, z_0), \dots, (x_n, y_n, z_n))$, with $(x_i, y_i) \in \mathbb{R}^2$ indicating the stop's geographical location, and $z_i$ the corresponding zone id, with $z_i \in \{Z_1, \dots, Z_m\}$ for $i \in \{0, \dots, n\}$.

It can, however, happen that a driver returns to a zone already visited. Fortunately, we find that these deviations sporadically occur, see also the introduction. Moreover, if they occur, the impact on the performance is minor. The cases in which they occur are likely the stops that are close to a zone's boundary or an idiosyncrasy. In all cases, we propose a procedure that obtains the intrinsic zone sequence that is able to deal with these revisitations. To explain our procedure, consider a route with $n$ delivery nodes. First, we consider the sequence of zones for each stop $(z_1, \dots, z_n)$. Then, the route's zone sequence is mapped to the sequence of pairs $((\zeta_1, \tau_1) \dots, (\zeta_\ell, \tau_\ell))$, where each pair $(\zeta_i, \tau_i)$ represents how many nodes $(\tau_i)$ belonging to the same zone $(\zeta_i)$ are visited in a row. When the $\zeta_i$ are pairwise distinct, we consider $(\zeta_1, \dots, \zeta_\ell)$ to be the desired sequence of unique zones.

**Example 1.** The following route consisting of six stops $(z_1, \dots, z_6)$ lying in three zones $Z_1$, $Z_2$, and $Z_3$, which is reduced into a sequence of three unique zones:

$$(z_1, z_2, z_3, z_4, z_5, z_6) = (Z_1, Z_1, Z_1, Z_2, Z_3, Z_3) \rightarrow$$
$$\big((\zeta_1, 3), (\zeta_2, 1), (\zeta_3, 2)\big) \rightarrow (\zeta_1, \zeta_2, \zeta_3) = (Z_1, Z_2, Z_3).$$

◊

In case of a zone's reoccurrence ($\zeta_i = \zeta_j$ for an $i < j$), we only keep the occurrence with the highest number of stops by comparing $\tau_i$ to $\tau_j$ or keep the earliest appearance in case of a tie, here that is $\zeta_i$. This procedure is repeated until we are left with a sequence of singly occurring zones. Next, two extreme examples illustrate some minutiae of the procedure.

**Example 2.** Still six stops situated over three zones; first consider the sequence

$$(Z_3, Z_1, Z_1, Z_2, Z_3, Z_3) \rightarrow ((\zeta_1, 1), (\zeta_2, 2), (\zeta_3, 1), (\zeta_4, 2)) \rightarrow (\zeta_2, \zeta_3, \zeta_4)$$
$$= (Z_1, Z_2, Z_3).$$

As seen zone $Z_3$ ends last as $\tau_1 = 1$ and $\tau_4 = 2$. In the next case, zone $Z_1$ occurs thrice, but breaking the tie twice puts zone $Z_1$ at its first occurrence in the sequence instead:

$$(Z_1, Z_3, Z_1, Z_2, Z_2, Z_1) \rightarrow ((\zeta_1, 1), (\zeta_2, 1), (\zeta_3, 1), (\zeta_4, 2), (\zeta_5, 1)) \rightarrow$$
$$(\zeta_1, \zeta_2, \zeta_4) = (Z_1, Z_3, Z_2).$$

◊

Evidently, not all zones are visited within each route, therefore we keep track of all $m$ zones that have been visited at least once in any route. Having reduced each route to its corresponding zone sequence, we compute an asymmetric count matrix $N$ where each entry $N_{ij}$ essentially represents the number of times a driver went from the $i$th to the $j$th zone so that $i$ and $j$ range from 1 to $m$. In addition, the count matrix also tracks the station-to-zone and zone-to-station transitions to arrive at an $m$+$s$ matrix, where $s$ is the number of stations in the dataset, so $s = |S|$. The pseudocode related to these procedures is provided in the Appendix.

### 3.2. Predicting the zone sequence

In the next phase, we predict the route that a driver will take. For that purpose, we translate the count matrix, created in the previous phase, to a transition matrix $P$ via $P_{ij} = \frac{N_{ij}}{\sum_{j=1}^{m} N_{ij}}$, so that $P_{ij}$ reflects the Markovian probability of transitioning from zone $i$ to zone $j$.

Besides the zone sequences, the locations of the station and zones are necessary ingredients in the suspected trade-off a driver makes. For this purpose, we define the center of a zone, if not given, by computing the mean latitude and longitude of all stops that were visited in that zone. Having the zone centers' location, we can compute the distances $\delta_{ij}$ between zone $i$ and zone $j$. We normalize the distances by dividing each element through the largest, i.e., $\max_{i,j} \delta_{ij}$, and store these distances in a matrix $D \in \mathbb{R}^{(s+m)\times(s+m)}$.

As an alternative to the Euclidean distances in $D$, a sensible choice is to use the travel times (or another measure for costs between stops) as a link to measure 'distance' between zones. However, in the case when we are only given route data, we can proceed with a two-step procedure to have an approximation of the travel times between zones. Since stops are typically assigned to zones, we first identify for each zone the closest stop within the route to the zone center by using the Euclidean distance metric. Second, once each zone has a stop assigned, we extract the travel times between these stops to compose a matrix $T$ which consists of elements $T_{ij}$ corresponding to normalized travel times from station/zone $i$ to $j$, i.e., $T_{ij} = \frac{t_{ij}}{\max_{i,j} t_{ij}}$ where $t_{ij}$ are the provided travel times. The normalizations are needed to balance it against the transition matrix that consists of probabilities between 0 and 1.

Since we are to minimize a cost matrix, we reverse the transition matrix to $1 - P_{ij}$ in the cost matrix to arrive at the following linear combination with $\omega \in [0, 1]$ being a weight parameter:

$$C_{ij} = \omega D_{ij} + (1 - \omega)(1 - P_{ij}), \tag{1}$$

where $D_{ij}$ can be interchanged with $T_{ij}$ — we will compare the performance of both options — and additionally we set the diagonal entries $C_{ii} = 0$. In fact, this problem can be considered as a TSP with unknown costs as we do not a priori know how distance is evaluated against the likelihood of zone-to-zone transitions, which contain the implicit tacit knowledge and drivers' preferences. Remark that a good value of $\omega$ is still to be determined. Next, we are able to use this cost matrix in a TSP formulation, which optimal solution constitutes a tour through all zones in which one or more stops are located. For details about the implementation, we refer the reader to the Appendix.

### 3.3. Predicting the stop sequence

At this point, we have a sequence of zones that yet has to be transformed into a sequence of stops, e.g., we are in Fig. 2(a) where the zone sequence is $(Z_A, Z_B, Z_C)$. As the problem size in each zone is considerably smaller, drivers are able to make (near-)optimal navigational decisions that minimize travel time or distance, see Section 2.3. To mimic that behavior, we employ again the framework for solving TSPs, but with the additional flexibility to be an open TSP (OTSP) as we do not need to close the loop within a zone, but rather traverse from one zone to another. Although there are several options available, the procedure detailed in Fig. 2 is in accordance with practice. The approach builds on two elements, which are applied iteratively in each zone:

- The last stop of a previous zone (or station, Fig. 2(b)) is used as the starting point of the OTSP for the current zone.
- An additional stop of the next zone is added to each OTSP to add a sense of direction to find a good stop to finish its zone.

As seen in Fig. 2(a), the extra stop (not filled) is selected as the one closest (in terms of Euclidean distance) to the computed zone centers (×). Naturally, after the solution for an OTSP is found with the additional stop added, the last stop is removed from the solution such that for the subsequent zone the starting point is again a stop in its previous zone. The procedure continues until the last zone is reached for which the station is used as the additional 'stop'.

Because of the clarity of the exposition, we considered Euclidean distances in the figure, but the implementation can also be fed with travel times. The resulting solution thus forms a logical sequence from a navigational point of view as it avoids path crossings and mimics human behavior when solving (O)TSPs, cf. Section 2.3. A full implementation of the approach is given in the Appendix and is tailored such that it fits the dataset corresponding to the Amazon Last Mile Research Challenge. Also, we use this dataset to show the adequacy of the approach, as the achieved performance puts the approach among the top-tier submissions for this challenge.

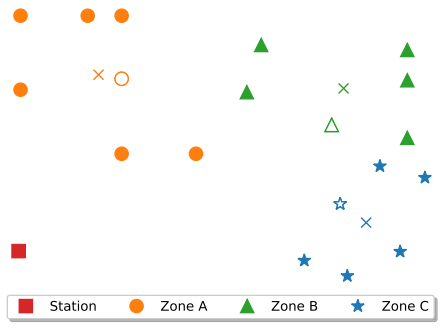
## 4. Performance evaluation

To assess the performance of the proposed methodology we employ it on a routing dataset, which was part of the Amazon Last Mile Research Challenge (Amazon and MIT Center for Transportation & Logistics, 2021a). Also, the metrics that we use are adopted from this challenge and are provided in Section 4.1.

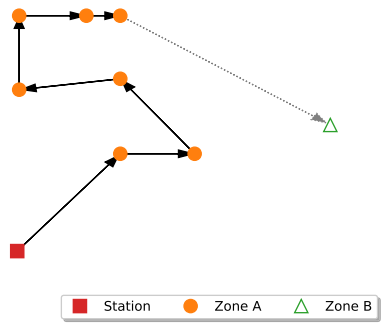
The dataset counts 6112 routes from 17 different stations. We split the data such that 5112 routes are used for learning and 1000 routes are reserved as an out-of-sample test set. Furthermore, we impute missing zone ids by taking over the id of the closest stop in terms of travel time, which is a sensible choice from the driver's perspective. The experiments related to our methodology are run using an Intel Core i7-10850H 2.70 GHz CPU and 32 GB RAM. The learning takes approximately 1 min and with a test set of 1000 routes, it takes on average 30 min to render all routes. However, for choosing the right value of $\omega$ for Eq. (1) and reconsidering it in 4.3, we rely on 5-fold cross-validation over the training set of 5112 routes.

### 4.1. Route prediction metrics

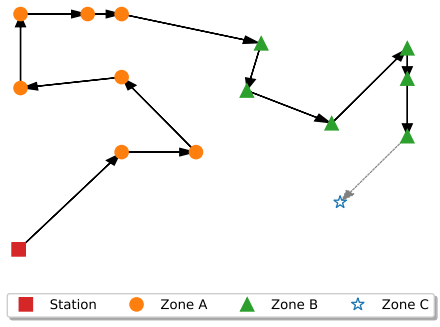
The route prediction performance can be evaluated in two dimensions, that is, how *often* and by how *much* a prediction $B$ deviates from the realized stop sequence $A := (0, 1, \ldots, n)$, and thus $B$ is, in essence, a permutation of $A$. Below, we describe these different components and how they together form an overall route performance metric.



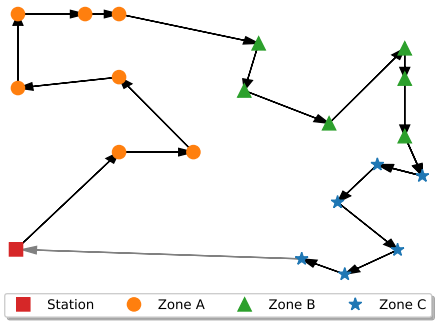

(a) Overview of stops with zone centers (×).




(b) Solution of the OTSP for zone A.




(c) Solution of the OTSP for zone B.




(d) Solution of the OTSP for zone C.

**Fig. 2.** An example of the iterative procedure that is applied on (a), wherein for each zone, subfigures (b), (c), and (d), an open traveling salesman problem is solved locally.

- The *sequence deviation* $\mathbf{SD}_{\mathbf{stop}}(A, B)$ measures the difference in stop sequences $A$ and $B$. Given that in both cases all $n+1$ stops have been visited, we take the stop sequence of $B$, and for each position, we trace back when it has been visited in $A$ to create a vector $(a_0, a_1, \dots, a_n) = \pi(A)$ so that

$$\mathbf{SD}_{\mathbf{stop}}(A, B) := \frac{2}{n(n-1)} \sum_{i=1}^{n} \left( |a_i - a_{i-1}| - 1 \right).$$

  Note that, if $A = B$, the deviation indeed becomes 0. Lastly, due to the fact that we deduce the zone sequence from a route by running the procedure outlined in Section 3.1, we can also compute the zone sequence deviation $\mathbf{SD}_{\mathbf{zone}}(A, B)$ that on the higher level compares sequence deviations between two zone sequences $A$ and $B$.
- Besides considering the position in the sequence, one can also count the number of edit operations needed to come from the predicted route to the realized route, familiarly known as the *Levenshtein* distance as introduced by Levenshtein (1966). This distance metric, renamed to $\mathbf{ERP}_{\mathbf{edit}}(A, B)$, counts the number of deletions and insertions to get the same sequence.
- Apart from considering the concurrence and concordance of stop sequences, the *size of deviation* between two sequences $A$ and $B$ should be evaluated as well. To this end, normalized travel time is introduced to evaluate the difference between two routes on a stop basis; let $A[i]$ and $B[j]$ be the $i$th stop of $A$ and the $j$th stop of $B$, with $i, j = 0, 1, \dots, n$, then it is defined as

$$\mathbf{time}_{\mathbf{norm}}(A[i], B[j]) := Y_{A[i],B[j]} - \min_{i,j} Y_{A[i],B[j]},$$

$$Y_{A[i],B[j]} := \frac{t_{A[i],B[j]} - \bar{t}}{\mathrm{std}(t)},$$

  where $t_{A[i],B[j]}$ is the travel time from stop $A[i]$ to stop $B[j]$, and $\bar{t}$ and $\mathrm{std}(t)$ are the mean and standard deviation over all travel times between the stops in the route. Considering per stop the difference between two sequences, we compute the so-called Edit Distance with Real Penalty component ($\mathbf{ERP}_{\mathbf{norm}}$). Formally, it is computed as

$$\mathbf{ERP}_{\mathbf{norm}}(A, B) := \sum_{i=0}^{n} \mathbf{time}_{\mathbf{norm}}(A[i], B[i]).$$

  As an aside, dividing $\mathbf{ERP}_{\mathbf{norm}}(A, B)$ by $\mathbf{ERP}_{\mathbf{edit}}(A, B)$ translates to the average additional travel time incurred by deviating, and is called $\mathbf{ERP}_{\mathbf{ratio}}(A, B)$.

Besides these separate metrics, they are combined to create a comprehensive score

$$\mathbf{route\_score} = \frac{\mathbf{SD}_{\mathbf{stop}}(A, B) \cdot \mathbf{ERP}_{\mathbf{norm}}(A, B)}{\mathbf{ERP}_{\mathbf{edit}}(A, B)},$$

so that a performance score is obtained by averaging over all $I$ routes to be predicted:

$$\mathbf{Performance} = \frac{1}{I} \sum_{i=1}^{I} \mathbf{route\_score}_i. \tag{2}$$

Evidently, a lower route score means that the predicted sequence coincides more with the realized sequence; 0 means they are exactly the same.

### 4.2. Choosing the weight

As a first experiment, we consider varying the weight parameter $\omega$ in Eq. (1). Furthermore, we can replace distance by travel times $T$ as these are provided in Amazon's dataset. The performances when varying $\omega$ in both versions are displayed in Fig. 3.

As expected, we find that only relying on historical information ($\omega = 0$), or only using distance or travel times ($\omega = 1$) does not perform

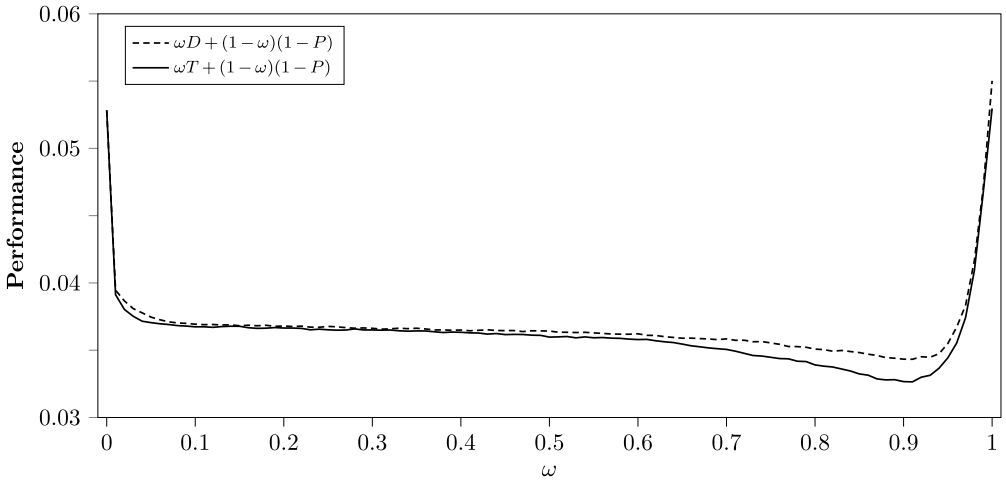



**Fig. 3.** Comparison of performances, as defined in Eq (2), when varying $\omega$ in Eq. (1), where Euclidean distance is dashed and (expected) travel time solid. The curves are obtained by averaging over the 5-fold cross-validation scores.

as well as having a trade-off between the two. Overall, we observe that an $\omega$ value around 0.9 achieves the best trade-off; actually any weight between 0.1 and 0.9 in Eq. (1) (dashed and solid lines) will result in a performance of around 0.035. This observation shows that there is a trade-off between historical information and distance, but the exact specification of it is of lesser importance. In addition, it demonstrates the robustness of the model as being insensitive to the misspecification of $\omega$. In addition, we conclude that the difference between travel times or Euclidean distance is small, and thus distance can be used as a good proxy for travel time.

### 4.3. Adjusting the station's weights

Reconsidering Eq. (1), wherein we define an $\Omega$ matrix to weigh historical information against travel times, we further study the role of the station as the starting and ending point of any route. For the sake of readability, we consider a matrix $\Omega^{(r)}$ which contains the weights in $\Omega$ corresponding to the station and zones that are visited in a specific route $r$ where the station is related to the first row and column. So as to refine the role of the station, we define $\Omega^{(r)}$ matrix as

$$\Omega^{(r)} = \left(\begin{array}{c|ccc} 0 & \omega_F & \dots & \omega_F \\ \hline \omega_L & \omega_Z & \dots & \omega_Z \\ \vdots & \vdots & \ddots & \vdots \\ \omega_L & \omega_Z & \dots & \omega_Z \end{array}\right),$$

where $\omega_F \in [0,1]$ is the weight in the cost function from the station to the *first* zone. Analogously, $\omega_Z \in [0,1]$ sets the balance for *zone-to-zone* transitions, and *lastly* $\omega_L \in [0,1]$ is the weight corresponding to the return to the station. The elements $\Omega_{ij}$ from this matrix are put in the following refinement of Eq. (1) with travel times:

$$C_{ij} = \Omega_{ij}\, T_{ij} + (1 - \Omega_{ij})(1 - P_{ij}), \tag{3}$$

where, similarly to Eq. (1), the diagonal values ($C_{ii}$)are set to zero. With this generalized cost matrix, we first set $\omega_L = 1$ as we anticipate that at the end of a route heading back to the station should be the only concern, and thus the number of times that a specific zone acted as the finish point of a route can be disregarded. This resonates with the thought that a driver is only focused at the end on the travel time it takes to return to the station. Having this parameter restrained, we study combinations of $\omega_F$ and $\omega_Z$ to obtain the contour plot in Fig. 4. In the plot, we observe that the optimal point should lie somewhere in the region where $\omega_F \in [0.1, 0.2]$ and $\omega_Z \in [0.7, 0.8]$. So when starting a route in the first transition from the station, historical information should be weighted more importantly than regular zone-to-zone transitions. It implies that preferences and navigational considerations are more dominant at the start of a route.

The choice of $\omega_L = 1$ is justified by the degradation in performance for values less than 1. This is shown in Fig. 5 by taking four combinations of $(\omega_F, \omega_Z)$ that lie around the inner contour — close to the optimum. In all, when comparing the points in detail, we conclude that the best performance in our cross-validation is obtained when $(\omega_F, \omega_Z, \omega_L) = (0.2, 0.8, 1)$.

### 4.4. Comparison against benchmarks

In Table 1, we provide the performance of our approach abbreviated to *LG-OL* and compare it to the solutions generated via the Nearest Neighbor algorithm and solving a TSP on the entire set of stops. Besides the metrics outlined in Section 4 we also added the mean travel time in seconds in the **Time** column. The results are obtained on the out-of-sample test set of 1000 routes.

In Fig. 6, we report the route generated by the (a) Nearest Neighbor heuristic and by solving the (b) TSP for the same set of stops used in Fig. 2. On the one hand, the Nearest Neighbor algorithm sequentially adds a connection from the last stop to the nearest unvisited stop; a strategy that is sometimes considered to model human route planning behavior (Graham et al., 2000; Wiener et al., 2009). On the other hand, the route generated by solving a full TSP is generally harder to compute, but within the range of state-of-the-art solvers (e.g., Gurobi). Surprisingly, the driver's average travel time (9,496 s) is about 18% higher than when travel time minimization was the only objective (Full TSP, see Section 2.1); indeed it shows that drivers' navigational decisions cannot be approached as a standard optimization problem.

Our methodology obtains a prediction performance (**Performance**) of 0.0299 — a more than 64% reduction compared to the aforementioned benchmarks. Also, the average (expected) travel time is reduced by 5% compared to the actual average travel time of the drivers. Interestingly, the Nearest Neighbor and full TSP models have similar $\mathbf{ERP_{ratio}}$ scores, but the full TSP renders better sequences (lower $\mathbf{SD_{zone}}$ and $\mathbf{SD_{stop}}$ scores). In Fig. 7, the distribution of the prediction performance of our methodology is compared to that of the above-mentioned benchmarks. We find that our model succeeds to have 40% of the test routes to be near-perfectly predicted, as they are below 0.01; and even gets 80% of them below the 0.05 threshold. In retrospect, reconsidering Fig. 3 we find that even for any $\omega$ the unadjusted approach outperforms these benchmarks, supporting the fundamental idea of our learn global, optimize local approach. Furthermore, to verify the potential of the approach, we did an additional hypothetical experiment on the test set of which the results are provided in Table 1 as *LG-OL (hypothetical)*. We extracted the zone sequences using Section 3.1 on the test set so as to consider the situation that we are fully capable to predict the

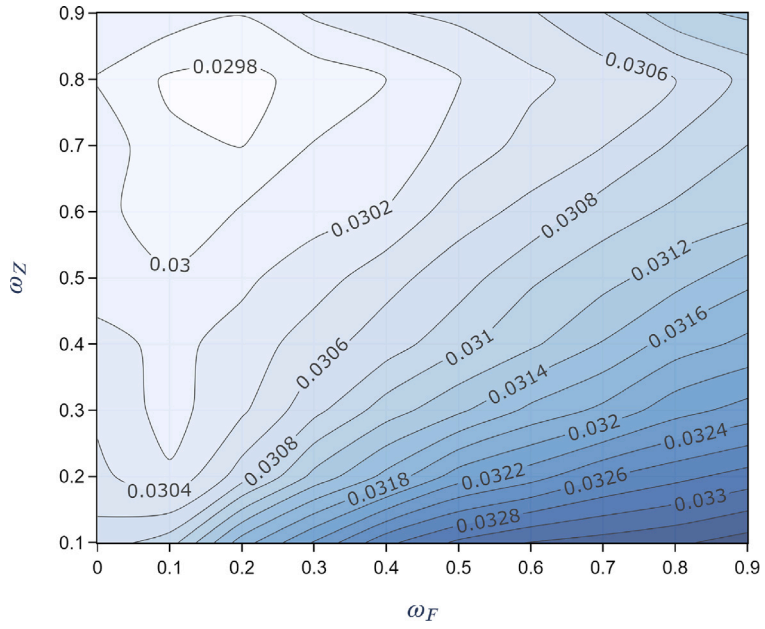



**Fig. 4.** A contour plot showing the performance across different weightings of $\omega_F$ (station-to-zone) and $\omega_Z$ (zone-to-zone) transitions, while keeping $\omega_L = 1$ (zone-to-station) in Eq. (3). The values are obtained averaging over the 5-fold cross-validation scores.

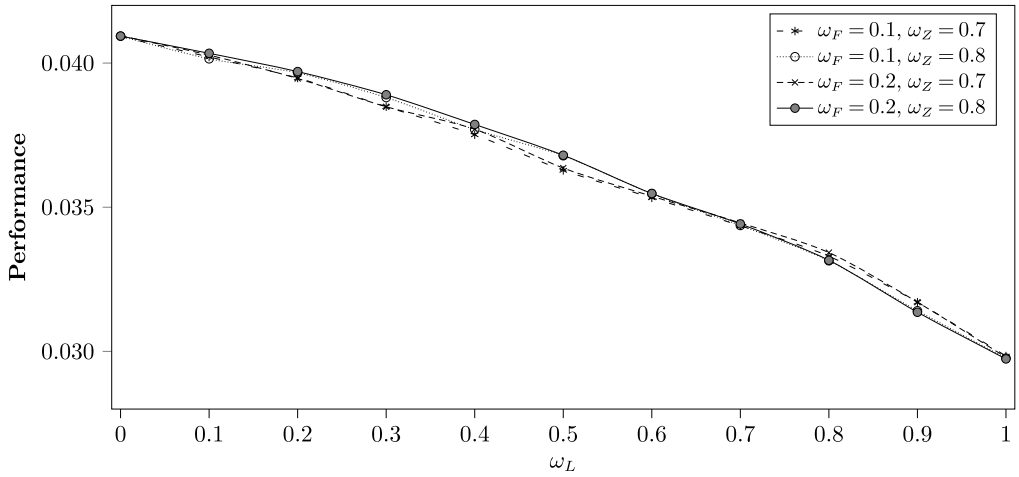
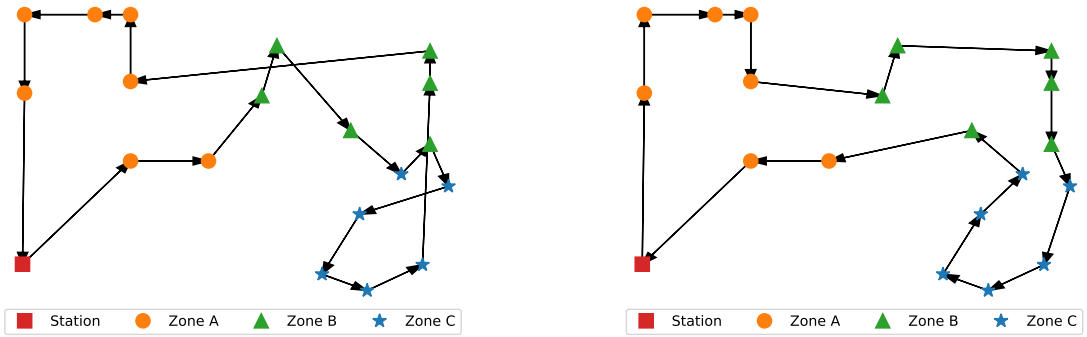



**Fig. 5.** A sensitivity study on $\omega_L$, (zone-to-station), for selected combinations of $\omega_F$ and $\omega_Z$ that lie around the inner contour of Fig. 4. The performances, as defined in Eq. (2), are obtained by averaging over the 5-fold cross-validation scores.




(a) Nearest Neighbor solution. (b) TSP solution.

**Fig. 6.** An example of a route generated using the (a) Nearest Neighbor algorithm and solving the (b) Traveling Salesman Problem to optimality with respect to the traveled distance.

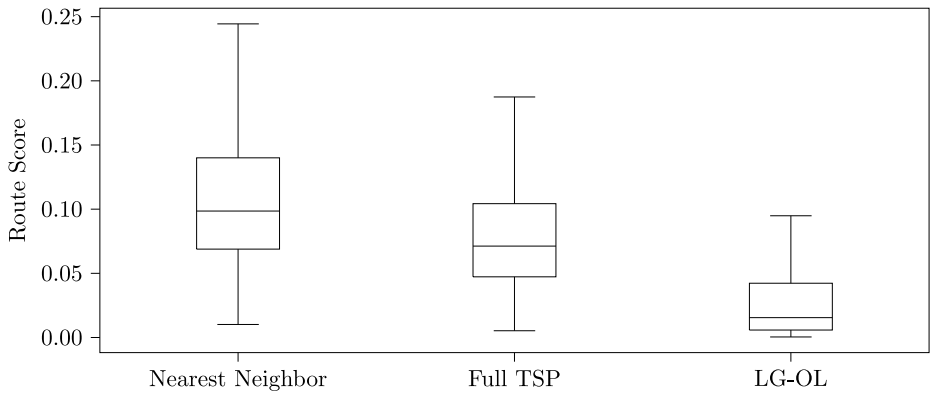



**Fig. 7.** Boxplot representing the route score distribution on the out-of-sample test set of 1000 routes for our model, and the two benchmarks of Table 1.

**Table 1**
Comparison of the methodology against benchmarks on the out-of-sample test set.

| Model | $SD_{zone}$ | $SD_{stop}$ | $ERP_{ratio}$ | Time (s) | Performance |
|---|---|---|---|---|---|
| Nearest Neighbor | 0.2441 | 0.0873 | 1.3382 | 9865 | 0.1119 |
| Full TSP | 0.1912 | 0.0650 | 1.3330 | 8036 | 0.0826 |
| LG-OL | 0.0940 | 0.0335 | 0.7389 | 8994 | 0.0299 |
| *LG-OL (hypothetical)* | 0 | 0.0238 | 0.3831 | 9535 | 0.0094 |
| *Driver* | 0 | 0 | 0 | 9496 | 0 |

**Table 2**
Leaderboard of the Last-Mile Routing Research Challenge.

| Team | Rank | Score | Model Build (s) | Model Apply (s) |
|---|---|---|---|---|
| Just Passing Through | 1 | 0.0248 | 109 | 12095 |
| ***LG-OL*** | (⋆) | 0.0300 | 195 | 4615 |
| Permission Denied | 2 | 0.0353 | 27 | 1825 |
| Sky is the Limit | 3 | 0.0391 | 6 | 12080 |
| MEGI | 4 | 0.0436 | 566 | 2128 |
| UPB | 5 | 0.0484 | 5 | 10434 |

zone sequence right. Thus remaining are the inconsistencies between the design choices made in Section 3 and drivers' behavior that is not reflected in the methodology. In the experiment, we find that the performance becomes 0.0094, which is well below all submissions for this challenge. The difference between *LG-OL* and its hypothetical counterpart can be attributed to insufficient learning and/or limitations of using a Markov model.

Finally, in order to evaluate our performance against the submissions for the challenge (Amazon and MIT Center for Transportation & Logistics, 2021b), we ran our model on an m5.4xlarge instance of Amazon's EC2 servers, consisting of 16 CPUs @3.1 GHz and 64 GB RAM, adhering to the technical specifications of the Amazon challenge. The model was built on all 6112 routes and evaluated on the test set containing 3052 new routes, see Merchán et al. (2022). Table 2 compares the scores of the top performing teams in the challenge, where Model Build refers to the duration of the learning phase in seconds (cf. Section 3.1) and the Model Apply refers to the duration of the evaluation on the 3052 unseen routes (cf. Sections 3.2 and 3.3). The proposed approach propels itself into the top-tier of submissions, after the submission of Cook et al. (2022).

## 5. Conclusion and discussion

The costs associated with last-mile delivery surmount any other shipping costs. Hence, good predictions about the route taken are crucial from an operations management point of view. However, there is a large gap between theoretical route optimization models, e.g., optimal solutions from solving a traveling salesman problem, and route execution in practice. Therefore, Amazon in a recent challenge supplied contestants a vast number of drivers' routes with the task of finding a good route prediction model. Using that data, the proposed methodology positions itself in the top-tier of submissions to this challenge, see Table 2. Besides its excellent performance confirming our modeling approach, the proposed approach is elegant and leads to several practical insights about delivery driving.

The approach first determines a zone sequence by solving a traveling salesman problem (TSP) over a cost matrix which is a weighted combination of historical information, which has to be learned from realized routes, and distance (or travel time) between zones. Next, on the local level, adhering to the established zone sequence on the global level, the order of stops to be visited in a zone is found by solving an open TSP (OTSP), which generates locally optimal solutions. Repeating this for all subsequent zones and patching them together generates a feasible route. Therefore, it can be considered to learn global (over zones) and optimize local (within zones).

Besides that the approach and solutions follow the principles of human navigation, it is elegant and convenient because it only requires learning on the zone level by tracking zone transitions. Additionally, the breakdown makes the approach fast, because the zone-sequence TSP globally and within-zone OTSPs locally are not computationally involved compared to solving a single TSP for an entire route, such as in the case of a clustered TSP (Chisman, 1975). This renders the approach amenable to real-time training using newly realized routes of the same day.

### 5.1. Practical insights

A key element in the approach is the computation of a cost matrix that combines distance and historical information when establishing the zone sequence as a TSP. The experiments show that a weighted

combination with travel times provides a slight performance improvement over using Euclidean distance. This fact underpins that when determining the zone sequence an implicit trade-off is made between the driver's zone-to-zone preferences and a measure of distance. Studying the weight value by changing it from 0 to 1, where setting it to 1 corresponds to only considering travel time, uncovers that the exact value is not critical. Overall, the best performance is obtained when the value is close to 0.9. Furthermore, the solutions stemming from our methodology are on average 5% shorter in time while they adhere to human navigational considerations as captured by the design of our approach.

The weighting of station-to-zone and zone-to-station transitions can be further adapted in the cost matrix to better mimic the driver's behavior resulting in another performance improvement. In fact, these adaptations are motivated by the logic that at the beginning of a route the driver is more concerned with where to start, whereas, at the end, when all stops are visited, the only consideration is the return to the station. Indeed, the experiments confirm these hypotheses and we find that the travel time matrix should be weighted less important in the station-to-zone transitions than in the zone-to-zone transitions, and from zone-to-station, it is the only factor.

Also, the test set provides us an opportunity to quantify the remaining learning gap. By first extracting the zone sequences in the test set followed by the application of our approach, we arrive at a score just below 0.01. This shows that the learn global and optimize local approach can yield a score as low as 0.009 (as a reference we get it to 0.0299, as shown in Section 4), surpassing the winners of the challenge. The learning took place on only 5112 routes, which results in a sparse transition matrix as the zone transition probabilities, when available, are learned based on the relatively few routes available for each region. Increasing the number of routes for learning, for example by daily tracking of realized routes and updating the count matrix accordingly, the approach can likely improve its accuracy on zone sequence prediction, which thus has a profound impact on the overall performance.

### 5.2. Future research

The approach as-is only utilizes historical stop data (with zone ids) in making predictions. Using additional data which are part of the case study, such as package size, start time, and quality of the route, did not provide any clear starting points for improvement. Since the approach heavily relies on the given zone ids were given, it is worthwhile to further study the choice of segmentation. In our case, we have seen that occurrences of zone revisitations are rare, showing that the segmentation is good. However, in 2.6% of the routes, there are more than five revisitations, which hints that for some zones in these routes, it might be valuable to reconsider the segmentation. In the case of frequent revisitations between zones, drivers might consider these zones as one, and therefore it might be beneficial to combine two or more zones in a *super-zone*; a concept introduced in Cook et al. (2022). Also, one could accommodate zone revisitations by keeping all stops in a full TSP formulation — instead of our method of breaking it into separate OTSPs — and adding penalties for possible zone-sequence violations, see Canoy et al. (2022). However, besides such an approach being computationally more intensive, setting the right penalties turns out to be a tedious task. Lastly, one can make the methodology more comprehensive by directly extracting zones from the route data or incorporating additional human navigational considerations in the design of routes, such as urban factors (e.g., parking availability and crowded streets).

Another endeavor is to study the cost matrix structure used for the problem on the global level. The methodology readily incorporates other functional forms as opposed to the current linear and interpretable combination. At the same time, one can also consider the modification of the weightings for the transitions, for example, to let them depend on the position in or the length of a route. This cannot be captured by keeping track of weights in a static matrix as currently being done. In such a case it requires a more dynamic treatment, as for each zone sequence instance the values can relate to different zone-to-zone transitions. Note that the transitions from and to the station are exceptions and can be captured in a static sense, because for any route it is always the start and end point, for details see Section 4.3.

Finally, the methodology can be deployed in other (last-mile) delivery concepts, see for example Boysen et al. (2021), or generalized to other operational optimization problems, which are susceptible to human interference. A logical starting point might be the allied Vehicle Routing Problem, e.g., Bräysy and Gendreau (2005a). In all, this research demonstrates that integrating learning in an optimization pipeline leverages its connection to practice.

## CRediT authorship contribution statement

**Mayukh Ghosh:** Conceptualization, Methodology, Software, Writing – original draft. **Alex Kuiper:** Methodology, Supervision, Writing – review & editing. **Roshan Mahes:** Conceptualization, Methodology, Software, Writing – original draft. **Donato Maragno:** Conceptualization, Methodology, Software, Writing – original draft.

## Data availability

The authors do not have permission to share data. However, more information about the data set can be found in Merchán et al. (2022).

## Acknowledgments

This work was supported by the Dutch Scientific Council (NWO) through grant OCENW.GROOT.2019.015, Optimization for and with Machine Learning (OPTIMAL), and grant 024.002.003, Gravitation project NETWORKS.

## Appendix. Pseudocode

In this Appendix, we provide the pseudocode to the methodology presented in Section 3. Although the pseudocode has been designed on the basis of the Amazon Last-Mile Routing Research Challenge dataset, it can be easily adapted to different data structures. The pseudocode used for the learning phase, i.e., the phase where we obtain a count matrix from the historical routes, is reported in Algorithm 1. The function `ToZoneSeq` maps the stop sequence (with zone ids) to a unique zone sequence. A full description of the method along with working examples of the function can be found in Section 3.1. These mappings are then saved in `ZoneSequences`. Besides, together with the stations, all unique zones we encounter are stored in the set `AllZones`. After applying this function to all routes in the dataset, the function `ComputeCountMatrix` is called to create an asymmetric matrix where the $(i, j)$-th entry represents the number of times a driver went from the $i$th to the $j$th zone of `AllZones`.

**Algorithm 1** Learning Zone Preferences and Preprocessing

**Input:** $D$: route data set
**Input:** $S$: set of stations
**Input:** `StopSeqs`: actual sequence of stops for each route in $D$
**Output:** `CountMatrix` $\in \mathbb{Z}_{+}^{(s+m)\times(s+m)}$: counts of zone transitions

```
ZoneSequences ← ∅
AllZones ← S
for route in D do
    ZoneSequences[route] ← ToZoneSeq(StopSeqs[route])
    AllZones ← AllZones ∪ {ZoneSequences[route]}
end for
CountMatrix ← ComputeCountMatrix(AllZones, ZoneSequences)
```

Algorithm 2 illustrates the pseudocode used in the prediction phase. Here, in order to compute the cost matrix, as thoroughly described in Section 3.2, we compute the distance matrix (with the function `DistanceMatrix`) where each entry refers to the travel time between two different zone centers (computed with `EstimateZoneCenter`). Finally, the function `ComputeCostMatrix` computes the weighted combination of the distance matrix ($D$ or $T$) and $1 - P$, which can be computed by normalizing each row of the count matrix ($N$) as described in Section 3.2. The code used to run the experiments presented in Section 4 is available upon request.

**Algorithm 2** Predicting a Route

**Input:** `CountMatrix`: output of Algorithm 1
**Input:** `station`, `Stops`, `Zones`: station and set of stops and zones in the route
**Input:** `TravelTimes`: travel times between each pair of stops in the route
**Input:** $\Omega$: weight matrix
**Output:** `PredictedStops`: predicted sequence of stops

```
ZoneCenters ← EstimateZoneCenter(Zones, Stops)
DistanceMatrix ← ComputeDistMatrix(Zones, ZoneCenters)
CostMatrix ← ComputeCostMatrix(Ω, DistanceMatrix, CountMatrix)
PredictedZoneSeq ← TSP(station, Zones, CostMatrix)
PredictedStopSeq ← [station]
for zone in PredictedZoneSeq do
    PrevStop ← PredictedStopSeq[-1]          ▷ last element
    AddStop ← ClosestToNextZoneCenter(zone, Stops)
    PredictedStopSeq ← PredictedStopSeq + [OTSP(PrevStop, Stops[zone], AddStop, TravelTimes)]
end for
```

## References


Amazon and MIT Center for Transportation & Logistics, 2021a. Amazon last-mile routing research challenge. URL https://routingchallenge.mit.edu/about-the-challenge/. (Accessed 30 April 2022).

Amazon and MIT Center for Transportation & Logistics, 2021b. Amazon last-mile routing research challenge. URL https://routingchallenge.mit.edu/last-mile-routing-challenge-team-performance-and-leaderboard/. (Accessed 30 April 2022).

Applegate, D.L., Bixby, R.E., Chvátal, V., Cook, W.J., 2011. The Traveling Salesman Problem. Princeton University Press.

Boyer, K.K., Prud'homme, A.M., Chung, W., 2009. The last mile challenge: Evaluating the effects of customer density and delivery window patterns. J. Bus. Logist. 30 (1), 185–201.

Boysen, N., Fedtke, S., Schwerdfeger, S., 2021. Last-mile delivery concepts: a survey from an operational research perspective. OR Spectrum 43 (1), 1–58.

Bräysy, O., Gendreau, M., 2005a. Vehicle routing problem with time windows, part I: Route construction and local search algorithms. Transp. Sci. 39 (1), 104–118.

Bräysy, O., Gendreau, M., 2005b. Vehicle routing problem with time windows, part II: Metaheuristics. Transp. Sci. 39 (1), 119–139.

Campuzano, G., Obreque, C., Aguayo, M.M., 2020. Accelerating the Miller-Tucker-Zemlin model for the asymmetric traveling salesman problem. Expert Syst. Appl. 148, 113229.

Canoy, R., Bucarey, V., Molenbruch, Y., Mulamba, M., Mandi, J., Guns, T., 2022. Probability estimation and structured output prediction for learning preferences in last mile delivery. arXiv preprint 2201.10269.

Canoy, R., Guns, T., 2019. Vehicle routing by learning from historical solutions. In: International Conference on Principles and Practice of Constraint Programming. Springer, pp. 54–70.

Ceikute, V., Jensen, C.S., 2013. Routing service quality–local driver behavior versus routing services. In: 2013 IEEE 14th International Conference on Mobile Data Management, Vol. 1. IEEE, pp. 97–106.

Cesari, G., 1996. Divide and conquer strategies for parallel TSP heuristics. Comput. Oper. Res. 23 (7), 681–694.

Chisman, J.A., 1975. The clustered traveling salesman problem. Comput. Oper. Res. 2 (2), 115–119. http://dx.doi.org/10.1016/0305-0548(75)90015-5.

Chopra, S., 2003. Designing the distribution network in a supply chain. Transp. Res. E 39 (2), 123–140.

Cook, W., Held, S., Helsgaun, K., 2022. Constrained local search for last-mile routing. Transp. Sci. http://dx.doi.org/10.1287/trsc.2022.1185.

Current, J.R., Schilling, D.A., 1989. The covering salesman problem. Transp. Sci. 23 (3), 208–213.

Dantzig, G.B., Ramser, J.H., 1959. The truck dispatching problem. Manage. Sci. 6 (1), 80–91.

Deloison, T., Hannon, E., Huber, A., Heid, B., Klink, C., Sahay, R., Wolff, C., 2020. The Future of the Last-Mile Ecosystem: Transition Roadmaps for Public-And Private-Sector Players. World Economic Forum.

Dry, M., Lee, M.D., Vickers, D., Hughes, P., 2006. Human performance on visually presented traveling salesperson problems with varying numbers of nodes. J. Probl. Solving 1 (1), 20–32.

Flood, M.M., 1956. The traveling-salesman problem. Oper. Res. 4 (1), 61–75.

Gantenbein, D., 2021. Winning Last Mile Challenge team addresses problem of combining mathematical routes with driver knowledge. URL https://www.amazon.science/academic-engagements/winning-last-mile-challenge-team-addresses-problem-of-combining-mathematical-routes-with-driver-knowledge. (Accessed 30 April 2022).

Gendreau, M., Hertz, A., Laporte, G., Stan, M., 1998. A generalized insertion heuristic for the traveling salesman problem with time windows. Oper. Res. 46 (3), 330–335.

Gendreau, M., Laporte, G., Vigo, D., 1999. Heuristics for the traveling salesman problem with pickup and delivery. Comput. Oper. Res. 26 (7), 699–714.

Gevaers, R., Van de Voorde, E., Vanelslander, T., 2011. Characteristics and typology of last-mile logistics from an innovation perspective in an urban context. In: City Distribution and Urban Freight Transport. Edward Elgar Publishing.

Gevaers, R., Van de Voorde, E., Vanelslander, T., 2014. Cost modelling and simulation of last-mile characteristics in an innovative B2C supply chain environment with implications on urban areas and cities. Procedia Soc. Behav. Sci. 125, 398–411, Eighth International Conference on City Logistics 17-19 June 2013, Bali, Indonesia.

Goodman, R.W., 2005. Whatever You Call It, Just Don't Think of Last-Mile Logistics, Last. Glob. Logist. Supply Chain Strateg., SupplyChainBrain, URL https://www.supplychainbrain.com/articles/601-whatever-you-call-it-just-dont-think-of-last-mile-logistics-last. (Accessed 30 April 2022).

Graham, S.M., Joshi, A., Pizlo, Z., 2000. The traveling salesman problem: A hierarchical model. Mem. Cogn. 28 (7), 1191–1204.

Gunasekaran, A., Patel, C., McGaughey, R.E., 2004. A framework for supply chain performance measurement. Int. J. Prod. Econ. 87 (3), 333–347.

Jiang, J., Gao, J., Li, G., Wu, C., Pei, Z., 2014. Hierarchical solving method for large scale TSP problems. In: International Symposium on Neural Networks. Springer, pp. 252–261.

Karp, R.M., 1977. Probabilistic analysis of partitioning algorithms for the traveling-salesman problem in the plane. Math. Oper. Res. 2 (3), 209–224.

Kool, W., van Hoof, H., Welling, M., 2019. Attention, learn to solve routing problems!. In: International Conference on Learning Representations. URL https://openreview.net/forum?id=ByxBFsRqYm.

Krumm, J., 2008. A Markov model for driver turn prediction. In: Society of Automotive Engineers (SAE) World Congress. April 2008.

Lawler, E.L., 1985. The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization. Wiley-Interscience Series in Discrete Mathematics.

Levenshtein, V.I., 1966. Binary codes capable of correcting deletions, insertions, and reversals. Sov. Phys. Doklady 10, 707–710.

Liao, E., Liu, C., 2018. A hierarchical algorithm based on density peaks clustering and ant colony optimization for traveling salesman problem. IEEE Access 6, 38921–38933.

Lin, C., Choy, K., Ho, G., Chung, S.H., Lam, H., 2014. Survey of green vehicle routing problem: Past and future trends. Expert Syst. Appl. 41, 1118–1138.

Lipsman, A., Liu, C., 2020. US ecommerce 2020, Coronavirus boosts ecommerce forecast and will accelerate channel-shift. eMarketer, URL https://www.emarketer.com/content/us-ecommerce-2020. (Accessed 30 April 2022).

MacGregor, J.N., Ormerod, T., 1996. Human performance on the traveling salesman problem. Percept. Psychophys. 58 (4), 527–539.

Merchán, D., Arora, J., Pachon, J., Konduri, K., Winkenbach, M., Parks, S., Noszek, J., 2022. 2021 Amazon last mile routing research challenge: Data set. Transp. Sci. http://dx.doi.org/10.1287/trsc.2022.1173.

Miller, C.E., Tucker, A.W., Zemlin, R.A., 1960. Integer programming formulation of traveling salesman problems. J. ACM 7 (4), 326–329.

Montoya-Torres, J.R., López Franco, J., Nieto Isaza, S., Felizzola Jiménez, H., Herazo-Padilla, N., 2015. A literature review on the vehicle routing problem with multiple depots. Comput. Ind. Eng. 79, 115–129.

OpenStreetMap, 2022. Planet dump retrieved from https://planet.osm.org. URL https://www.openstreetmap.org. (Accessed 30 April 2022).

Orman, A., Williams, H.P., 2007. A survey of different integer programming formulations of the travelling salesman problem. In: Advances in Computational Management Science, Vol. 9. Springer Berlin Heidelberg, pp. 91–104.

Papadimitriou, C.H., 1977. The Euclidean traveling salesman problem is NP-complete. Theoret. Comput. Sci. 4 (3), 237–244.

Pizlo, Z., Stefanov, E., Saalweachter, J., Li, Z., Haxhimusa, Y., Kropatsch, W.G., 2006. Traveling salesman problem: A foveating pyramid model. J. Probl. Solving 1 (1), 83–101.

Purkayastha, R., Chakraborty, T., Saha, A., Mukhopadhyay, D., 2020. Study and analysis of various heuristic algorithms for solving travelling salesman problem–A survey. In: Mandal, J.K., Mukhopadhyay, S. (Eds.), Proceedings of the Global AI Congress 2019. Springer Singapore, pp. 61–70.

Sengupta, L., Mariescu-Istodor, R., Fränti, P., 2018. Planning your route: Where to start? Comput. Brain Behav. 1 (3–4), 252–265.

Speranza, M., Archetti, C., 2014. A survey on matheuristics for routing problems. EURO J. Comput. Optim. 2, 223–246.

Taniguchi, E., Thompson, R.G., 2002. Modeling city logistics. Transp. Res. Rec. 1790 (1), 45–51.

Toledo, T., Sun, Y., Rosa, K., Ben-Akiva, M., Flanagan, K., Sanchez, R., Spissu, E., 2013. Decision-making process and factors affecting truck routing. In: Freight Transport Modelling. Emerald Group Publishing Limited, pp. 233–249.

United Nations, 2018. 68% of the world population projected to live in urban areas by 2050, says UN. United Nations Department of Economic and Social Affairs 2018, URL https://www.un.org/development/desa/en/news/population/2018-revision-of-world-urbanization-prospects.html. (Accessed 30 April 2022).

Van Rooij, I., Stege, U., Schactman, A., 2003. Convex hull and tour crossings in the Euclidean traveling salesperson problem: Implications for human performance studies. Mem. Cogn. 31 (2), 215–220.

Vickers, D., Lee, M.D., Dry, M., Hughes, P., 2003. The roles of the convex hull and the number of potential intersections in performance on visually presented traveling salesperson problems. Mem. Cogn. 31 (7), 1094–1104.

Vidal, T., Crainic, T.G., Gendreau, M., Prins, C., 2013. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. Comput. Oper. Res. 40 (1), 475–489.

Wang, X., Ma, Y., Di, J., Murphey, Y.L., Qiu, S., Kristinsson, J., Meyer, J., Tseng, F., Feldkamp, T., 2015. Building efficient probability transition matrix using machine learning from big data for personalized route prediction. Procedia Comput. Sci. 53, 284–291.

Wiener, J., Ehbauer, N., Mallot, H., 2009. Planning paths to multiple targets: Memory involvement and planning heuristics in spatial problem solving. Psychol. Res. PRPF 73 (5), 644–658.

Wiener, J.M., Mallot, H.A., 2003. 'Fine-to-coarse' route planning and navigation in regionalized environments. Spatial Cogn. Comput. 3 (4), 331–358.

Ye, N., Wang, Z.-q., Malekian, R., Lin, Q., Wang, R.-c., 2015. A method for driving route predictions based on hidden Markov model. Math. Probl. Eng. 2015.