GEOAWARE - A SIMULATION-BASED FRAMEWORK FOR SYNTHETIC TRAJECTORY GENERATION FROM MOBILITY PATTERNS

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Computer Engineering

by

JAMESON D. MORGAN
B.S.C.E., Wright State University University, 2018

2020
Wright State University

Wright State University
Graduate School

December 3, 2020

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY Jameson D. Morgan ENTITLED GeoAware - A Simulation-based Framework for Synthetic Trajectory Generation from Mobility Patterns BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Master of Science in Computer Engineering.

Derek Doran, Ph.D.
Thesis Director

Mateen Rizki, Ph.D.
Chair, Department of Computer Science and Engineering

Committee on
Final Examination

Advisor, Derek Doran, Ph.D.

Adam Nolan, Ph.D.

Michael Raymer, Ph.D.

Barry Milligan, Ph.D., Interim Dean of the Graduate School

ABSTRACT

Morgan, Jameson D. M.S.C.E., Department of Computer Science and Engineering, Wright State University, 2020. *GeoAware - A Simulation-based Framework for Synthetic Trajectory Generation from Mobility Patterns.*

Recent advances in location acquisition services have resulted in vast amounts of trajectory data; providing valuable insight into human mobility. The field of trajectory data mining has exploded as a result, with literature detailing algorithms for (pre)processing, map matching, pattern mining, and the like. Unfortunately, obtaining trajectory data for the design and evaluation of such algorithms is problematic due to privacy, ethical, dataset size, researcher access, and sampling frequency concerns. Synthetic trajectories provide a solution to such a problem as they are cheap to produce and are derived from a fully controllable generation procedure. Citing deficiencies in modern synthetic trajectory procedures, we propose a data-driven, seasonally-aware and simulation-based procedure that incorporates macro- and micro-level patterns from reference trajectories. The procedure is implemented as an alpha-release package; allowing an analyst to produce synthetic trajectories via the use of a modular coding framework and analysis tools.

Contents

List of Figures

List of Tables

Acknowledgment

As a task-driven individual, the completion of a task is great motivation for me. It is not that I don't enjoy the journey along the way but once I have planned and begun the task at hand, nothing excites me more than to complete that task. But as I contemplate my collegiate career, I realize that if I were to have merely followed my task-oriented mindset, I would have missed the support and guidance of many individuals who have helped me become the individual I am today.

First, I would like to thank my advisor, Dr. Doran. As I reflect, it seems like just yesterday I arrived in your office out of breath and talking ninety miles-a-minute. You (Ning was also there) were extremely encouraging and thus started my 3+ years with you as my advisor. I know I put up quite a fight to do things my way, but I am extremely grateful for the academic and professional guidance you have supplied throughout the years. Despite all the consternation I am sure I caused at times, I hope I always met your expectations.

Second, I want to express my tremendous thanks for the support, both financially and emotionally, my parents have provided throughout my collegiate career and entire life. Words fail to describe the impact you both have had. I know for certain that my mother's dedication to homeschooling me has had a tremendous impact on my success in college and I must admit that my mom told me I would pursue a graduate degree from the beginning. Additionally, my brothers and sisters — Brandon, Nicholas, Connor, Annissa and Caroline — have provided much needed mental breaks and fellowship. Thank you!

Third, I would like to thank everyone (Amanda, Jace, Kyle, Mahdieh, Matthew B., Matthew P., Ning, and Walter) I worked with in the MLaCS (formerly WaCS) lab. Your kind companionship will always be remembered. I would like to particularly thank Matthew Piekenbrock and Jace Robinson for exposing me to the world of `R` programming.

Last, but certainly not least, I want to acknowledge the support of my Lord and Savior, Jesus Christ, without whom no task, big or small, is possible. For it says in Proverbs 16:9, *"The mind of man plans his way, but the LORD directs his steps" (NASB).*

Dedicated to

my Lord and Savior, Jesus Christ

who has made my life

"worth the living just because He lives."

—

For God so loved the world, that He gave His only begotten Son, that whoever believes in Him shall not perish, but have eternal life.

John 3:16 (NASB)

Chapter 1 — Introduction

Understanding human mobility is vital for a variety of undertakings in urban planning. Models of human mobility have been applied in traffic planning contexts [206] [54] [19] [166] [24], evacuation planning [233] [110] [138], and even for privacy protection [94] [126]. As with any model, the inner-workings of human mobility models must be tested and their ability to model reality demonstrated [171]. Often, such verification and validation relies on external data.

As location acquisition services and mobile technology continue to improve and become more widespread, vast amounts of trace-level data detailing the path followed by an individual agent are being produced [143]. Such *trajectories*, as they are called, are of immense value when trying to understand mobility and facilitate fine-grained analysis that was not possible using the historically prevalent aggregate data sources (e.g. roadway counts) [237] [119]. This ever growing collection of trajectories has sparked new mobility models specifically designed to take advantage of the low-level information provided by such a data source [237]. As such, the field of *trajectory data mining* [237] has exploded with literature detailing trajectory (pre)processing [51], map matching [231], access and storage [49], pattern mining [83] and the like.

Ideally, because trace-level mobility algorithms are used in the real world, the verification and validation of such algorithms would be performed with real world trajectories [64]. Obtaining such data, however, is problematic. Although recent improvements in technology and penetration have substantially reduced the cost of obtaining real trajectory

datasets [50], other considerations, such as user privacy, ethical concerns, dataset size, researcher access, and sampling frequency still make acquiring a suitably large and detailed dataset of trajectory data challenging [171] [143] [119] [174]. While the collection of such information is often passively performed by many companies for internal purposes (such as service delivery, quality of service or analytics), any publicly available data is typically aggregated to protect user privacy[1]. Moreover, even if a dataset of suitable detail and scale could be found, various aspects of the dataset, such as the roadway network and/or the demand generation process, would be beyond the control of the analyst, thereby limiting the analysis that could be performed.

Such a situation makes it difficult to test new (or old) models of human mobility which rely on low-level trajectories. In essence, a dichotomy exists between the suitability of trajectory data for understanding human mobility and the availability of such data for verifying and validating models. An answer to this dilemma is the generation of *synthetic* trajectories (synthetics).

1.1 Synthetic Trajectories

Synthetic trajectories are valuable for a number of reasons. First, the production of synthetic trajectories is controllable. In real-world scenarios, it is impossible to have access to the underlying process generating the observed trajectories but in the production of synthetic trajectories, the analyst has full control over the generation process. Quantities such as the level of detail, the vehicle classes involved, the roadway network, the output produced and even the situations induced (e.g. traffic accidents, anomalies or congestion) are fully controllable based upon the application and desires of the analyst. Because of this, synthetic data or examples are quite common in transportation modeling literature during the discussion of results [119] [64], as a comparison point [80] [57] [75] [174] [40] or for

[1] See https://movement.uber.com for instance.

numerical analysis [132] [179].

Another motivating aspect of synthetic trajectories is that data generation is cheap and trajectories can be produced for any geographic region at any scale and level of detail. As such, access, use restrictions, and privacy concerns become irrelevant, enabling greater flexibility in usage.

Synthetic trajectories also facilitate reproducible experimentation. As noted in [121], scientific computing — the use of a computer to perform scientific experimentation — has become pervasive in the scientific community but unfortunately how to perform proper research and experimentation has not. In their book, [121] argue that reproducibility is critical to effective research and experimentation and the greater goal of open science. Of course, research on both real-world and synthetic trajectories can benefit from proper methods but synthetic trajectories provide an unrivaled testbed for research, experimentation, and analysis. For instance, if pseudo-random number generators (RNGs) with known seed values are used when generating synthetic trajectories, probabilistic procedures can be recreated so that experiments can be compared without wondering whether the variation between the experiments is attributable to probabilistic differences.

Before leaving this section, it is important to note that despite the advantages of synthetics, real world trajectories are not without merit. It must be understood that while useful, synthetic trajectories are merely a model of the real-world. Because various assumptions must be made during the modeling process, the output is necessarily a "reflection" of reality and will never replicate the real world exactly. On the other hand, real-world trajectories capture a myriad of subtleties, such as noise, agent-to-agent interactions or network-demand interactions, which synthetic trajectories are unable to fully replicate. A second advantage of real-world data is that various situations are captured without thought by the analyst. In contrast, when generating synthetic trajectories the analyst must consider what types of situations are likely to occur and make sure the produced synthetics contain such events. Indeed, the overlooking of situations is a major challenge which must

be mitigated when using synthetic trajectories.

1.2 Synthetics Production via Traffic Simulator

One approach to creating such synthetics is to use a traffic simulator. Simulation-based techniques to transportation modeling, despite their mathematical intractability (due to the inherently ill-behaved nature of traffic) [169], have enjoyed much success because of their ability to capture complex interactions, particularly that of congestion, which cannot be effectively defined mathematically [195] [191] [115] [169] [70] [20] [238]. *Microscopic* traffic simulators are a specific type of traffic simulator designed to operate at the individual agent level [20] and have become particularly prevalent in recent years due to increases in computer hardware speed [76]. Yet, despite the success of microscopic models in simulating roadway conditions (see examples such as [165] [229] [236]), a largely neglected topic is the use and appropriateness of microscopic traffic simulators for generating synthetic trajectories. This omission is likely due to the fact that in most microscopic traffic simulators the trajectories produced are merely treated as intermediaries whose purpose is to induce various macro-level demand-supply interactions (such as link counts or queuing) [64] [15]. In other words, the low-level routing of agents is assumed to only be of use for bringing about certain phenomenon which an analyst wishes to evaluate. Because the production of trace-level output is not the main purpose of microscopic simulators, a procedure must exist or be created in order to efficiently extract such information from the simulation.

Utilizing microscopic traffic simulators for synthetics production also requires that the simulation be calibrated if the synthetics are to be used in real-world situations. Merely running a microscopic simulation and collecting the traces produced will result in traces that are reflective of simulation parameter defaults not real-world conditions. Despite the prevalence of calibration techniques for microscopic traffic simulations in the literature (see [101], [210], [16], or [120] for typical approaches and [23], [235], [211] for case studies),

such procedures often (i) utilize aggregate [101] or behavioral [206] data sources for calibration, (ii) focus on recreating a set of observed conditions (termed base-year conditions) [172], and (iii) are often applied as a post-fact corrective measure to fix simulation output. Each of these points presents issues when trying to obtain accurate synthetics from a microscopic simulation.

First, by utilizing aggregate data, the low-level aspects of a simulation cannot be calibrated against real-world data. When microscopic traffic simulators are used for planning purposes at a macroscopic scale, it may be possible to ignore such inconsistencies but when one wishes to utilize a microscopic traffic simulator for producing synthetics, such low-level accuracy is required. Second, as [172] notes, focusing only on replicating a set of conditions makes the simulation applicable only in such conditions. Indeed, as they argue, any simulation can be made to conform to a set of conditions but this does not mean it captures the average or general conditions present [172]. If the variables used to calibrate the simulation are not transferable to other situations, neither will the resulting simulation. Third, often calibration, particularly that of demand calibration, is performed after the initial demand has been generated and its only purpose is to try and correct the generated demand to fit with observed values. While such processes are needed, its need arises from the fact that the underlying demand models are not generating demand in line with real-world observations.

1.3 The Proposed Framework

Motivated by the appropriateness of microscopic traffic simulators for producing realistic synthetics that mirror real life, we present a framework for producing such synthetics via a microscopic traffic simulation calibrated from transferable seasonal demand patterns extracted from reference vehicle traces. As presented earlier, trajectories provide rich information about mobility patterns and therefore, we seek to push the state of practice in

synthetics generation by utilizing such trajectories to calibrate the demand model of a microscopic simulation so that the synthetics produced reflect real-world demand patterns. The framework is called GeoAware as it presents a geographically-aware framework for producing synthetic trajectories at a desired scale and detail.

The framework's demand model is calibrated from the reference traces using a data-driven process which incorporates the seasonality present in mobility data [199] [142] [87] [184] [157]. The demand model captures mobility at both macro- and micro-level scales. Macro-level demand between regions in the network is captured using a seasonally dynamic stochastic block model (SDSBM). Micro-level demand patterns capture the seasonal patterns of departure times and incident weights. Path-level demand patterns[2] are replicated through the incorporation of a map-matching algorithm which allows the simulation to reproduce routes exhibited in the reference trajectories.

A motivating example for such a framework can be found in the taxicab dataset made publicly available by the city of Chicago[3] [156]. For privacy reasons, this dataset only provides the pickup and dropoff regions of agents, excluding any of the route details. Thus, if one where interested in studying mobility patterns in the Chicago area, such a dataset would necessarily limit analysis to high-level pickup and dropoff statistics. However, if one where able to infer mobility pattern statistics from this more limited data source, a calibrated microscopic simulation could be used to generate synthetic trajectories whose behavior is based upon such pickup and dropoff trends but at a scale and detail not available in the original dataset [119].

We implement this framework in an alpha-release package[4] aimed at enabling fully reproducible, data-driven vehicle simulations. Using this package, an analyst is able to go from ingesting *reference* trajectories to generating *synthetic* trajectory data via the use of a modular coding framework and analysis tools. The package components are designed

[2]As will be presented subsequently, path-level demand pattern inference requires that the trajectories provided are full vehicle traces.

[3]The latest collection of records can accessed at: https://data.cityofchicago.org/Transportation/Taxi-Trips/wrvz-psew

[4]Due to sponsor requirements, the source code cannot be made publicly available.

to be deployable and extensible. Due to the author's strong belief in reproducible and self-documenting research, the high-level framework modules are implemented in the `R` programming language [178] in order to take advantage of the data analysis, documentation and reproducibility tools available in `R`. Low-level, memory and computation intensive modules are written in `C++` and are exposed to the `R` programming environment. The package modules are designed to interconnect, allowing an analyst to define vehicle simulations rapidly and efficiently. The package interfaces with SUMO (Simulation of Urban MObility) [140], an open-source package for microscopic road simulation, allowing us to take advantage of the rich feature set provided by SUMO when performing the microscopic roadway simulations necessary for generating synthetic mobility-records.

1.4 Contributions

The goal of the GeoAware framework is to use a demand-calibrated microscopic traffic simulator to produce synthetics that (i) behave realistically, (ii) can be generated at any scale, and (iii) are not subject to privacy concerns. Such a goal took inspiration from a prior research effort that inferred geospatial mobility patterns and anomalies from trajectory data [153] and results in contributions to the current state-of-practice in synthetics generation in the following areas:

(i) The development of a data-driven, seasonally-aware, and simulation-based synthetics generation procedure incorporating transferable macro- and micro-level trend information;

(ii) The use of trajectory data for generator calibration; and

(iii) The incorporation of this generation procedure into an extensible, application focused, alpha-release software package for generating fully reproducible vehicle simulations within the R programming environment.

The focus of this work is on presenting a solution to a common research problem and is therefore application focused. As such, we do not seek to show that the proposed framework is suited for generating synthetics in a specific scenario but instead, aim to convey the general principles of the framework which may be used by an analyst to generate synthetics in a variety of contexts. Rather than establishing new theory, we focus on the integration of (mostly) pre-established tools and algorithms into a software pipeline capable of achieving the aforementioned goal. It is our hope that the resulting framework will be of immediate utility and pushes the state of practice.

1.5 Organization

We now detail the organization of this work. We begin with a literature review in chapter 2. Here, we focus on approaches to generating synthetic vehicle traces and motivate the modern approach presented in the current work. Next, the methodology chapter (chapter 3) defines the GeoAware framework. The chapter begins by detailing the framework data requirements in section 3.1, proceeds to describe the demand model in section 3.2, outlines the creation of a simulation scenario in section 3.2.6, and concludes by describing the simulation process in section 3.3. Following the methodology chapter, the framework is evaluated in chapter 4. We evaluate the framework both qualitatively (section 4.1) and quantitatively (section 4.2). Finally, we present our concluding remarks, discuss areas of improvement and hint at future work in chapter 5.

This work is primarily an applied project, focused on incorporating available tools and techniques in order to solve a real-world problem. At times, this incorporation requires some preliminary knowledge about a tool or technique. Rather than present all of these preliminaries up front in a separate section, we have opted to introduce any required preliminaries "just-in-time". We believe that introducing the applicable content when required aids reader understanding and focuses our writing on explaining and justifying the use of

such preliminaries in our project.

Finally, as an applied research work, an emphasis has been placed on reproducibility. To assist in this endeavor, the methodology and evaluation chapters (chapters 3 and 4, respectively) are presented in a vignette-style, providing a step-wise description of the framework which is easy to understand and evaluate. As our purpose is to present a general framework, we refrain from specifying an exact scenario during these discussions. Nevertheless, to demonstrate the overall utility, we do present an example in the concluding section of the evaluation chapter.

Chapter 2 — Related Works

The goal of synthetic trajectory generation is to create a set of artificial trajectories from a collection of *a priori* and/or empirically derived distributions as real world trajectories are often unavailable or of limited size due to factors such as user privacy and ethics [119], collection or storage limitations [64] [119], and/or researcher access. A major application of synthetic trajectories is in the design and evaluation of spatiotemporal algorithms [171] [237] [49] [106] [6]. Synthetic trajectories are also valuable for privacy protection [94] [126] and testing the performance of spatial database operations [171] [64] [40] [56] [159].

A review of the related literature indicates that there are three commonly used approaches for constructing human related, synthetic trajectories, namely (i) moving object generators, (ii) human mobility models, and (iii) machine learning techniques. We review the literature pertaining to each approach in the subsequent sections. We conclude this chapter by comparing and contrasting the reviewed works with the proposed framework. It is our intention that such a discussion will motivate the proposed work.

2.1 Moving Object Generators

A spatiotemporal moving object generator (MOG) is a configurable algorithm for producing high resolution spatiotemporal traces of moving objects. The earliest spatiotemporal MOG is [208]. Its construction was motivated by the need to evaluate the performance of spatiotemporal database systems that were, at that time, being developed [40] [208]. At

that time, what was needed was an algorithm that could produce artificial traces that incorporated spatiotemporal properties. Despite the original purpose of MOGs, the ability to construct trace-level data given a set of configurable parameters makes them useful for tasks beyond performance evaluation; such as the evaluation of spatiotemporal algorithms and data structures [171].

The related literature concerning moving object generators is not vast and can be grouped into two categories[1]. The first category generates trajectories in unconstrained *free space*. The second, generates trajectories that are *network constrained*. We begin by reviewing the free space approaches.

2.1.1 Generation in Free Space

As noted earlier, the earliest MOG was detailed in [208]. The generator was presented alongside design and evaluation considerations for spatiotemporal databases and is called GSTD (Generate Spatial Temporal Data). The framework operates in free space and generates timestamped location tuples. The framework supports point and rectangular data. The algorithm allows users to specify distributions (from a set of supported distributions) controlling the jump between subsequent timestamps (called the *duration*), the shift of an object and the resizing of an object (non-point data only). A later extension [173] enables directed movement by adding an additional parameter and supports the creation of rectangular infrastructure objects to simulate movement in obstructed free space.

A central issue with the original GSTD algorithm presented above is that the movement is not necessarily as smooth as might be expected in the real world. While [173] later addresses this through the introduction of a direction interval (holds the direction for a certain period of time), [187] also addresses this issue in an earlier work. Their work is inspired by a fishing scenario where trawlers go out in search of fish while avoiding stormy

[1]Moving object generators for indoor environments also exist but are of limited relevance to the current work as vehicle traces are external events. See [104] and [135] for two approaches.

areas. This work uses object avoidance as ships use a repulsion vector to avoid stormy areas and an attraction vector to navigate towards shoals of fish. While motivated by a very specific scenario, the general principles can be used to create moving objects for any scenario where attractive and repulsive forces are present.

While the GSTD and Oporto algorithms allow for the generation of moving objects which change their size, the shapes are limited to rectangles. While it is true that complex shapes are often represented by bounding rectangles [187], at times it is necessary to fully represent the complex shape. Examples include tracking storms or the evaluation of pattern mining algorithms [13]. [215] presents the G-TERD (generator of time-evolving regional data) framework which supports arbitrary 2D shapes that change their spatial location, size and/or shape according to speed, zoom and rotation angle parameters. The framework also hints at object interaction, something that will become more readily implemented in network-constrained MOGs, by allowing regions to avoid each other and pass over others (e.g. a cloud passing over an island). The framework produces a sequence of multicolored images.

The ERMO-DG framework also generates moving object regions of arbitrary size but does so with a focus on incorporating spatiotemporal patterns so that the produced trajectories can be used in evaluating spatiotemporal pattern mining algorithms [13]. The framework operates by constructing randomly generated (but configurable) feature types that control the shape, area, lifetime, velocity and acceleration attributes of the generated objects. Next, a collection of core and overlap patterns are produced by randomly combining feature types. Those patterns are assigned to spatial neighborhoods which will generate instances that follow the patterns specified. The framework is highly parameterized, allowing an analyst to construct datasets that meet certain pattern criteria.

Moving object generators have also been proposed for creating call detail record (CDR) data. The CENTRE (CEllular Network Trajectory Reconstruction Environment) framework extends the GSTD algorithm and features many notable improvements [82].

First, through the use of a GIS program, obstacles of arbitrary shape can be constructed. Second, group behavior, where objects share similar attributes, can be achieved in a single simulation run and objects can shift between groups. The synthetic CDR data can be transformed into trace-level trajectories through a naïve trajectory reconstruction algorithm. The WHERE framework provides another approach to generating synthetic CDR data but unlike CENTRE, WHERE incorporates empirically derived distributions controlling home and work locations, commute distance, spatial call patterns and temporal call patterns [108]. While the framework does not claim to be a MOG, the synthetically produced CDRs implicitly provide a description of movement, even if it is at a course level.

The GAMMA (Generating Artificial Modeless Movement by genetic-Algorithm) framework presents the earliest attempt at tuning a MOG from field data[2] [102]. The framework views trajectory generation as an optimization problem and uses the genetic algorithm to select valid trajectories according to a fitness score while permuting others to obtain new (hopefully more representative) solutions. The generator requires a collection of reference trajectories from which to infer the fitness score and thereby incorporate real-world behavior. Applications are presented for generating CDRs and semantic trajectories.

2.1.2 Network Constrained Generation

In contrast to the previous set of MOGs which generate objects in an unconstrained (or minimally constrained) environment, network constrained MOGs generate objects which are confined to a network topology. Obviously, in some situations such a constraint makes sense. For instance, motorists must travel from their onset location to their destination according to the roadway network available to them.

The work presented in [40] is widely cited as the first attempt to constrain the movement of generated objects in a realistic way (even though SUMO [140] technically appeared

[2]The SUMO framework presented later [140] does feature some native calibration tools for configuring demand but such tools are not requirements in the simulation pipeline and therefore a default SUMO configuration is unlikely to produce accurate synthetics.

earlier) . Motivated by the field of traffic telemetrics, the author details a generator capable of producing network constrained objects on any arbitrary network. The framework selects an onset and terminus node (through user controlled procedures) and routes an agent along the network according to fastest path principles. Interaction between other objects in the network is limited to the induced characteristics on edge speed and the resulting route chosen. The framework is often referred to as *Brinkhoff* after the author's name.

The BerlinMOD generator is similar in spirit to the generator presented in [40] but focuses on the production of long-term observations (e.g. a month) [64]. The framework focuses primarily on trips between home and work with additional trips being modeled in the evenings or weekends. The trips are generated in a heuristic fashion and objects do not interact with each other on the roadway (i.e. it is as if they are the only object on the network). The framework features a number of native data export options and the default map of Berlin that is used for generation can be substituted for another location.

Microscopic traffic simulators may also be used to generate network constrained moving objects if the simulator supports the exporting of vehicle positions. Microscopic traffic simulators increase the fidelity of the generated output by incorporating car following, lane-changing and gap acceptance theories; ensuring that interactions between objects are modeled according to established traffic theory [70]. A prominent example of a traffic simulator supporting the generation of moving object data is the Simulation of Urban MObility (SUMO) framework [140]. SUMO is highly configurable and allows an analyst to specify, amongst other quantities: traffic control structures (e.g. stop signs or traffic lights), lane-level details, vehicle types, traffic assignment approaches and intermodal routing. (Indeed, SUMO's rich feature set provided strong motivation for its incorporation into the present work.) The enhanced realism of microscopic traffic simulators does come at a cost and therefore, distributed architectures have also been proposed [228] [234].

Intermodal routing is not a common feature in MOGs, prompting the creation of MW-Gen (mini world generator), a MOG specifically designed to handle intermodal routes

[230]. The simulation space is constructed from five infrastructure representations that detail the roadway network, outdoor environment, bus network, metro network and indoor floor plans. A global space is constructed on top of the infrastructure to facilitate infrastructure interaction. Synthetic trips between any two arbitrary points are constructed using intermodal routes, if necessary.

A drawback in many of the reviewed MOGs thus far is that, with few exceptions, no formal method is given to calibrate the various framework parameters so that the produced trajectories reflect reality. Indeed, merely moving objects along network edges only partially increases the reality of the trajectories produced. To obtain more representative parameter settings, information must be inferred from representative data. A handful of literature pieces address this concern and present various solutions to the problem.

The ST-ACTS (Spatio-Temporal ACTivity Simulator) framework uses "various geostatistical data sources and intuitive principles" when constructing synthetics; features, the authors claim, were neglected in earlier MOGs [84]. The simulator relies on real-world data sources detailing (i) demographics, (ii) business/facility information, and (iii) consumer surveys. Synthetics are constructed by creating a synthetic population according to the distributions present in the real-world data and then performing a discrete event simulation using the generated population. The trajectories produced are symbolic. A potential drawback of this approach is the large amount of data required.

Recent examples of MOGs continue to incorporate field data when producing synthetics. Hermoupolis produces annotated, network constrained, pattern incorporating trajectories by using generalized mobility patterns (GMPs) extracted from reference trajectories [170]. [200] presents a SUMO-based framework which generates traffic demand using L2 logistic regression and routes the resulting demand through SUMO.

2.2 Human Mobility Models

Moving object generators are not the only technique available for obtaining synthetic trajectories. As research continues to explore and demonstrate the predictability of human mobility [199] [60], increasingly better models of human mobility are being proposed which can be used to generate synthetics at the scale of the individual[3]. In contrast to moving object generators which are typically based on simplified models of mobility (notable exceptions being SUMO [140], Hermoupolis [170] and GAMMA [102]), human mobility models are concerned with accurately capturing a characteristic (or characteristics) of human mobility. The ability to generate trajectories from such a model is merely a side effect.

2.2.1 Random Walk Models

Random walk models are an easy and often used, albeit not particularly accurate, approach for modeling human mobility [86]. In a random walk model, the position (any arbitrary number of dimensions) of an agent after N steps is given by the random variable[4] $\mathbf{X}_N = \sum_{i=1}^{N} \Delta\mathbf{X}_i$, where the displacement $\Delta\mathbf{X}_i$ is a random variable extracted from the distribution $f(\Delta\mathbf{x})$ with statistically independent draws. Each draw is referred to as a *jump* as it represents a movement of the agent to a new position.

There are three techniques derived from the random walk model which are particularly common in human mobility modeling [17]. The first is Brownian motion[5] (also used for modeling particles suspended in fluids [221]) which stipulates that the displacement $\Delta\mathbf{X}_i$ be drawn from a Gaussian distribution with a mean of zero and a variance proportional to time t. The second commonly utilized technique is known as the Lévy flight[6]

[3] While models of human mobility exist at both the individual- and population-level, we restrict our literature review to those of individual movement since they are capable of producing trace-level trajectories and are therefore most relevant to the proposed work. For an excellent and current review of human mobility at both the individual and population level, the reader is encouraged to consult [17].

[4] We borrow the notation of [17] for all equations in this section.

[5] Named after the botanist Robert Brown who first described it.

[6] Named after the French mathematician Paul Lévy.

model and can be constructed by requiring that the displacement $\Delta\mathbf{X}_i$ follow a heavy-tailed distribution. The last random walk derivative commonly used for modeling human mobility is the continuous time random walk (CTRW) model[7] which extends the basic random walk by also modeling the time between jumps according to the random variable $\mathbf{T}_N = \sum_{i=1}^{N} \Delta\mathbf{T}_i$. [41] argues that human mobility is characterized as a CTRW Lévy flight model with heavy-tailed jump and time distributions but evidence suggests that such results are not representative in general [86].

Instead, [86] argues that human mobility exhibits strong spatial and temporal regularity as individuals regularly frequent a few select locations. [198] proposes a solution to account for this trait by introducing two extensions to the traditional CTRW model. The first is termed *exploration* as it models the probability that an agent will visit a new location and is given by $P_{new} = \rho S^{-\gamma}$, where S is the number of previously visited locations. The second is termed *preferential return* as it models the probability that an agent will return to a previously visited location and is given by the complimentary probability $P_{ret} = 1 - P_{new}$. When returning, a location is chosen with a probability proportional to the number of visits $\Pi_i = f_i$ and the model parameters ρ and σ are empirically derived from data. Collectively, the exploration and preferential return steps are collectively referred to as the EPR model.

Despite the improvement of the EPR model, it is not without its faults. As noted in [18], by only considering the frequency of visitation when returning to a location, the preferential return extension will cause earlier visited locations to receive more visits and prevents an individual from changing his/her preferences. To address this issue, the authors present a model where with probability $1 - \alpha$ an individual chooses to return to previously visited location and each location's selection probability is proportional to its recency rank K_S.

Social interactions, such as family and friends, have also been shown to influence mobility patterns. As [17] points out in its review of state-of-the-art mobility models (see

[7] Introduced in [152].

section 4.1), the social network of an individual reflects the geography of their life. One example further extends the EPR model for short-term social contexts by making a proportion of the locations selected in the exploration and return stages dependent upon locations visited by a similar social contact [214].

2.2.2 Markov-based Models

A commonly held assumption in many of the models discussed thus far is that human mobility is Markovian [127]. This Markov assumption implies that agents are memoryless and therefore the next state of an agent is only based upon its current state (or a limited number of previous states k). Of course, such an assumption is not entirely true [127] [73] but Markov processes do have their place and are often advantageous due to their low computational complexity [127], their inherently generative nature (generating a trajectory for each step conditioned on the previous k states [94]), and the ability to define transition probabilities between locations based on past trajectories [77] [73].

The Markov property is commonly applied to the task of location prediction [142] [221] [11] [78] [180] [94] [162] [136] and such algorithms can be used to sequentially construct trajectories (with known issues; see [56]) by taking advantage of the inherently generative nature of Markov processes [56]. Probably the earliest work to use a Markovian model for location prediction is [136] which presents a Gauss-Markov model for predicting the future location of a mobile device in a cellular network. As with any location-based model, the resolution of the generated trajectories when using such models is dependent upon the level of discretization used in the spatial domain [116].

Because Markov-based models are naturally generative, some works do not detail a formal generation procedure for producing synthetics. Others, however, explicitly incorporate the mobility model into a process for generating synthetic trajectories. For instance, [94] presents a framework capable of generating *trace-level* synthetic trajectories with ϵ-differential privacy. The notable contributions of this work include the use of a hierarchical

reference system (HRS) to decompose the spatial domain so as to capture variances in agent speed and the inclusion of differential privacy.

[111] defines a generative process for constructing *location-level* trajectories between three location types (home, work and other) as inferred from suitable data (GPS or CDRs). The workplace is assumed to have fixed location, start time and duration. The remaining temporal aspects are modeled at the individual level by a time-inhomogeneous Markov chain with three parameters capturing (i) the weekly home-based tour number (to other locations), (ii) the dwell rate, and (iii) the burst rate. Spatial choices, excluding workplace selection, are modeled using a rank-based exploration and preferential return (r-EPR) model. [162] proposes a framework which similarly separates the temporal and spatial aspects for modeling purposes. The process begins by learning time dependent trip diaries which specify abstract locations (e.g. home, workplace, shopping mall). Then, the d-EPR algorithm [164] [161] is used to to assign physical locations to the abstract ones in a data-driven manner. Once again, the trajectories produced are *location-level*, not *trace-level*.

2.3 Machine Learning Techniques

One drawback with the approaches presented thus far is that each assumes a model of human mobility can be defined and the required parameters estimated [74]. Given the complexity of human mobility [74] [128], defining a model that strikes the right balance between expressiveness and simplicity can be challenging. Motivated by the complexity of such a situation, researchers began applying *parameter-free* (or non-parametric) machine learning techniques to synthetics generation. The earliest attempt is claimed by [201] in which the authors proposes a four-layer, "multi-task deep LSTM learning architecture" for learning (predictive) models of movement and transportation mode from GPS traces [201].

In a more recent (2019) [105], the authors note that often the datasets used when training a neural network are of limited size due to privacy and access constraints and

propose the use of a variational autoencoder to construct a hidden space which captures characteristics of the input and can be used to construct synthetic trajectories. Results indicate the potential of the solution but the error may be too high for some applications.

[159] presents a state-of-the-art, GAN-based (generative adversarial network) technique for trajectory generation that utilizes trains a neural network (called the *generator*) to produce trajectories which are indistinguishable from truth. The resolution of the trajectories is dependent upon the network grid discretization used. Other GAN-based techniques for generating synthetic trajectories include [74], [128], and [56].

2.4 Comparison to GeoAware

Given the brief introduction of the framework in chapter 1 and the just presented literature review, one can see that there are similarities and differences between the proposed approach and the relevant literature. Probably the most important similarity is that the GeoAware framework appears to span all three categories. While the framework appears to most closely fit within the network constrained MOG category, it also has features that fall into the other two categories as well. Specifically, the the GeoAware framework and the reviewed MOGs share the same purpose, namely, producing trace-level trajectories of moving objects.

Nevertheless, unlike many MOGs, the GeoAware framework does not resort to purely heuristic models but instead produces synthetics that are backed by a rational, seasonally-aware process. The incorporation of a model that is not merely heuristic resembles the approach taken by the reviewed human mobility models (see section 2.2). The GeoAware framework is also similar to the reviewed machine learning approaches as it also posits that mobility models are complex and subject to definition errors and expressiveness issues. As such, the models used in GeoAware do not attempt to reason about the observed patterns but instead only aim to accurately capture such patterns.

Just like other MOGs, the GeoAware framework also focuses on providing an analyst accessible framework for producing synthetics. In contrast, the mobility models and machine learning approaches are typically not designed with such a goal in mind and therefore typically require much more analyst involvement in order to obtain usable trajectories. The focus on producing a framework means that the goal is not merely to present a method by which synthetics may be produced but is instead motivated by the need to make synthetics generation available in many contexts where analyst knowledge of human mobility is limited. In GeoAware, accessibility is achieved by making the entire framework controllable from the `R` programming environment. The MNTG framework shares this philosophy and presents a web-based portal for producing synthetics using the reviewed, Brinkhoff [40] and BerlinMOD [64] generators [151]. [163] presents a similar framework focused on human mobility models. The framework allows an analyst to load, represent, clean, analyze, generate and assess the privacy risk of mobility data.

Probably the chief difference between the proposed work and the reviewed literature is the focus the GeoAware framework places on using field data to configure the MOG so that it produces synthetics which are realistic. To be sure, recent MOGs increasingly incorporate field data for calibrating generator parameters [84] [170] [200] [108] [102], but the integrated approach and the data source used by GeoAware is uncommon in the literature. Of the reviewed MOGs, only a few specifically use trajectories when constructing their models for synthetics generation [170] [102] [200]. Of these, none present a modular and reusable component aimed at producing calibrated settings from such data. Instead, because of the extremely general nature or a highly specific application, an analyst would have to properly ingest the trajectory data in order to utilize the framework. In contrast, the GeoAware framework presents an approach which only requires an analyst to point to the proper source files. In many modules, if an analyst wishes to provide a custom algorithm it can be done but he/she need not do this in order for the framework to function.

Furthermore, in many of the reviewed MOGs the calibration of generator settings is

limited to minor parameter settings or specifying the infrastructure. While such settings are important, they are not sufficient for producing realistic trajectories. Of course, the GeoAware framework does not calibrate all possible settings but by using low-level trajectories, the GeoAware framework calibrates the demand used to generate the synthetic trajectories. The idea is that if realistic demand can be provided to a microscopic traffic simulator, the simulator's car following, lane changing and gap acceptance models can be used to produce realistic synthetics.

Chapter 3 — Methodology

The GeoAware framework for producing intelligently-calibrated synthetics is illustrated in figure 3.1 and can be divided into two main components. The **geoaware** component calibrates the demand component of a vehicle simulation scenario using inferred mobility patterns from the supplied trajectory data while the **libsumor** component generates the desired synthetic vehicle trajectories by building a simulation from the produced scenario and running it. Splitting the framework in this manner highlights the distinction between building a scenario to represent a real-world environment and the actual running of that scenario to produce the desired synthetic trajectories. This partitioning also enhances code reusability, as the base-level simulation engine becomes a reusable module that performs simulations according to a given scenario. The inputs to the framework are (i) trajectory data, (ii) a roadway network, (iii) community definitions, (iv) timing parameters, and (v) a configuration file. Before proceeding to detail the framework and its use of such information, we start with a high-level sketch that illuminates the interaction of the various components.

To obtain the synthetic vehicle trajectories we seek, it is necessary to build a simulation scenario for the transportation system under study. A simulation scenario is a representation of a real-world system that is described by a fitted model. A simulation scenario is a powerful data structure as it allows an analyst to adjust properties of the model in order to analyze various situations. For instance, one scenario might represent the current-state

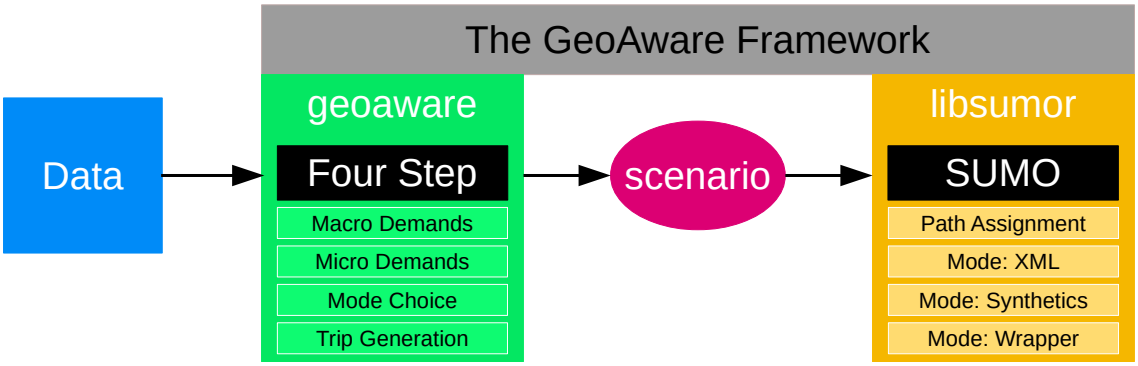


Figure 3.1: High level component decomposition of the GeoAware framework

whereas under a different parameter set a future-state scenario may be represented.

The construction of a calibrated simulation scenario is handled by the **geoaware** component and is represented by the green block in figure 3.1. The first step towards creating such a scenario is specifying the location of the required inputs. Section 3.1 presents this discussion, detailing the relevance and formatting requirements of the (i) trajectory data, (ii) roadway network, (iii) community definitions, (iv) timing parameters, and (v) configuration file. This step is represented by the blue data block in figure 3.1.

Next, in section 3.2 we construct a model of agent demand from the reference trajectories. We align our work with the state of practice in urban demand modeling and present an approach based on the traditional, four-step transportation model [3]. This fact is noted by the black block shown within the **geoaware** component of figure 3.1. Section 3.2.2 opens with a historical preliminary that justifies the use of the four-step model in the current work. Afterwards, we construct the various pieces of the four-step model by inferring quantities from the reference trajectories. Specifically, this means (i) inferring demand through seasonal, macro- and micro-level procedures as detailed in section 3.2.3; (ii) specifying the mode of travel as detailed in section 3.2.4; (iii) generating stochastic trips and (iv) assigning paths to the trips using the procedures presented in section 3.2.5. In figure 3.1, these steps are represented by the green sub-blocks of the **geoaware** component and the *Path*

Assignment sub-block of the **libsumor** component (path assignment relies on SUMO).

The demand model fitted, the last step of the **geoaware** component is to build a simulation scenario. In our discussion of the framework so far, we have specified the required inputs and outlined how to fit a demand model using inferred mobility patterns but we have not used this data to obtain a more realistic simulation environment. Section 3.2.6 addresses this topic by detailing the components of a scenario. This process is shown as the pink ellipse in figure 3.1.

At the conclusion of the **geoaware** component, a scenario calibrated according to the observed mobility patterns now exists. The next step is to evaluate this scenario in order to obtain synthetic vehicle trajectories. Due to the complex (and often unobservable) set of interactions in transportation models, realistic evaluation becomes intractable using analytic approaches [70] [169]. Instead, simulation must be used to evaluate the scenario in order to capture the rich set of intricacies that analytic approaches fail to recover [70].

In the GeoAware framework, we have chosen to use the SUMO microscopic traffic simulator due to its open-source nature, its extensive suite of features and its ability to be extended [123]. We implement our extension for scenario evaluation in the **libsumor** component as detailed in section 3.3. The **libsumor** component is shown as the orange block in figure 3.1. The black box within the **libsumor** component notes that we rely on SUMO to perform the microscopic simulation.

The **libsumor** component takes the specified scenario and evaluates it via a custom-built interface with the SUMO engine. The network provided at scenario definition details the roadway on which to route the simulated agents and details supply constraints such as the number of lanes, junction locations, speed limits, traffic lights, and many other such features. The trips specified in the scenario guide the onset and destination selection of the simulation routing procedure such that the generated vehicle paths reflect the patterns inferred from the reference trajectories. The synthetic mobility records are then extracted by retaining the relevant statistics from each of the simulated agents injected into the sim-

ulation. In addition to producing synthetics, the **libsumor** component also provides XML-based simulation support for SUMO as well as a prototype wrapper class that provides enhanced control of the SUMO simulation engine from within the R programming environment. The various operation modes are shown as the orange sub-blocks of the **libsumor** component in figure 3.1.

3.1 Data Preprocessing

In order to obtain a simulation scenario that is calibrated from micro- and macro-level mobility trends, the GeoAware package requires data inputs detailing the (i) trajectory data, (ii) roadway network, (iii) community definitions, (iv) timing parameters, and (v) a configuration file. Where necessary, these inputs must undergo a preprocessing step to ready them for use by the framework. In this section, we detail the relevance of such inputs and the formatting requirements for each. We begin by considering the reference trajectories.

3.1.1 Reference Trajectory Data

A trajectory (also known as a mobility-record) represents a geospatial trace of agent movement across time. The detail level of the trace may vary from being minimalist, only including details such as the onset and terminus [156], to exhaustive, specifying the entire path traveled [190]. To a large extent, the level of detail dictates the applicability of the trajectories for certain applications.

The positions of agents are captured through a location acquisition system such as GPS (global positioning system), GSM (global system for mobile communications) or WAMI (wide area motion imagery) [174]. As noted earlier, mobility-records are capable of providing powerful insights which aid in knowledge discovery of environment dynamics. In the present work, we are interested in inferring and replicating certain macro-level mobility

trends as well as, to the degree supported by data, micro-level trends present within such trajectories since such trends are indicative of the dynamics driving a geospatial environment.

The macro-level inference performed within the GeoAware framework requires that the trajectories minimally detail a starting and ending *region* with associated timestamps whereas micro-level inference requires the specification of a starting and ending *location*. Minimally then, we require that the trajectory data CSV provided to the GeoAware calibration procedure detail at least the onset and terminus locations with rows consisting of the four-tuple of attributes `(<id>, <timestamp>, <x>, <y>)` for the onset and terminus locations of each agent (when no precise location data is available, the `x` and `y` locations are set to the centroid of the region).

3.1.2 Roadway Network

The layout of roadways and intersections within a geographic environment may be naturally modeled as a directed graph $\mathcal{G} = \{V, E\}$ where the vertices V represent intersections and the edges E represent directed roadways between vertices [81]. To make the graph more realistic, one often ascribes various features to its components, such as the intersection type (traffic light, all-way stop, 2-way stop, etc.), intersection location, speed limit, shape of the roadway, or number lanes. We refer to such a graph and its collection of features as a roadway network map. A map is foundational when generating synthetic traces as it specifies the physical constraints of a roadway environment as well as, when loaded into a simulation, the temporal aspects of the environment (e.g. speed of link under demand). The accuracy and detail of the map directly affect the accuracy and detail of the generated mobility-record traces.

Given the importance of obtaining an accurate roadway network map, a logical next question concerns how to obtain such a map. Unfortunately, no single answer exists as each situation requires a different level of accuracy and may mandate a particular data

source. Typical *free* sources for roadway maps include OpenStreetMap[1], the Census Bureau's TIGER/Line Shapefiles[2] or governmental authorities. In situations where a high degree of accuracy is required such sources may not be enough and therefore obtaining a roadway network may become a more manual process, requiring the fusion of multiple resources. In projects with more moderate accuracy requirements, minor adjustments to such sources may be all that is necessary. Such maps may be stored in a variety of file types, with the most popular being shapefiles, GeoJSON (JSON with geometry support), GML (Geography Markup Language), OSM (open street map XML), and KML (Google Keyhole Markup Language).

The SUMO traffic engine used by GeoAware details its own XML-based format for defining roadway networks[3], making the conversion of a source map to the SUMO format an almost certainty. The SUMO specification splits the network into five (5) different XML files that detail the nodes, edges, edge types, connections and traffic light logic associated with the network. Once the files are specified, the SUMO `netconvert` command must be issued to generate a single network file (`*.net.xml`) from the component files.

Thankfully, SUMO natively supports conversion from a handful of sources, including shapefiles and the OpenStreetMap XML format (OSM), avoiding the need to hand-define the component files. Nevertheless, despite the native support, this does not mean that everything will be converted as expected or without any analyst input. At a minimum, a review of the converted network is necessary but more often parameter tweaking is required in order to obtain a network of the desired quality. The GeoAware framework expects the paths to the various network component files.

[1] https://www.openstreetmap.org

[2] https://www.census.gov/geographies/mapping-files/time-series/geo/tiger-line-file.2019.html

[3] https://sumo.dlr.de/docs/Networks/PlainXML.html

3.1.3 Community Definitions

The ability to infer mobility patterns assumes that the roadway network may be decomposed into a collection of known and disjoint, geographically-defined communities. Such community definitions provide a level of aggregation, allowing for the recovery of trends that may not be noticeable (or accurate) when only observing individual tracks.

The communities are merely a polygon (or a collection of polygons) that is associated with a unique identifier. Typical examples include census tracts or communities within a city, although, grouping by different metrics, such as income, is also possible. The framework expects the community polygons to be defined in a single CSV file with a row per polygon point that details the following attributes:
`<region-label>, <community-label>, <x>, <y>`. The `<region-label>` field must be unique to each polygon, however, the `<community-label>` may be reused, allowing communities to consist of multiple regions.

The traffic analysis zones are built from the community definitions provided and therefore it is assumed that TAZs are known prior to utilizing the GeoAware framework. In the GeoAware framework, traffic analysis zones (TAZs) are a geographic discretization of a roadway network into delineations of edges that share common properties. A TAZ may be any size depending upon the needs of the analyst. The GeoAware framework represents TAZs as objects which associate the edges belonging to TAZ with a label. The TAZ may have any number of fields, allowing for storage by other components in the framework and extensions. TAZs are used to define the blocks used by the SDSBM-based trip demand procedure and for inferring micro-level inference at a per-TAZ level.

It is known that arbitrary delineations can skew the statistics obtained [158], therefore, care should be taken when constructing the community delineations. In traditional transportation modeling, zones are typically coordinated with census data in order to determine demographic properties.

3.1.4 Timing Parameters

The timing parameters control the aggregation used when inferring patterns and the resolution of the produced synthetics. There are three parameters that must be specified. The SDSBM *fit intervals* control the level aggregation used when inferring macro-level demands. Coarse resolutions result in more data aggregation. The fit intervals must be provided to the framework as a two column CSV file with headers `<onset-time>` and `<terminus-time>` where each row is assumed to represent a sequential interval $(a, b]$.

The *seasonal periodicity* parameter details the length of the seasonal cycle. This value is set based upon analyst intuition and/or exploratory data analysis. For instance, to model mobility with weekly seasonality, the analyst would set the periodicity parameter to a week. The periodicity is used by both the macro- and micro-level inference procedures.

The *simulation granularity* controls the resolution of the produced synthetics by subdividing the SDSBM fit intervals according to the granularity specified. This parameter is necessary to translate the macro-level demands into a finer resolution as the macro-level procedure scales with the sample size and it is therefore often impractical to set the fit intervals to the resolution that is ultimately desired.

3.1.5 Configuration File

The configuration file controls various parameters pertaining to the **libsumor** component. The file serves three (3) main functions, namely, (i) configuring the simulation engine, (ii) configuring the **libsumor** mode, and (iii) detailing the locations of required files. The file is YAML-based[4] and hence, encodes the settings and corresponding values as a series of `<key>: <value>` pairs. A table summarizing the available keys is presented in table 3.1. Keys requiring additional detail are discussed below.

[4] https://yaml.org

Key	Description	Required
`sumocfg`	**string** — Where to write the SUMO configuration file (`.sumocfg`) to.	✓
`input.net-file`	**string** — Location of the network that should be used during simulation.	✓
`input.route-files`	**string** — Where to write the constructed trips to.	✓
`input.additional-files`	**string** — Where to write the additionals (`*.additionals.xml`) file to. Stores vehicle type information here.	If `vType` is set.
`<configType>.<setting>`	**string** — General format for specifying arbitrary SUMO parameters.	
`vType`	**collection** — Specifies custom vehicle types. See body text for formatting.	
`geo`	**bool** — Store trajectory positions in Cartesian (`FALSE`) or latitude/longitude format (`TRUE`). Requires *synthetics* mode to be effective.	
`csv`	**string** — Where to write trajectories CSV. Need not be supplied if all records are to be stored in memory. Requires *synthetics* mode to be effective.	
`csv_detail`	**collection** — Bitmask indicating CSV level of detail. See body text for formatting details. Requires *synthetics* mode to be effective.	If `csv` is set.

Key	Description	Required
`csv_precision`	**integer** — Precision when writing out floating point values. Requires *synthetics* mode to be effective.	If `csv` is set.
`csv_sep`	**string** — The CSV separator character. Requires *synthetics* mode to be effective.	If `csv` is set.
`depart`	**bool** — Filter synthetics by departure time field. Requires *synthetics* mode to be effective.	
`edges`	**collection** — Filters the synthetic vehicles produced by the edges in the provided list. See body text for formatting details. Requires *synthetics* mode to be effective.	
`intervals`	**collection** — Filters the synthetic vehicles produced by the supplied time intervals. See body text for formatting details. Requires *synthetics* mode to be effective.	

Table 3.1: Configuration file parameters

The keys used to configure the SUMO simulation engine come from the XML-based SUMO configuration schema[5] and provide an analyst with fine-grained control over such quantities as the time settings, processing parameters, routing algorithm and output. The keys are constructed by preprending the configuration type of the setting (a type of parent class denoted in the schema as `configurationType`) to the setting name with a dot

[5] https://sumo.dlr.de/xsd/sumoConfiguration.xsd

(“.”) between the quantities. For instance, the key for configuring the file path to the roadway network is `input.net-file` since the `net-file` setting is a member of the `input` configuration type.

The keys used to configure the operating mode allow an analyst to control the detail of the produced synthetics, specify edge- and time-based filtering, and specify vehicle types. The `csv_detail` key takes a bitmask value that allows an analyst to control the amount of data written out to the produced CSV file. An example bitmask with the associated fields is presented in table 3.2. In this example, the `<id>`, `<x>`, `<y>` and `time` are saved only for the bounds of the trip, namely the onset and terminus locations.

bounds	id	type	x	y	time	speed	angle	distance	edge
1	1	0	1	1	1	0	0	0	0

Table 3.2: CSV output bitmask

The **libsumor** component also supports multiple vehicle types and is further detailed in section 3.2.7. The vehicle types are defined through the `vType` key which takes a YAML-encoded collection of dictionaries: `[{id: <type1>, ...}, {id: <type2>, ...}, ..., {id: typen, ...}]`. For example, an analyst could define a `taxicab` type through the following statement: `vType: [{id: taxicab, accel: 5, maxSpeed: 70, vClass: taxi}]`. The SUMO simulation engine supports many vehicle type attributes[6] controlling the maximum speed, acceleration, car following model, vehicle class, vehicle capacity, and GUI parameters.

The `depart`, `edges` and `intervals` keys define the filtering that is performed on the produced synthetics. For the `edges` key, if `depart: TRUE`, the synthetic vehicle must depart from an edge in the provided list in order to be retained, otherwise, only the portions of the trajectories that coincide with the edges in the provided list are retained from each of the produced vehicles. In the `intervals` key, if `depart: TRUE`, the

[6] https://sumo.dlr.de/docs/Definition_of_Vehicles,_Vehicle_Types,_and_Routes.html

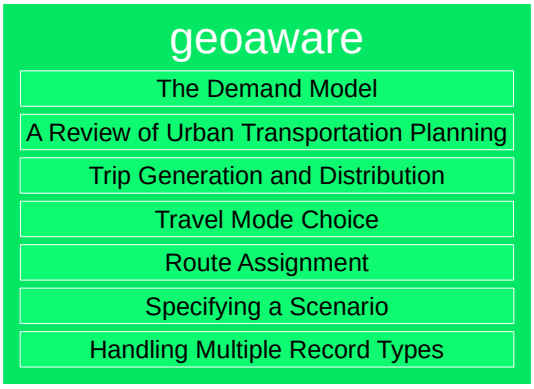


Figure 3.2: Breakdown of the `geoaware` component

synthetic vehicle must depart during a valid interval, otherwise, only the portions of the trajectories that coincide with valid intervals are retained from each of the produced vehicles. The intervals are supplied as consecutive intervals in ascending order. For example, `intervals: [a, b, c, d]` corresponds to the intervals $[a, b)$ and $[c, d)$.

3.2 The `geoaware` Component

The data preprocessing steps complete, we now define the `geoaware` component. The job of the `geoaware` component is to produce a demand-calibrated simulation scenario. The `geoaware` component achieves this goal in two steps. First, the supplied data is ingested and a model of the agent demand produced. Next, the demand model is stochastically queried to produce a simulation scenario with calibrated demand.

The organization of this section is centered around these two steps. Sections 3.2.1 — 3.2.5 detail the production of the demand model while sections 3.2.6 and 3.2.7 discuss the production of a demand-calibrated simulation scenario. A graphical overview of these sections is shown in figure 3.2. Throughout the discussion, we provide updated graphics which indicate our current position within this figure.

3.2.1 The Demand Model

We begin by detailing the model we will use for constructing demand. Vehicle mobility models fall into either trip- or activity-based approaches, with the former being far more prevalent [38] [206] [192] [8]. Trip demand modeling is the process by which statistical properties of trips are inferred from data and used to forecast (predict) the amount of trips that should be generated (demanded). In the context of our current application, a trip demand model is necessary for two reasons. First, we desire to extract *patterns* of mobility exhibited in the trajectories. We are not merely interested in replicating the trajectories but instead seek to discover patterns that are characteristic of the data. The second reason why a demand model is necessary is so that it can be queried during the production of the synthetic vehicle traces. Having a model makes the querying process trivial.

Arguably, the most famous (and infamous) demand model is the Urban Transportation Planning (UTP) procedure which presents a trip-centric, four-step process for modeling trip demand [192] [147]. Such an approach is contrasted with the activity-centric view of activity-based approaches. Activity-based approaches argue that travel (i.e. trips) is the consequence of actors who need/want to engage in certain geospatially located activities and therefore emphasis is placed on modeling the *behavior* that produces travel demand[7] [206] [8]. Regardless of the approach taken, its important to understand that demand modeling is not purely scientific as it involves irrational humans and therefore, since all models will by their nature be "metaphors" of reality, the best one can hope for is a model that provides the necessary level of detail and *approaches* the true state while being fully aware

[7]For a thorough discussion of activity-based demand models, please see the book-length treatment published by the Transportation Research Board (TRB) [45] or the academic paper by Algers, Eliasson and Mattsson [8].

that such a model is not the *true* state [192].

Given our deliberate desire to remain as data-driven as possible and avoid placing any unnecessary assumptions on the data, we have chosen to utilize UTP-based procedure for modeling demand in the GeoAware framework due to the lack of behavioral data in trajectories (as well as the general lack of behavioral data [192]). In aligning our work with the UTP procedure, we do not intend to present a framework for traffic planning (although it could certainly be used for it) nor are we unaware of drawbacks associated with the UTP procedure. Instead, despite the drawbacks, we utilize the UTP procedure as it presents an established, logical, and extendable procedure for constructing a trip demand model capable of producing the synthetics we seek. Additionally, due the prevalence of the UTP procedure in Metropolitan Planning Organizations (MPOs are the federally recognized organizations responsible for transportation planning within an urban area) [206] [192], aligning our approach with such a procedure supports the immediate utility of the work.

The UTP procedure is an iterative approach to travel demand modeling that is commonly employed by traffic forecasters to assess future travel demand in response to land usage, population demographics and/or infrastructure [25]. Using forecasts of land use, population, employment and infrastructure, future estimates of travel demand and the induced supply-demand interactions can be obtained using the UTP procedure [25].

The procedure is divided into four sequential steps and is really an abstraction of a traveler's decision making process. As noted in [206], the itemization of four steps is not meant to imply that travelers go through these exact four steps in the order specified but is instead a model of traveler behavior which produces suitable results in traffic research [38] [206]. The model's fundamental unit is a trip which represents the movement of some quantity of people (e.g. a single individual or a vehicle with riders) between an onset and terminus location without any intermediate stops [206].

The initial step[8] in the UTP procedure is the *trip generation* step. This step seeks to detail how much traffic should be attracted to and departing from an aggregate geographic region known as a TAZ based upon the land use and demographic indicators associated with the TAZ [206]. Trips are segmented by purpose under the assumption that different trip purposes give rise to different behavioral characteristics [206]. The three classic trip purposes are "home-based work", "home-based nonwork" and "non home-based" each of which illustratively detail the general goal of the trip [206].

The next step is *trip distribution* and it is concerned with disaggregating the trip productions and attractions produced for each TAZ during the trip generation step into macro-level flows between TAZs in an attempt to satisfy the demanded flow properties (incoming and outgoing) for each TAZ. A gravity model is typically employed for the decomposition [206].

The next step is known as *mode assignment* and it further decomposes the trip distributions by travel mode based upon a variety of factors [90] [206]. Two popular mode choices are the private vehicle and public transit.

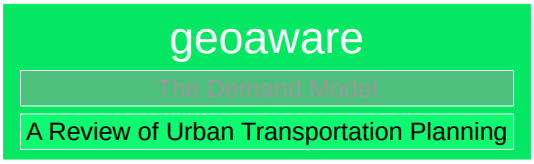
The final step, and arguably the most involved step, is called *route assignment*. Route assignment is concerned with assigning routes through the network to the trips demanded by the trip distribution step according to the chosen modalities [206]. This step is done so as to obtain the induced affects of supply-demand interactions, such as edge flow and congestion, on travel times through the network [54] [206].

In sections 3.2.2 — 3.2.5, we detail each of these steps and indicate how the model is constructed from the supplied data. Specifically, we begin with a brief history of the UTP procedure in section 3.2.2 which establishes the prevalence and extensibility of the UTP procedure. Then, in section 3.2.3 we detail the trip generation and distribution steps via the use of macro- and micro-level inference procedures. Specifically, we (i) infer macro level travel patterns between TAZs using an SDSBM, (ii) capture the temporal distribution of

[8] We present the four steps in the traditional order but alternative orderings do exist. See [38] for a discussion.

trips within each block, and, when available, (iii) capture the edge selection choices. Next, section 3.2.4 details the mode assignment step by specifying how the GeoAware framework supports multiple modalities. Last, in section 3.2.5 we detail how routes are assigned to each of the trips demanded.

3.2.2 A Review of Urban Transportation Planning



In this section we present a brief history of the UTP procedure. This presentation is meant to be more than merely informative and seeks to establish a context for the proposed modeling approach and demonstrate the prevalence of the UTP procedure throughout the history of urban transportation planning. For a more thorough treatment, the interested reader is encouraged to consult the works of Shuldiner and Shuldiner [192], Boyce and Williams [38] and Jones [112].

Urban transportation planning originated in the United States and arose out of a culmination of advances in the 20th century. The dominance of the United States' economy, widespread automobile ownership, urbanization (both urban and suburban) and an array of infrastructure projects (most notably the Federal Aid Highway Act of 1956) resulted in an ever-increasing number of vehicles on the roadway and its inevitable side-effect: congestion [192] [30] [112]. Obviously an annoyance when traveling, early traffic planning (prior to the 1960s) focused predominantly on measuring traffic and "fixing" congestion through infrastructure capacity additions [192]. Congestion still remains a central motivation for traffic planning today.

By the mid 1950s, because of the large-scale highway infrastructure contemplated with the passage of the 1956 Federal Aid Highway Act, a paradigm shift was needed in

order to appropriately model the contemplated situations. Two foundational transportation studies of Chicago, Illinois (1955-1959) and Detroit, Michigan (1955-1956) during this era led the way in what would become the current, decades old practice of using land-use characteristics to forecast travel demand [192] [206]. The Chicago-based study was the first transportation study to use a process resembling the current four-step UTP procedure [38].

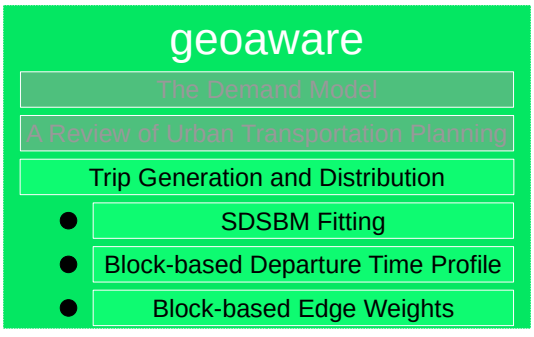
The 1960s saw the solidification of the land use-based, four-step UTP procedure and its prevalence in traffic planning due to the passage of the 1962 Federal Aid Act which hinged the expenditure of Federal funds in urban areas of more than 50,000 in population on some form of transportation planning [192].

The 1970s brought many lasting changes to transportation modeling. Various disaggregate improvements to the UTP procedure based upon microeconomic consumer demand theory were proposed [227] [177] [224] [10] [37] [38], some of which have become standard practice [206]. Dynamic traffic assignment (DTA) materialized as researchers sought to model variations in traffic flows and conditions which were, up to that point, assumed to remain static (time invariant) throughout the forecasting period [169] [54]. The use of simulation became widespread during this era (with an "explosion" during the 80s and 90s) and as a result, the mainframe-based modeling programs were replaced by private-sector (or academic), personal computer-based applications [137] [35].

Activity-based approaches also began to appear in the 1970s, bringing activity-based travel theory to transportation modeling in order to rectify certain inadequacies of the UTP procedure [37] [92]. Such approaches were fundamentally different than those previously proposed as they saw travel as merely demand derived from individuals who need/desire to engage in certain geospatially located activities [37]. Notable examples of activity-based frameworks include [186], [37] [89], [182], [181] and [216]. Activity-based approaches have been applied in cities such as San Francisco, California; New York, New York; and Columbus, Ohio [206].

As we advance to the current era in transportation modeling, despite an ever growing collection of traffic planning related literature, the theoretical popularity of activity-based approaches (particularly in academia), and the overwhelming amount of data available (or capable of being captured) regarding an individual, the state of practice in travel planning has not changed substantially since the introduction of the four-step UTP procedure back in the 1950s [35] [38] [206]. The push for deployable solutions by traffic planning practitioners has resulted in a rift between the practical and the theoretical which every so often is theoretically challenged but more often is bridged by various theoretical extensions which aim to solidify the predominantly pragmatic underpinnings of practical approaches. [38] hints at this notion in their review of travel forecasting when they say that "the remarkable longevity of those models of the 'traditional form [such as the UTP procedure],' however, is due to their capacity to absorb innovations" and "in spite of known deficiencies, professionals trained in the use of these methods are, on the whole, comfortable with their results." It is within this historical context that we introduce the **geoaware** component; another extension to the UTP procedure which increases the realism of the demand produced.

3.2.3 Trip Generation and Distribution



The first step in our model towards generating calibrated trip demand is to generate trips. In the UTP procedure, this step is separated into trip generation and trip distribution

steps[9] [206]. The goal of the trip generation step is to generate a pair of trip productions and trip attractions per TAZ over the analysis interval [226]. The idea is that features of the TAZs induce a certain production and attraction of trips. For instance, a central business district in a city is likely to attract many more trips than it produces in the morning hours due to the high number of businesses within the area.

The trip distribution step of the UTP procedure disaggregates the trip productions and attractions into macro-level flows between pairs of TAZs [226]. The output is typically represented in an origin-destination matrix (O-D matrix); so called because the form succinctly codifies the distribution of the trips between the origin and destination TAZ pairs as a matrix where the rows and columns represent the TAZs. This step can be thought of as attempting to satisfy the flow constraints specified by the production and attraction pairs articulated during the trip generation step[10]. As there are a myriad of onset-terminus pairs that satisfy the flow demanded, a model must be used to select the relative frequency of each pair. For instance, consider a three TAZ network where the TAZs are categorized according to whether the land is predominantly used for residential dwellings, commercial establishments or dining/recreation. In this environment, the trip generation step might say that for the time period under study the residential TAZ should produce 1000 trips. The choice in model dictates how the required trips should be distributed between the TAZs.

As one may have noticed in the above discussion, the trip generation and trip distribution steps of the UTP procedure are merely a procedural decomposition of the original trip creation objective. While such a decomposition may be necessary in methodologies that rely on aggregate statistics which contain no information regarding the distribution of trips, the use of disaggregate statistics may make this decomposition unnecessary if the higher

[9] Strictly speaking, the UTP procedure segments by trip purpose before generating trips, however, in the GeoAware framework we do not consider trip purpose as the SDSBM natively supports such demarcations through its block structure if desired.

[10] We note that a drawback of estimating productions and attractions separately is that while the total number of productions and attractions across the network are equal in theory, the separate estimation of these quantities during the trip generation step may produce unbalanced productions and attractions in practice [206].

fidelity data implicitly includes such trip distribution information. The reference trajectories used by GeoAware are such a source and for this reason, we merge the trip generation and distribution steps into one simultaneous step[11].

In the **geoaware** component, the creation and distribution of trip demand begins by using a SDSBM to infer macro-level, seasonal patterns between TAZs in the reference trajectories. Many authors have cited the seasonal (or regular) nature of mobility patterns [199] [142] [87] [184] [157], making a process capable of inferring seasonal trends valuable for accurately modeling mobility. Next, the temporal distribution of the trips is captured through a micro-level inference procedure that constructs seasonal time profiles of trip departures for each of the SDSBM blocks. Finally, to disaggregate the distribution of trips to the edge level (i.e. roadway), an edge-weight inference engine captures the edge selection behavior of agents at a per-block level by producing a seasonally-aware edge selection distribution for the onset and terminus TAZs of each block.

SDSBM Fitting

To accurately capture the quantity and macro-level distribution of trips from reference trajectories, we utilize a dynamic stochastic block model (DSBM) that is specifically designed to capture the seasonality exhibited in mobility patterns. The model is known as a seasonal DSBM (SDSBM) and has been shown to accurately model regular, time dependent seasonal processes[12] [183].

The SDSBM, like all stochastic block models (SBMs), requires that the data to be modeled is formatted as a graph $\mathcal{G}$ consisting of a set of vertices $\mathcal{V}$ that represent communities and a set of edges $\mathcal{E}$ that represent movement between communities [85]. To construct such vertices $\mathcal{V} = \{1, ..., n\}$ from the reference trajectories, we associate the onset and terminus locations of the trips with the TAZ (defined from the supplied community definitions; see section 3.1.3) that encompasses each location. In this way, the multitudinous

[11] Various steps have been merged in earlier literature [47] [212] [225].

[12] As an applied work, we present only a review of the SDSBM. The interested reader is encouraged to consult [183] and [184] for a more thorough and theoretically focused discussion.

node locations associated with trip onset and termination (i.e. residences, marketplaces, coffee shops, workplaces, etc.) are made structurally equivalent to a set of super-nodes representing communities of interest [85]. Edge formation in the graph is assumed to be dependent upon the (i) communities to which the incident nodes belong and (ii) time [183]. The time interval of study $\mathcal{T}$ is discretized into T intervals $\mathcal{T} = [1, 2, 3, ..., T]$ and edges are only drawn between vertices if a trip (now represented by an onset TAZ, terminus TAZ and trip duration) is active during the considered time interval. This produces a time-ordered series of static graphs $\mathcal{D} = \{\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_T\}$ which we call a dynamic network since the edges appear and disappear (i.e. are dynamic) with time. The time interval information is supplied via the SDSBM fit intervals presented in section 3.1.4.

The **geoaware** component *represents* the dynamic trajectory data network $\mathcal{D}$ as a time-ordered set of adjacency matrices $\mathcal{A} = \{\mathcal{A}_1, \mathcal{A}_2, ..., \mathcal{A}_T\}$. Each adjacency matrix $\mathcal{A}_t$ is a $n \times n$ matrix where the rows and columns represent the vertices and the value in cell $[\mathcal{A}_t]_{i,j}$ indicates the number of active trips in the current time interval t between communities (vertices) i and j[13]. In SBMs, each directional pairing of vertices (i, j) is known as a block and each block's observations $y[t]_{(i,j)}$ are assumed to be characterized by a latent (unobserved) random variable $\mathbf{x}[t]_{(i,j)}$ [148]. The relationship between a block's edge counts in the adjacency matrices $y[t]_{(i,j)}$ and its latent random variable $\mathbf{x}[t]_{(i,j)}$ is given by a state space model[14] (SSM) [93]. Specifically, for each block (i, j) the edge counts $y[t]_{(i,j)}$ are related to the $p \times 1$ latent signal vector $\mathbf{x}[t]_{(i,j)}$ according to the Bayesian linear model

$$y[t]_{(i,j)} = \mathbf{h}\mathbf{x}[t]_{(i,j)} + w \tag{3.1}$$

where $\mathbf{h}$ is a $1 \times p$ observation vector that transforms the latent signal state into the obser-

[13] For efficiency purposes, the GeoAware framework actually linearizes the collection of adjacency matrices to form a single $T \times N$ matrix where each cell represents the quantity of trips for a block (pair of vertices) during a specific time interval.

[14] The power of state space model representations lies in their ability to relate observations to unobserved latent states through an observation equation. For an excellent treatment on SSMs, the reader is encouraged to consult chapter 3 of [93].

vations and w is Gaussian-distributed observation noise with probability density function (PDF) $w \sim \mathcal{N}(0, \sigma^2_{obs})$. The $p \times 1$ latent signal vector $\mathbf{x}[t]_{(i,j)}$ evolves with time according to the Gauss-Markov model

$$\mathbf{x}[t]_{(i,j)} = \mathbf{A}\mathbf{x}[t-1]_{(i,j)} + \mathbf{B}\mathbf{u} \tag{3.2}$$

where $\mathbf{A}$ is a $p \times p$ transition matrix, $\mathbf{u}$ is a $r \times 1$ transition noise vector distributed according to $\mathbf{u} \sim \mathcal{N}(0, Q)$ and $\mathbf{B}$ is a $p \times r$ matrix that applies the noise to the correct terms of the signal state $\mathbf{x}[t]_{(i,j)}$. The transition noise $\mathbf{u}$ of the signal model serves to propagate the signal vector $\mathbf{x}[t]_{(i,j)}$ through time whereas the observation noise w represents the uncertainty in the observations. The *a priori* parameters of the model are $\theta = \{\mathbf{A, B, h}, \sigma^2_{obs}, \mathbf{Q}\}$.

With the structure of the block model defined, we now identify the elements of θ. To capture the seasonality exhibited within the edge counts of each (i, j) block, [184] proposes the use of a basic structural model (BSM) [93]

$$y[t]_{(i,j)} = b[t]_{(i,j)} + s[t]_{(i,j)} + \epsilon_{(i,j)} \tag{3.3}$$

consisting of a bias term $b[t]_{(i,j)}$ that establishes the general signal trend, a seasonal term $s[t]_{(i,j)}$ that shifts the bias according to the current seasonality position and a noise term ϵ. The bias term $b[t]$[15] is computed by adding transition noise $u_b \sim \mathcal{N}(0, \sigma^2_b)$ to the previous bias state

$$b[t] = b[t-1] + u_b \tag{3.4}$$

The seasonal term is calculated based upon the length of the seasonal period p according to the following zero-sum[16] formula

[15] As each (i, j) block is modeled using its own BSM, hereafter, we drop the explicit reference to the block for notational simplicity.

[16] The enforcement of the zero-sum criterion ensures that the seasonal components sum to zero over the seasonal period. The notion behind the zero-sum constraint is that the average of p seasonal components should be 0.

$$s[t] = -\sum_{i=1}^{p-1} s[t-i] + u_s \tag{3.5}$$

where u_s is transition noise distributed according to the PDF $u_s \sim \mathcal{N}(0, \sigma_s^2)$ and p is a hyperparameter set by the analyst that controls the seasonality length while the inclusion of the stochastic noise term u_s allows the seasonal effects to change with time [93].

Transforming the BSM presented in equation 3.3 into the SSM representation of equations 3.1 and 3.2 is trivial as all linear univariate structural models have a state space representation [93]. The bias terms $b[t]_{(i,j)}$ and seasonal terms $s[t]_{(i,j)}$ presented in equation 3.3 represent the latent state $\mathbf{x}[t]$ as we observe the affects of these variables (edge counts) but not the variables themselves. To represent these latent terms according to equation 3.2 we define the latent signal $\mathbf{x}[t]$ as a $p \times 1$ state vector[17]

$$\mathbf{x}[t] = \begin{bmatrix} b[t] & s[t] & s[t-1] & \dots & s[t-p+2] \end{bmatrix}^T \tag{3.6}$$

the $p \times p$ transition matrix $\mathbf{A}$ as

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & -1 & -1 & \dots & -1 & -1 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 \end{bmatrix} \tag{3.7}$$

[17] We note that the pth seasonal term is not included in the signal state (keeping the signal state of dimension p) since the zero sum constraint ensures that the pth term can always be recovered using the other $p-1$ seasonal terms.

the $p \times 2$ noise assignment matrix $\mathbf{B}$ as

$$\mathbf{B} = \begin{bmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 \end{bmatrix}^T \tag{3.8}$$

and the covariance matrix $\mathbf{Q}$ of the transition noise $\mathbf{u}$ as

$$\mathbf{Q} = \begin{bmatrix} \sigma_b^2 & 0 \\ 0 & \sigma_s^2 \end{bmatrix} \tag{3.9}$$

The system transitions from state $\mathbf{x}[t]$ to $\mathbf{x}[t-1]$ according to the transition matrix $\mathbf{A}$. Multiplying the first row of $\mathbf{A}$ by the signal vector $\mathbf{x}[t-1]$ computes the bias term $b[t]$ of equation 3.4 without noise. Similarly, multiplying the second row of $\mathbf{A}$ by the signal vector $\mathbf{x}[t-1]$ produces the seasonal term $s[t]$ of equation 3.5 without noise. The remaining $p-2$ rows serve to permute the location of the seasonal terms so each seasonal term is updated after p time steps. The $\mathbf{B}$ matrix identifies the terms in the signal state to which noise is added.

To translate the latent signal into the observed edge counts, we utilize the observation model specified in equation 3.1. In the BSM presented in equation 3.3, the edge counts $y[t]$ are related to the latent space through a simple summation of the bias term $b[t]$, the seasonality term $s[t]$[18] and some noise ϵ. Thus, the $1 \times p$ observation vector $\mathbf{h}$ is simply

$$\mathbf{h} = \begin{bmatrix} 1 & 1 & 0 & \dots & 0 \end{bmatrix} \tag{3.10}$$

It is easily seen that multiplying $\mathbf{h}$ times the the latent signal state $\mathbf{x}[t]$ results in equation 3.3 without the noise term. The noise term is handled by letting $\epsilon = w$ and drawing the

[18] We always reference the seasonal term in the second position of the signal state vector $x[t]$ because the transition matrix A (see equation 3.7) permutes the seasonal terms such that the seasonal term $s[t]$ for the current time interval t is always in the same location.

observation noise according to the PDF of w (defined above).

With the state space model defined, we now proceed to estimate the latent signal states $X = \{\mathbf{x}[t]_{(i,j)}; i, j \in \mathcal{V}\}$ so that an edge count model can be constructed for each block (i, j). Estimating the latent signal state for a particular (i, j) block amounts to finding the posterior PDF $Pr(\mathbf{x}[t] \mid y[t]; \theta)$ for all t using the edge count data observed for that block $y[t]$ and the *a priori* parameter set θ [117]. One extremely common approach to finding the posterior is to estimate it using a Kalman filter[19] [114] which sequentially estimates a signal embedded in noise using observation data [117]. Before proceeding to use the Kalman filter however, we must first estimate the unknown parameters in θ, namely $\theta' = \{\sigma^2_{obs}, \mathbf{Q}\}$. To estimate these quantities, we perform numerical maximum likelihood estimation (MLE) which seeks to find the values of σ^2_{obs} and $\mathbf{Q}$ that maximizes the likelihood of the observations $y[t]$ according to the state space model detailed in equations 3.1 and 3.2. A detailed treatment of the MLE procedure can be found in Chapter 7 of [117].

Once the missing *a priori* parameters have been estimated, the latent signal states $X = \{\mathbf{x}[t]_{(i,j)}; i, j \in \mathcal{V}\}$ governing the edge counts of each block can be estimated using using the aforementioned Kalman filter algorithm. The Kalman filter oscillates between *prediction* and *update* steps[20] for each time t. The goal of the prediction step is to compute a "best-guess" estimate of the current signal state $\mathbf{x}[t|t-1]$ given the immediately previous signal state $\mathbf{x}[t-1]$. The update step refines this prediction (i.e. the prior) through a weighted difference of the observation $y[t]$ and its prediction $\mathbf{x}[t|t-1]$. The weighting used is called the Kalman gain and can be scaled depending upon our trust in the observations. We assume that the latent signal (comprised of the bias $b[t]_{(i,j)}$ and seasonal $s[t]_{(i,j)}$ terms) is Gaussian distributed as are the observation noise $w_{(i,j)}$ and the transition noise $\mathbf{u}_{(i,j)}$, resulting in a minimum mean square error (MMSE) estimator[21] [117].

When the full set of observations is known in advance, as in the present case, a smooth-

[19] A process which estimates a signal $z[t]$ using data $\{c[0], c[1], ..., c[t]\}$ as t continues to increase is called a filter.

[20] Please consult Chapter 13 of [117] for details on each step.

[21] The interested reader is encouraged to consult Chapter 13 of [117] for the derivation.

ing technique can be applied to obtain better posterior state estimates by considering all the available data [184]. The GeoAware package uses the `KFAS`[22] package for performing the Kalman filtering as well as the Kalman smoothing. The smoothing procedure implemented in the `KFAS` package is based on the work of [66]. The central idea of the algorithm is to update the state predictions through a backwards recursive procedure that indirectly allows each state prediction to take advantage of the observations that comes after. For a detailed discussion of the smoothing procedure, the reader is encouraged to consult Appendix A of [98].

Upon termination, the SDSBM fitting procedure outlined above produces edge count fits for each block $\{(i,j); i,j \in \mathcal{V}\}$ by estimating the expected value of the posteriors $E(\mathbf{x}[t]_{(i,j)}|y[1:T]_{(i,j)})$. The edge count fits detail the number of trips to generate for a particular block over a specified time interval and are called trip demands. The resolution of the supplied SDSBM fit intervals directly controls the detail level of the demands obtained.

Block-based Departure Time Profile

The demands produced via the SDSBM fitting procedure are at the detail level specified in the provided SDSBM fit intervals (see section 3.1.4). Unfortunately, due to practical efficiency constraints, the time intervals used in fitting the SDSBM (the provided SDSBM fit intervals) are unlikely to be at the resolution desired for the produced synthetics. Such constraints arise because the MLE procedure and the Kalman filtering procedure scale with the size of the data. Thus, as the data increases in size so too does the processing time. For instance, if the SDSBM procedure detailed above is used to infer mobility patterns over a month of data at a resolution of a day, the resulting process is based on 31 data samples. If however, the same time period is modeled at a resolution of fifteen minutes, the process is now based on 2,976 data samples.

To address this lack of precision in a data-driven manner, we define an empirical inference procedure which constructs a departure time profile for each block based upon

[22] https://cran.r-project.org/web/packages/KFAS/

the departure times exhibited within the reference trajectories. Constructing a separate time profile for each block enables the **geoaware** component to capture block-level travel pattern differences and incorporate such information when generating synthetics. The resolution of the resulting time profile is controllable by a simulation granularity parameter which is one of the required data inputs (see section 3.1.4). By appropriately setting the simulation granularity, macro-level trends can be captured by the SDSBM and further refined according to the empirically modeled departure times.

The **geoaware** component allows an analyst to define a custom profiling procedure that is applied to each block's set of trajectory records or use the built-in method which constructs these time profiles by identifying the departure times associated with a particular block and then constructing empirical PMFs of the departure times for each period in the defined seasonal cycle (set in section 3.1.4). The built-in procedure also supports block-level additive smoothing [155] and Bayesian updating [97] for each period in the seasonal cycle; giving additional control to the analyst and preventing undesired "zeros" when no empirical data exists.

Block-based Edge Weights

Up to this point, the outlined trip creation procedure has inferred (i) macro-level trip demand between TAZs and (ii) micro-level departure time profiles for each block. A facet that is lacking from our model is the selection of a specific onset and terminus location. While TAZ-level aggregation may be useful for inferring patterns across structurally similar nodes (i.e. belonging to the same TAZ), it does not provide any information specifically detailing the onset and terminus locations that should be chosen for trips. To address this need, we present an empirical inference procedure for determining the selection weights of the edges associated with the onset and terminus TAZs of each block. The procedure jointly models the onset and terminus location selection associated with each block and is therefore able to capture the joint affect of the onset and terminus TAZs on onset and terminus location selection. In keeping with the rest of the framework, we assume that the macro-level trip

demand is not at the resolution desired and therefore a simulation granularity parameter has been specified indicating the desired resolution. Additionally, because we are seeking edge-level details, the reference trajectories provided to the framework must be trace level. If trace-level reference trajectories are not supplied, edge-level inference cannot be performed.

To model the edge selection probabilities for the onset TAZ of a block at the resolution specified, the nearest edge associated with the onset location of each trajectory in the block is found. Then, for each seasonally related set of onset records, an empirical PMF is constructed detailing the probability of starting from each of the edges in the onset TAZ. The seasonally related set of records is constructed by subdividing the SDSBM fit intervals into smaller intervals according to the specified simulation granularity and then grouping the records according to an augmented seasonality length[23]. For instance, if the SDSBM fits were at a daily resolution, the desired simulation granularity set to fifteen minutes, and the seasonality length set to seven days, because of the simulation granularity, every 96th interval would be seasonally related.

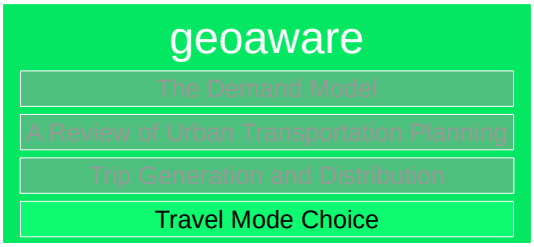
To provide greater flexibility and to avoid "zeros" when assigning the edge weights, the procedure supports block-level additive smoothing [155] and Bayesian updating [97] for each simulation interval in the seasonal cycle. The terminus weights are computed using the above process but now only considering the seasonally related terminus records. The output of this procedure is a pair of multidimensional arrays for each block that details the edge selection weights of the onset and terminus TAZs. The dimensions for these arrays are $e_C \times s \times p$, where e_C is the number of edges in TAZ C, s is the number of simulation bins each fit interval needs to be divided into in order to achieve the desired simulation granularity and p is the number of periods (fit intervals) in a seasonal cycle.

Unfortunately, despite the usefulness of detailed trajectory data for mirroring reality, not all trajectory datasets provide information at a such a fine-grained level (see chapter 1).

[23] Due to the subdivision of the fit interval into simulation time intervals, the originally provided seasonal period is no longer valid and must be augmented to take into account the subdivision.

Therefore, in an effort to avoid making the GeoAware framework overly specific to a particular trajectory dataset, we have also made the edge weight inference procedure extensible to allow different analyst-defined functions to be used when inferring edge weights.

3.2.4 Travel Mode Choice



The travel mode choice step of the UTP procedure is where a mode of travel is assigned to the trip demands identified in the preceding steps [206]. Examples of travel modes include walking, cycling, using a private vehicle, or using public transportation. The typical approach taken to assign travel modes employs some statistical regression model [53]. As presented in [31], the mode choice can affect not only on the quantity of demand but also the temporal distribution of that demand. According to [90], the key factors influencing travel mode choice are individual/household demographics, the built environment, individual preferences and trip-level specifics [193] [189] [32] [213] [63] [26] [204] [144] [219]. Imagining the influence of these decision factors in choosing a travel mode is not difficult as it is a process familiar to most.

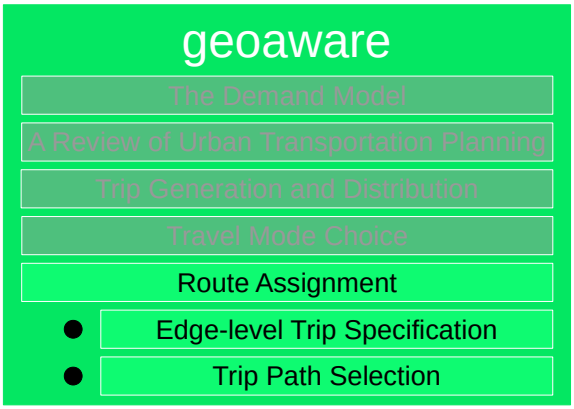
The **geoaware** component does not explicitly model travel mode choice as such information is not relevant to the macro- and micro-level inference procedures used to construct trip demand from reference data. Nevertheless, multiple, time-dependent travel modes may be indirectly modeled using the built-in mobility-record type support provided by the framework (see section 3.2.7).

In this approach, a trip demand model for each mode is constructed from reference

trajectories. Next, trip realizations are drawn from each trip demand model (see section 3.2.6) and then merged together using the built-in mobility-record type support provided by the framework (see section 3.2.7). The result is a multimodal realization that not only models the relative frequency of each travel mode but also preserves the temporal mobility patterns (as inferred from the trajectory data) associated with each modality. This approach is advantageous as the underlying macro- and micro-level procedures used to infer trip demands are agnostic to the data generating procedure, imposing minimal modeling assumptions and lessening the traditional subjectivity induced on the travel mode model by omitted explanatory variables (see [46] for a discussion of this in relation to the built environment) [53].

Despite the power of this approach, it is important to recall that a core assumption of the framework is that the provided trajectories must be considered representative. Certainly, trying to infer patterns of a population from an unrepresentative sample is inexpedient. Also, it should be noted that the GeoAware package exclusively focuses on vehicular travel modes as we rely on SUMO for synthetics generation.

3.2.5 Route Assignment



Once the generated trip demands have been assigned to a transportation mode (if nec-

essary), the next step is to disaggregate the demands into actual trips which can be individually routed through the network. The route taken is known as a path and is simply a navigable sequence of edges through the network that fulfills the onset and terminus specifications of the trip. The process by which such paths are determined and assigned is known as **route assignment** and constitutes the final step of the traditional UTP procedure.

In traditional transportation forecasting models, this step is done so as to obtain the induced effects of supply-demand interactions, such as edge flow and congestion, on travel times through the network [54] [206]. In the present work, we seek the simulated vehicles which induce such network properties rather than just the effects themselves[24].

In the **geoaware** component, route assignment occurs in two steps. In the first step, the block-level trip demands are converted into edge-level trip specifications by assimilating the inferred demands, departure time profiles and edge weights into a collection of trip specifications that minimally detail the origin edge, destination edge, and the departure time. Then, a route assignment procedure is specified which finds a navigable path through the network to fulfill each of the specified trips. We begin by looking at the specification of edge-level trips.

Edge-level Trip Specification

Before synthetic vehicles can be simulated by the **libsumor** component, it is necessary to translate the inferred macro- and micro-level quantities into edge-level trip specifications. This process can be thought of as constructing a vectored event signal where each event indicates the arrival of an agent into the simulation and is minimally detailed as `<id>, <depart.time>, <from>, <to>`. The `<depart.time>` details when the agent should depart from the edge specified in `<from>`. The edge specified in `<to>` indicates the edge an agent should end at. The actual path taken between the `<from>` and `<to>` locations is handled by a route assignment procedure which we detail shortly. Constructing such a signal from the inferred quantities is trivial but before we detail such a procedure we

[24] As we will explain in the forthcoming **libsumor** section (section 3.3), this necessarily requires that we utilize a microscopic traffic simulation but we leave the formal justification for that section.

review the creation of simulation intervals as the edge-level trip specifications are generated with respect to such intervals.

Recall from the trip generation and distribution steps (see section 3.2.3) that each block contains a demand signal which details the number of agents to generate over each fit interval. As was noted earlier, often the fit intervals used when fitting the SDSBM are not at the resolution desired for simulation and therefore the fit interval may be subdivided according to a simulation granularity parameter. Such subdivisions of the original fit interval are referred to as *simulation* intervals. Just as the SDSBM fit intervals were seasonally related so are the simulation intervals. This means that each trip belongs to (i) a particular seasonal period and (ii) a particular simulation interval within that period. For instance, given a month of data fit using a SDSBM with daily counts, weekly seasonality and hence a seasonal period of a day, each record would fall into one of the seven days of the week (e.g. Sunday, Monday, Tuesday, etc.) and thereby influence the resulting SDSBM fit based upon which day (seasonal period) a record belonged to. Now, if in order to achieve the desired simulation granularity, we discretize each day into 24 bins of one hour each, then, when presented with a month of data, each record would fall into a seasonal period — the day of the week — and a simulation bin — the hour of the day.

We now define an iterative procedure for constructing such edge-level trip specifications. For each block[25], the quantity of trips for each simulation interval is obtained by multiplying the quantity of trips demanded over each fit interval by the inferred time profile that the fit interval belongs to. The affect of this is a set of pairings detailing the simulation interval and the quantity of trips to generate over the finer simulation interval. Trip departure times (the time when an agent enters the simulation and departs from its onset location) are then drawn for each simulation interval. The trip departure procedure natively supports uniform, random and exponentially distributed departures but can be extended to support other distributions. Next, onset and terminus edges are drawn for each of the depar-

[25]While the procedure is presented in a nested structure for readability, a linear algorithm is actually implemented based upon some indexing niceties.

ture times from the inferred edge weights based upon the seasonal period and simulation interval each departure time belongs to. Lastly, once the trips have been generated for each block, all the records are organized by departure time.

It is important to understand what is being done through this trip generation procedure. By constructing the trips in this manner, we are calibrating the route assignment step of the UTP procedure by intelligently influencing the quantity of agents we expect to see within each block for a given time period and selecting the specific edges according to the edge popularity weights. Another benefit of generating the trips in this manner is that it allows us to re-generate trips according to the same demand model without having to re-perform the inference procedures. This allows us to experiment with different time and edge draws as well as change the parameters of the simulation.

Trip Path Selection

At this point, the trip specifications are not considered a complete assignment as there is no path (a listing of edges from the onset to terminus) detailed for each of the trips. In order to route agents through a network and collect the resulting location trace, a path must be assigned to each of the specified trips. We present such a path assignment procedure next.

The goal of route assignment is to choose an optimal[26] path through the network that satisfies the onset and terminus requirements of the trip subject to the network supply constraints and supply-demand interactions [15]. The premise behind route assignment lies in the assumption that path selection is influenced by various costs which a traveler wishes to minimize [54]. Arguably, one of the most influential costs is that of travel time, but other costs such as transportation cost (fare), parking fees and toll roads may also contribute to the route chosen[27] [54].

Unfortunately, finding such optimal paths is non-trivial as many costs, such as travel time, are dependent upon how many other individuals are traveling; making it is neces-

[26] We emphasize that the definition of optimal can take on different meanings.

[27] Indeed, as there are many such costs and each individual likely weights such costs somewhat differently, the selection of a finite set of factors is necessarily an approximation of the real. Please see [1] for an analysis of commuters' route choice behavior.

sary to consider dependence between trips [54]. This necessarily induces a time dependency on optimality which is easily substantiated through experience. For instance, commuters choosing their preferred route through a roadway network consider a variety of time-varying properties that influence optimal path selection, such as path travel time variability [1], variable speed limits [88], or time-of-day based tolls [52].

A myriad of analytical and simulation-based approaches exist for route assignment but all can be generally categorized as either static or dynamic depending upon how the time dependency of network supply-demand interactions is modeled within the approach. Due to the significant influence of congestion on travel times which is, as noted above, often the primary source of cost information when performing route assignment [1] [206], much of the distinction between static and dynamic approaches centers around the handling of congestion [54].

Static assignment approaches assume that time has no influence over the network supply-demand quantities for the period under study (hence the name static). As such, the quantities of interest governing network supply-demand interactions are time invariant and represent average behavior [54]. For instance, congestion in traditional static approaches is described by a volume-to-capacity ratio where the edges of the network and the paths assigned are assumed to be time independent [146] [54]. While historically prevalent [206] [39], applicable in large-scale situations [54], and able to converge to a single equilibrium [196] [54], static approaches suffer from a lack of accurate congestion modeling as the volume-to-capacity ratio used neglects queuing effects [146]. Rather than queuing agents behind the bottleneck and reducing the rate to that of the bottleneck capacity, no queuing occurs and the rate simply continues to grow [146]. Obviously, such a situation does not reflect reality and leads to misrepresenting congestion which results in inaccurate travel time estimates [80] [146].

Dynamic traffic assignment (DTA) approaches are based on the exact opposite assumption of static approaches by arguing that traffic networks are rarely in a steady-state

for the entire modeling duration and therefore insist that time should influence network supply-demand quantities [54]. To capture the temporal aspect, DTA departs from the volume-to-capacity ratio of static approaches and models edges dynamically using a mathematical function known as the *fundamental diagram (FD) of traffic flow* [54]. Each edge is associated with a FD (possibly unique [54]) that relates changes in edge density k to edge velocity v, thereby allowing varying velocity estimates for an edge as the density of that edge changes with time [54]. So long as the inflow is less than or equal to the outflow $q_{in} \leq q_{out}$, congestion does not occur. But when the inflow becomes greater than the outflow $q_{in} > q_{out}$, congestion occurs, bringing a reduction in edge velocity and the queuing of vehicles since there are more vehicles entering than can be released. When the queue reaches the edge entrance, the inflow can no longer be greater than the outflow since the edge is at capacity and therefore must be immediately reduced to that of the outflow [54] [194]. When this happens, the congestion now spills-back onto the incoming edges, at which point the above process repeats itself on those edges.

Introducing the ability for the velocity of an edge to change with time results in two notable improvements over static approaches. First, the travel time estimates can now be more accurately estimated as the velocity estimates upon which travel times are based are dependent upon the density of the edge when traveled (i.e. dynamic) not some average value. Second, unlike the volume-to-capacity ratios used in static approaches, the velocity-vs-density curve of the FD has a physical meaning that enables its utility in real-world scenarios.

The goal of the **geoaware** component is to create the necessary demand model for generating realistic synthetic vehicles *not* to define a new route assignment methodology. For this reason, the exact route assignment approach chosen is irrelevant so long as the approach is able to assign paths through the network to each of the previously generated stochastic trips. Nevertheless, simulation-based DTA approaches are recommended due to: (i) the enhanced traffic realism brought by simulation; (ii) the native support for dynamic

modeling of network supply-demand quantities; (iii) the ability to handle the commonly occurring, non-FIFO (First In, First Out) behavior of vehicle overtaking[28] [54]; (iv) the ability to handle non analytical behavior [169]; (v) the ability to use the simulation framework itself for computing paths between points thereby avoiding the additional overhead of an independent traffic router [169], and (vi) the focus on deployable solutions [169]. The SUMO framework provides a number of route assignment algorithms which can be used to assign paths to the specified trips. We review two of the most prominent examples.

SUMO implements the stochastic user-equilibrium (SUE) approach given in [79] where each driver is assigned an origin, destination and departure time and chooses a path through the network from a restricted path set according to a probability distribution [123]. Initially, the path set contains only the shortest path through the empty network, but as network conditions change (through the iterative assignment procedure), new shortest paths through the network are computed and added [123]. Each driver is assigned to one of the paths in its path set according to the route selection probabilities which are continually updated to favor lower cost paths [79]. The procedure can be configured to terminate after a fixed number of iterations or based upon convergence to a desired average travel time[29]. As the approach is simulation-based, a microscopic simulation is used to calculate the travel times for each of the driver's chosen routes.

SUMO also supports an alternative approach to SUE, termed one-shot assignment [124]. One-shot assignment is advantageous when an analyst wishes to model trip behavior based upon instantaneous travel times (at the instant of departure) or when the time required to converge to an approximate equilibrium solution is not available; a situation which can readily occur in networks of reasonable size [124]. There are three variations to this approach which vary in the amount of feedback utilized during assignment [124]. The first approach, incorporates no feedback of network conditions and simply assigns paths to

[28] Overtaking allows a vehicle that enters an edge after another vehicle to be able to *overtake* the earlier vehicle by passing it and therefore exit the edge before the earlier vehicle. While not all DTA approaches support overtaking, such a concept is completely foreign to FIFO enforcing static approaches [54].

[29] https://sumo.dlr.de/docs/Demand/Dynamic_User_Assignment.html

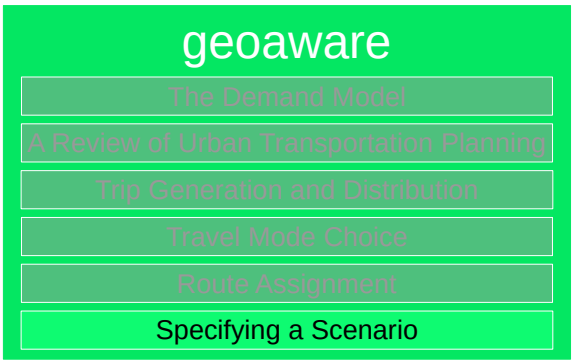
origin-destination pairs from an appropriate set of fixed paths [54]. The second approach regularly updates a set of shortest routes between the origin-destination pairs and assigns the current shortest (or minimum cost) route between the origin-destination pair to the trip at the time of departure [54]. The last approach incorporates the most feedback and in addition to regularly updating and assigning the shortest route to new trips, also periodically updates the routes of en-route vehicles based upon the current network conditions [54]. In SUMO, edge weights are updated every so often (a configurable parameter) based upon current network conditions. Rather than being simply the travel time of the edge in the last simulation period, the edge weights are instead smoothed using either a moving or exponential average[30].

In addition to the SUMO route assignment algorithms, if the reference trajectories are at trace-level, the **geoaware** component also supports the ability to replicate routes taken by agents in the reference trajectories. This process works by running a map matching algorithm on the reference trajectories and associating the path taken with the departure time of the trip. Then, when the trips are being specified, a certain percentage of trips in each simulation interval are generated which have identical paths to those in the reference trajectories during the same time frame. The percentage of trips is controllable by an analyst and allows a user to specify the percentage of trips for each simulation interval in each period of a seasonal cycle. For these trips, the route is already specified and does not need to be further computed. We use the Fast Map Matching algorithm proposed by [231] but any map matching algorithm should be compatible with proper extension.

3.2.6 Specifying a Scenario

The trips specified within the route assignment step (see section 3.2.5) are a stochastically generated realization drawn from the constructed demand model. When the realizations

[30]Please see the help documentation at https://sumo.dlr.de/docs/Demand/Automatic_Routing.html and [123] for additional details regarding the edge weighting algorithms.



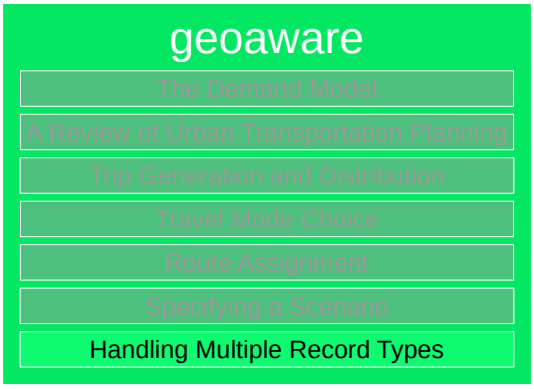
are coupled with a network and simulation settings we refer to the collection as a *scenario* since it represents a configuration of a particular model instance. The scenario is detailed using the `libsumor::Scenario` construct from the **libsumor** component and requires specifying the (i) roadway network, (ii) trip specifications, and (iii) configuration file.

The roadway network specifies the SUMO-compatible map that establishes the infrastructure of the environment. This map is one of the assumed inputs (see section 3.1.2). The trip specifications intelligently calibrate the simulation demand by specifying the onset and terminus locations according to the inferred mobility patterns. As previously detailed, such trips are specified through the macro- and micro-level procedure detailed in sections 3.2.3 — 3.2.5. The YAML-based configuration file allows an analyst to configure the behavior of the **libsumor** component. Just like the map, the configuration file and associated parameters are assumed as *a priori* input.

Containerizing the model realization and simulation settings in this way supports modularity and reusability, enabling the exchange of scenario components as application or experimentation dictates. For instance, under this methodology, one can easily perform Monte-Carlo style simulations by simply re-drawing the trips component of the scenario.

3.2.7 Handling Multiple Record Types

At times, it is useful to differentiate mobility-records according to some type identifier so that properties can be set independent of other mobility-record types. One instance of this



was presented when discussing travel mode choice. Supporting various travel modes however, is not the only use of type identifiers. Consider, for example, being able to adjust the vehicle model used within the simulation based upon the time of day or coloring GUI visualizations based upon destination type. In the GeoAware framework, the trip realizations produced during the route assignment step are assumed to belong to a single type; thus, it is necessary to define an assimilation procedure to support multiple type identifiers.

The assimilation procedure requires that the mobility-records have been split by type and a trip demand model for each type has been fit using the relevant data (see sections 3.2.3 — 3.2.5 for details on this procedure). Next, trip realizations must be drawn from each trip demand model. This step ensures that the mobility patterns captured for a particular type are preserved. The trips are then merged together and organized by departure time. This produces a collection of trips with type identifiers where the relative frequency of each trip type comes from the frequency of that trip type in the original mobility-records. The final step is to produce a multi-typed scenario by combining the merged trips with a roadway network model and a configuration file that supports multiple vehicle types.

3.3 The `libsumor` Component

The concluding step in our goal to obtain synthetic vehicle traces is to capture the movement of the agents specified in the generated scenario (see section 3.2.6). The scenario specified during the last step of the `geoaware` component is a realization that links the supply constraints of the roadway network (e.g. lane counts, junction locations, traffic light timing, maximum speeds, etc.) with the trips generated from the macro- and micro-level inference procedures.

The desire to obtain the traces of vehicles as they propagate through the network subject to the supply-demand interactions necessitates the use of microscopic simulation [70] [140]. Transportation simulation is a broad field that has proved useful for modeling traffic environments because of its ability to capture the rich set of interactions between agents which is not possible in analytical formulations [70] [169] [42] [99] [167] [20] [140] [133] [61] [72] [21] [191] [154].

Computer simulation is an alternative to analytic evaluation that imitates a system through an algorithmic process on a system model [19] [95] [65] [70]. Microscopic simulations [72] [140] [205] [96] differentiate themselves from other types of transportation simulations by modeling agent movement at the individual level rather than at an aggregate level, as in macroscopic simulations [160] [100] [96], or a hybrid level, as in mesoscopic simulations [220] [191] [62] [133] [61] [130].

In microscopic simulations, the individual agents are simulated using car-following, lane-changing and gap-acceptance theories; fundamental quantities in traffic theory [7] [70]. Modeling traffic in this individual manner means that important travel quantities, such as vehicle velocity, are tied to specific agents not the result of some aggregate computation [42]. Of course, the additional detail level comes at a cost, often resulting in longer running times and increased storage requirements for microscopic-based transportation studies [7][124].

Having justified the need for a microscopic simulator when producing the desired syn-

thetics, the next step is to define a microscopic simulation framework capable of producing the synthetics we seek. Certainly, one approach would be to create a custom simulator, however, such a framework would likely be highly specific to a particular situation and would necessarily require redoing basic simulation components that have already been designed by others [123]. Indeed, as [123] points out in his work, often the defining of a traffic simulation is tangential to the objective of the work. As our goal is not to construct a new traffic simulator, we integrate our work with the SUMO microscopic traffic simulator [140] due to its extensive suite of features, its open-source nature, and its ability to be extended [123]. In addition to preventing re-implementations of core simulation components, this modularity also helps to divorce the objective of the work from the actual implementation, thereby increasing the utility of the proposed work.

With the simulation framework chosen, the next step is to "wrangle" the simulation scenario produced by the **geoaware** component into a format suitable for simulation by the SUMO framework so that synthetics may be produced. While an ad-hoc procedure could be defined to create the desired synthetics, such a procedure would share many tasks which are foundational to any traffic simulation irrespective of the modeling approach chosen. Therefore, rather than produce a highly specific procedure that will only work with demand produced by the **geoaware** component, we create a general, accessible and extensible framework that allows an analyst to craft and run arbitrary traffic simulations.

Due to the widespread applicability of synthetic trajectory generators (see chapter 2 for a discussion), it is important that such generators are analyst accessible. While our use of a microscopic simulator enables us to produce realistic synthetics, it also poses a challenge to accessibility. Each microscopic traffic simulator is unique and has a myriad of configurable options. Expecting that a user knows or wants to calibrate such parameters when his/her primary goal is just to obtain synthetics is questionable. Instead, it would be preferable if a user could utilize such a generator through a common research platform with which he/she is already familiar. Such an approach would directly increase accessibil-

ity, prevent re-implementations and reduce educational overhead. One commonly[31] used research platform is the R programming environment [178].

R provides a feature-rich ecosystem for data analysis, statistical modeling, machine learning, visualization, and communication of results [222]. The R programming environment is particularly well suited for data wrangling and provides an excellent platform for rapid development due to its functional (interpreted) nature. R also supports reproducible research through a variety of constructs (such as R notebooks[32], R markdown[33], or shiny[34]). Many features of R are considered invaluable to the future of transportation modeling and smart cities [150] [176] [122].

Despite the prevalence of R for spatial data analysis [141] [34] and R packages for various steps in transportation modeling[35], a formal package enabling native microscopic traffic simulation from within the R programming environemnt does not exist[36]. The **libsumor** component of the GeoAware framework is our endeavor to provide a data-driven, analyst friendly contribution to this lacking area of the R programming ecosystem. Using R and the proposed **libsumor** component, a transportation professional can easily visualize, model and experiment with transportation data; something not as easily accomplished in

[31] While quantifying the exact popularity of a language is difficult, recent language rankings, articles, and conferences hint at usage statistics. The TIOBE Index (https://www.tiobe.com/tiobe-index, accessed 10/8/20) places R at 9th place for October 2020. The 2020 IEEE Spectrum language ranking (https://spectrum.ieee.org/static/interactive-the-top-programming-languages-2020, accessed 8/7/20) places R as the sixth most popular language. The PYPL Index (based upon language tutorial searches) indicates a somewhat steady share of users in recent years (http://pypl.github.io/PYPL.html, accessed 10/8/20). Several recent articles share a similar view (see *Programming language rankings: R makes a comeback but there's debate about its rise*, https://www.zdnet.com/article/programming-language-rankings-r-makes-a-comeback-but-theres-debate-about-its-rise by Liam Tung, accessed 10/8/20 or *Python, R, Other Programming Languages Thriving Long-Term*, https://insights.dice.com/2020/08/04/python-r-programming-languages-thriving-long-term/ by Nick Kolakowski, accessed 10/8/20). R also features an active development community including local groups (https://www.meetup.com/topics/r-project-for-statistical-computing/, accessed 10/8/20) and conferences (useR conference: https://user2020.r-project.org, accessed 10/8/20; rstudio::conf: https://rstudio.com/conference/, accessed 10/8/20).

[32] https://rmarkdown.rstudio.com/lesson-10.html.

[33] https://rmarkdown.rstudio.com/lesson-1.html

[34] https://shiny.rstudio.com

[35] Examples include: `osrm`: https://github.com/rCarto/osrm, `stplanr`: https://github.com/ropensci/stplanr, `leaflet`: https://rstudio.github.io/leaflet, `igraph`: http://igraph.org, and `travelR`: http://travelr.r-forge.r-project.org

[36] To be sure, simulations can be constructed in R and some simulation packages exist for R. Prominent examples include `simmer` (https://r-simmer.org) and `simulator` (http://github.com/jacobbien/simulator). The CUBE Voyager traffic simulator even facilitates analysis in R. Nevertheless, none are specifically structured for microscopic traffic simulation.

languages such as `C`, `C++`, or `Java`.

As the GeoAware framework relies on the SUMO traffic simulator to perform the low-level simulation, the design of the **libsumor** component was largely an architectural effort focused on the construction of a generic, analyst accessible framework for microscopic transportation simulations within the `R` programming environment [178]. Guided by the general requirements of a simulation scenario, we present a modular framework which is able to handle general simulation scenarios and efficiently produce synthetic vehicle traces. The modular decomposition of the **libsumor** component resulted in three complimentary approaches to simulation which differ in the degree of control granted to the analyst. In all approaches, the SUMO microscopic simulation engine is used but each varies in its intended use and thus the features that are exposed to the analyst for control.

3.3.1 Mode: XML Intermediaries

One approach an analyst can take to perform microscopic traffic simulations from within `R` is to utilize the *XML Intermediaries* mode of the **libsumor** component. This approach configures the simulation using a collection of XML files (hence the name) natively supported by SUMO and allows the simulation to be ran using the traditional `sumo` or `sumo-gui` utilities. The XML files (i) configure SUMO, (ii) detail the trips to simulate, and (iii) identify vehicle types (if applicable). `R`-based wrapper functions pass the SUMO configuration file to the `sumo` and `sumo-gui` utilities, allowing a user to run a simulation from within the `R` environment.

The configuration file is built according to the SUMO configuration file definition[37] using the relevant portions of the YAML-based configuration file provided at input. The trip definitions are derived from the trip portion of the scenario and are specified according to the SUMO trip and vehicle definitions[38]. The `libsumor::Trips` mod-

[37]SUMO Configuration File Definition: https://sumo.dlr.de/docs/Basics/Using_the_Command_Line_Applications.html#configuration_files

[38]Routes/Trips Definitions: https://sumo.dlr.de/docs/Definition_of_Vehicles,_Vehicle_Types,_and_Routes.html

ule provides a method for producing such a file from a collection of trip specifications. The trip specifications produced by the **geoaware** component are already formatted as a `libsumor::Trips` object, however, any arbitrary demand can be appropriately formatted (see section 4.1.2 for an algorithm). The vehicle types file is only necessary if vehicle types are supplied (see section 3.2.7) and is written out using a SUMO additional file[39].

The benefits of the *XML Intermediaries* approach include its simplicity, ease-of-use and low educational requirements. The approach is also the only approach provided by the **libsumor** component that supports visualization. Additionally, because it uses the `sumo` and `sumo-gui` executables, the *XML Intermediaries* approach does not require a source distribution of SUMO.

As with any approach though, there are some drawbacks. The most limiting drawback of the *XML Intermediaries* approach is that the configuration of SUMO is limited. While SUMO provides a fairly expressive set of configuration options, the basic structure of the simulation cannot be changed when using the `sumo` or `sumo-gui` utilities. Additionally, if the production of synthetics is the ultimate goal, the resulting output will have to be further processed. In a large simulation, such processing may require substantial effort. Finally, the *XML Intermediaries* approach does not provide in-memory access to the generated traces. This means that such information is not queryable from the `R` environment. Such a hurdle prevents an analyst from being able to quickly use the synthetics produced.

3.3.2 Mode: Synthetics Generation

The second approach provided by the **libsumor** component for performing microscopic simulations focuses exclusively on generating a portable collection of synthetic vehicle traces and is referred to as the *synthetics* mode of the **libsumor** component. The pro-

[39] SUMO Additional File Definition: https://sumo.dlr.de/docs/Simulation/Basic_Definition.html#additional_files (accessed10/9/20); Type Definition: https://sumo.dlr.de/docs/Definition_of_Vehicles,_Vehicle_Types,_and_Routes.html (accessed 10/9/20)

cedure is C++-based and makes use of SUMO's Libsumo[40] package for controlling the SUMO simulation and collecting the traces. The Libsumo component is provided with the source distribution of SUMO and provides low-level access to a running SUMO simulation (similar to the TraCI interface but without the networking overhead) via a C++ library. The C++-based synthetics module is made accessible to the R programming environment through the Rcpp package; an extension to R that allows it exchange data with dynamically loaded C++ modules [69] [67] [68].

Just like the *XML intermediaries* approach, the process begins by constructing XML files detailing the scenario configuration, trip specifications and vehicle types (if needed). Then, the configuration file is provided to a custom procedure which uses the Libsumo package to query the various files. The custom procedure tracks the positions of all the vehicles in the simulation and logs important metrics, such as the speed, angle or total distance. As detailed in section 3.1.5, the *synthetics* mode can be configured to filter the records retained by both edge and time. Additionally, the analyst can store the coordinates in longitude/latitude mode as well. The records can be stored in memory for easy access from R or dumped to a CSV file with variable levels of output and floating-point precision.

The most notable benefit of the *synthetics* approach is that it is specifically designed for producing synthetics. As a result, the instantiation of various modules, their interconnection and the parsing of data is internally handled so that a user need only focus on generating synthetic trajectories. Just like the *XML Intermediaries* approach, the *synthetics* approach relies on a (i) SUMO configuration file, (ii) trip specifications, and (iii) type specifications (if applicable). (Once again, the *geoaware* component produces properly formatted trip specifications but any demand can be properly formatted — see section 4.1.2 for an algorithm.) Additionally, due to the use of C++, the approach benefits from the efficiency of a compiled language. Another key benefit of the *synthetics* approach is that it provides in-memory access to the produced synthetics; making further computation and

[40]Documentation: https://sumo.dlr.de/docs/Libsumo.html

analysis much more efficient. Such in-memory access allows an analyst to easily run a traffic simulation and interactively investigate the resulting traces without having to store the information in an intermediary format.

Just like the *XML Intermediaries* approach, the most limiting drawback is that configuration is limited. The approach supports any SUMO configuration parameter in addition to a set of parameters controlling the filtering and output of the approach, but once again, fine-grained control over the underlying simulation is not possible. It should also be noted that the ability to store traces in memory is dependent upon the amount of available memory. For very large simulations, dumping the records to a CSV is preferable due to memory limitations.

3.3.3 Mode: Wrapper

The final approach provided by the **libsumor** component provides an analyst with the most flexible and low-level approach to performing traffic simulations from within the `R` programming environment. The approach is referred to as the *wrapper* mode of the **libsumor** component since it provides a set of wrapper functions around the `Libsumo` package. Using the `Libsumo` library for facilitating the communication between `R` and SUMO, the *wrapper* mode supports access to and modification of (where available) a myriad of settings associated with the simulation, edges, junctions, induction loops, vehicles and other such aspects.

The *wrapper* mode organizes the function calls of the `Libsumo` package to follow the nomenclature of the `R` ecosystem and uses `R6`[41] classes and the `Rcpp` package to generate wrapper functions around their `Libsumo` counterparts. The *wrapper* mode extends access to the SUMO framework to the `R` platform and is similar in premise to that of the Python and Java counterparts provided with the source distribution of SUMO. As this is an academic work, coverage of the `Libsumo` component is incomplete and instead our work

[41] R6 GitHub Page: https://github.com/r-lib/R6

focuses on establishing the necessary communication between `R` and `Libsumo` so as to support our forthcoming evaluation.

The most notable benefit of the `wrapper` mode is that it provides fine-grained, programmatic control over the simulation steps. This increase in control allows an analyst to construct and query simulations as needed without having to post-process results. Because the approach is `R`-based, the result is the ability to interactively configure, run and and query a traffic simulation.

Obviously, the increased control results in a greater educational burden for the user. The instantiation, running and output of a simulation is no longer automatically handled and must be explicitly configured by the user. This requires the user to have knowledge about traffic simulations and how to interconnect the various components. Additionally, while the ability to perform vehicle simulations from within the `R` environment greatly increases accessibility, the accessibility comes at a cost to performance. Because `R` is interpreted, its performance is not that of a compiled language. Nevertheless, while there is a performance hit for using `R`, we believe its interactive and self-documenting nature are benefits which are well worth the performance hit.

Chapter 4 — Evaluation

The goal of the proceeding methodology chapter was to present a pipeline by which a traffic simulation may be constructed and certain simulation demand metrics calibrated. In this chapter, our aim is to validate the various components of the pipeline by demonstrating that they achieve their respective design goals. In contrast to an application specific work, evaluating a general purpose framework such as the one proposed is tricky as it requires carefully enumerating goals common to many situations rather than merely evaluating performance within a specific scenario. To complicate analysis further, general purpose frameworks often proport many intangible benefits, such as ease of use, modularity or extensibility, which are hard to evaluate in any measurable sense. Therefore, in light of this, we seek to validate generic functionality of the pipeline that is relevant across situations instead of crafting and analyzing performance on, what would be, an application specific analysis. Because our framework features both tangible and intangible benefits, we evaluate the framework from both qualitative and quantitative viewpoints.

In the qualitative evaluation, we focus on evaluating the utility of the proposed framework for creating data-driven vehicular traffic simulations. The evaluation is a pragmatic one and is focused on assessing the framework's suite of complexity managing, general purpose tools for constructing simulations. Such tools are provided through the **libsumor** component and our evaluation of that component allows us to assess our goal of providing an accessible software package for data-driven traffic simulations. Because the **libsumor** component of the GeoAware framework is designed as a general purpose software package

for traffic simulations, the analysis presented in 4.1 focuses exclusively on the procedure required to turn traffic demand, whatever form it may take, into a simulation. Our evaluation demonstrates the ability to incorporate archetypical demand sources represented as origin/destination (OD) matrices. Such a qualitative evaluation is well suited for this type of component as a user of a software package is often less concerned about understanding the mundane inner-workings of the library and is more concerned with understanding how to use the package in their particular situation.

In contrast to the qualitative evaluation which focused on analyzing the capability of the framework for generating traffic simulations, the quantitative evaluation seeks to demonstrate that simulation demand calibrated using the **geoaware** component captures behavioral patterns in the reference trajectories. Whereas the general purpose nature of the **libsumor** component warranted a qualitative discussion, the **geoaware** component is specifically crafted to assist in calibrating the demand of a simulation and therefore necessitates a quantitative evaluation to demonstrate results. The traditional approach would be to compare the proposed calibration procedure to other techniques but we believe such a rank-comparison is ill-advised. As calibration procedures are extremely varied, a worthwhile rank comparison would be extremely difficult to articulate as each calibration approach may optimize different elements. Additionally, as the use of trajectory data in calibrating traffic simulations is not commonplace in practice or the literature, evaluating the proposed framework against typically employed calibration techniques would result in an "apples to oranges"-like comparison with results which should never be considered similar in the first place.

Instead, in a vein similar to that of [40], we believe a much more informative evaluation centers around demonstrating the ability of the **geoaware** component to calibrate several simulation demand quantities. We present four different quantities of simulation demand which the **geoaware** component calibrates. For each metric, we demonstrate how the framework is able to infer such quantities through simulation experiments. Then, we

provide a brief review of other literature pieces or simulation frameworks which also calibrate the same (or similar) quantity. Finally, for each quantity, we present a comparison of our approach to the previously reviewed techniques. By focusing on the individual demand quantities rather than on the final, resulting simulation we are able to compare our approach to any calibration procedure by examining how each approach calibrates (or does not calibrate) that quantity. Such a quantitative evaluation is presented in section 4.2.

Lastly, in an effort to be complete and demonstrate how the complete framework can be used to generate synthetic trajectories from reference data, we present a use case involving the taxicab data from the city of Chicago, Illinois. In this use case, we demonstrate how to turn publicly available taxicab data that logs the spatiotemporal pickup and dropoff regions of taxi trips into a collection of synthetic trajectories using the GeoAware framework proposed. This use case is presented in section 4.3.

4.1 Qualitative

In our discussion of related works (see chapter 2), we motivated the need for traffic simulations by citing their applicability in a myriad of applications, including the design and evaluation of spatiotemporal algorithms [171] [237] [49] [106] [6], privacy protection [94] [126], and testing the performance of spatial database operations [171] [64] [40] [56] [159]. As many of these applications are heavily reliant on data, in the methodology chapter (see chapter 3) we argued for a data-driven simulation framework that was readily accessible from a commonly used research platform and set about creating a first of its kind framework for the `R` programming environment. The **libsumor** component is the result and it allows an analyst to communicate with the popular and open source SUMO traffic framework.

In the forthcoming sections, we evaluate the benefits of the **libsumor** component and discuss future work that needs to be performed on the component. In section 4.1.1, we

begin by introducing the **libsumor** API and explaining the general purpose functionality provided to users of the package. Next, in section 4.1.2 we highlight how commonly used sources of trip demand data can be preprocessed (or used directly) when performing a simulation with the **libsumor** component. Finally, in the last section (section 4.1.3), we note areas of the **libsumor** component which need improvement and discuss the trip-based approach to simulation.

4.1.1 The libsumor API

The **libsumor** data model is structured around the concept of a simulation scenario which details the components of a simulation by specifying the (i) trip demand, (ii) roadway network, and (iii) configuration file for a particular situation. Constructing such scenarios is an inherently data-driven task and therefore the data-science friendly `R` programming language was chosen since it provides a rich suite of tools for efficient data manipulation and management (for a more thorough justification behind the use of the `R` programming environment, please refer to section 3.3). The general approach to constructing such a scenario is outlined in algorithm[1] 1.

input : (i) trip demand, (ii) a roadway network, and (iii) a configuration file
output: a simulation scenario (`libsumor::Scenario`)

```
# the inputs
demand ← a properly formatted trip demand
network ← a SUMO-compatible roadway network
config ← a configuration file (YML)

# generate scenario
scenario ← generate.scenario(demand, network, config)
```

Algorithm 1: Constructing a simulation scenario

It is assumed that the network and configuration settings are known to the analyst.

[1] While the discussion and code examples are presented in an `R`-like nomenclature, no knowledge of `R` is assumed.

Generating or obtaining such network and configuration data is presented in the methodology chapter in sections 3.1.2 and 3.1.5, respectively. The traffic demand can be obtained from a number of sources and is typically the most involved step in constructing a simulation scenario. Because of this, we outline a general algorithm for producing properly formatted traffic demand in the forthcoming section (section 4.1.2). In the context of the GeoAware framework, such traffic demand is constructed using the `geoaware` component.

Once a scenario has been constructed, it is passed to the `libsumor` component for simulation along with a character string indicating the operation mode. As outlined in section 3.3, the **libsumor** component may be operated in one of three different modes. Each mode is designed for a particular use case in order to give the analyst a fair amount of control when performing traffic simulations. The three operation modes are: (i) *XML Intermediaries*, (ii) *synthetics*, and (iii) *wrapper*. For details regarding each approach, please see section 3.3. Regardless of the operation mode chosen, an `R`-based connection to the SUMO traffic simulator can be obtained using the procedure detailed in algorithm 2.

input : (i) a simulation scenario and (ii) a string indicating the simulation mode
output: a connection to the SUMO traffic simulator

```
1 # the inputs
2 scenario ← a properly constructed simulation scenario (see algo. 1)
3 mode ← a string indicating the simulation mode

4 # generate scenario
5 simulation ← libsumor(scenario, mode)
```

Algorithm 2: Performing a simulation

The *XML Intermediaries* mode operates using SUMO's native XML support and can be configured as detailed in section 3.1.5.

The *synthetics* mode is specifically designed for producing synthetic agents from a traffic simulation and is therefore the ideal operating mode when performing synthetics

Method	Arguments	Description
reload		Reloads the connection to the SUMO traffic simulator thereby resetting the simulation
run		Runs the simulation and produces synthetics up to time end
	end	A numeric indicating the endpoint of the simulation
	$status\_freq$	An integer indicating how often progress updates should be printed out
getAgentRecord		Returns the filtered record for the agent specified by id
	id	A string indicating the agent id to obtain the record for
getAgentRecords		Returns the filtered records for all the agents who have completed their trips
writeCSV		Writes the filtered record for agent id out to the file path detailed by fp
	id	A string indicating the agent record to write out
	fp	A string indicating where the CSV file should be written
	$mode$	A binary vector codifying the properties of the agent to write out
	$precision$	An integer detailing the precision of the doubles written out

Table 4.1: Available methods of the **libsumor** component while configured in *synthetics* mode

generation. The API available while configured in *synthetics* mode is detailed in table 4.1 along with the method objective.

Configuring the **libsumor** component in the *wrapper* mode enables an analyst to completely[2] configure, run and query a simulation entirely from `R` and, unlike the previous two approaches, does not require that a simulation scenario be defined. The *wrapper* mode enables low-level use of the SUMO traffic simulator in `R` and allows an analyst to query a running traffic simulation in an interactive, REPL[3]-like fashion.

As traffic simulations are naturally data-driven, often deriving demand from in-field

[2]Recall, as this is an academic work, coverage of SUMO is not complete.
[3]read-eval-print-loop

measurements, an approach which fuses `R`'s data-driven workflow with a fast, `C++`-based traffic simulator permits rapid construction of traffic simulations, data analysis and prototyping capabilities. The `libsumor` component provides an API for simple, programmatic control of traffic simulations and reduces the amount of knowledge an analyst must know about the used traffic simulator when performing a traffic simulation or generating synthetics. The two main steps of this API are demonstrated in algorithms 1 and 2. Thus, as one can see, the provided API benefits traffic analysts by providing a suite of analyst accessible tools that help manage the construction and running of a simulation from a an easy to use programming environment.

4.1.2 Sources of Trip Demand

At its most fundamental level, travel is the movement from one location to another. When such travel is associated with a unique identifier, we call such movement a trip. A collection of such trips within a region is referred to as demand and such demand is often specific to a certain time interval. Obtaining accurate travel demand is critical to ensure that any conclusions drawn from an urban study are based on realistic conditions.

Despite its importance, travel demand is often unknown at worst and noisy at best. Therefore, modeling such demand is necessary based upon data sources deemed relevant to trip demand. Travel surveys, socioeconomic data, traffic counts, junction turning ratios, CDRs (call detail records) and GPS traces are the most commonly employed data sources for demand estimation. Travel surveys and censuses represent data sources that are derived from population samples and are typically used to estimate travel demand through regression analysis [206]. Traffic counts and junction turning ratios are aggregate measures of trip demand. Traffic count data is the most prevalent and is often used to generate a set of trips which will mirror the traffic counts observed [75]. CDRs and GPS traces are disaggregate sources that provide low-level mobility pattern information that is lacking in aggregate sources. To use disaggregate sources for modeling, the data is often aggregated as merely

replicating a trajectory dataset does not attempt to infer transferable characteristics of mobility.

Despite the various sources of data, the ultimate goal of all such data sources is to estimate (sometimes using multiple sources) the travel demand between locations. This data is typically depicted as a matrix where the rows and columns represent locations and the cell values indicate the quantity of trips between two such locations. This representation is commonly referred to as an origin destination (OD) matrix. An OD matrix is a general and extremely simple construct (in theory, not always to acquire) and is the standardized way of conveying demand information in traffic planning contexts. OD matrices are often an intermediary output produced at the end of the trip distribution phase of the UTP procedure and the locations are typically aggregated to zonal regions but in theory, such locations could be at any arbitrary resolution. OD matrices can also represent time dependent demand by associating a unique time period with each OD matrix.

The concept of an OD matrix, as presented above, is inherently aggregate in nature and therefore limited in its expressiveness. Using such a matrix format within the **libsumor** component would limit how easily low-level information could be associated with particular trips comprising the demand. Therefore, in order to provide greater flexibility when defining demand, the **libsumor** component uses a slightly different approach and represents demand as a collection of trip specifications where each trip minimally details a (i) trip id, (ii) departure time, (iii) onset edge, (iv) terminus edge, and (iv) vehicle type for each trip. Such a representation allows the framework to work in aggregate contexts where the time and edges might be drawn according to *a priori* distributions or in disaggregate situations where low-level information concerning departure time and edges are available. Additionally, because we only specify the minimum trip specification requirements, further information may be detailed for each trip if desired.

Due to the prevalence of OD matrices in traffic planning contexts, algorithm 3 details a process for converting an arbitrary, potentially time-dependent OD matrix to to the format

input : OD matrix (od)
output: a collection of trips (`libsumor::Trips`)

```
1  id ← 1
2  trip_specs ← a data frame storing properly formatted trips
3  for i ← 1 to nrow(od) do
4      for j ← 1 to ncol(od) do
5          n ← od[i, j]
6          for k ← 1 to n do
7              depart ← draw departure time for row-col block
8              onset ← draw onset edge from row TAZ
9              terminus ← draw terminus edge from col TAZ
10             type ← draw vehicle type for row-col block
11             append(trip_specs, id, depart, from, to, type)
12             id ← id + 1
13         end
14     end
15 end
```

Algorithm 3: Converting an OD matrix to the required trips format

used by **libsumor**. Through the utilization of such an algorithm, any general OD matrix can be arranged in a format usable by the **libsumor** component. As not all traffic simulations have the same requirements, the ability for the **libsumor** component to handle any arbitrary OD matrix greatly extends the applicability of the **libsumor** component.

The conversion procedure specified in algorithm 3 is not complicated and begins by instantiating a trip identifier and a storage container for the trip specifications. After this, for each cell in the original OD matrix, we obtain the specified number of trips to generate n and create individual trips by drawing the required information and appending the trips to the collection of trip specifications. Because not all data sources provide the same amount of information, certain pieces of required information may be unavailable. Obviously, obtaining the required information in a data-driven manner is preferable, but if such information is unavailable, the missing quantities may be chosen using prior knowledge (no matter how limited it might be) through the use of *a priori* distributions. If the OD matrix is time dependent, the above process can be repeated for each matrix and the resulting trip

specifications merged into a single collection of trips.

At first glance, it may seem that the minimum required trip information requires the fabrication of data which would be unnecessary if a different set of minimum trip features was used but in fact, if synthetic trips are to be generated, such required information constitutes the minimum required information as the simulator must know the departure time for the trip (i.e., when to insert the vehicle into the simulation) as well as the onset and terminus destinations, even if such quantities are chosen randomly[4].

4.1.3 Future Work

While we have demonstrated that the utility of the `libsumor` component (section 4.1.1) and its ability to simulate arbitrary OD matrices (section 4.1.2), it is equally important to highlight the framework's critical assumptions and areas of weakness.

One such assumption is the fact that the `libsumor` component assumes activity at the trip level and makes no attempt to chain trips or to model trip producing activities as is done in activity-based models. Obviously, ignoring such factors results in an abstraction of reality as common sense and research shows that such factors do influence trips [36], however, consideration of such factors may bring no appreciable benefit to the modeling at hand. For instance, when evaluating spatiotemporal algorithms (e.g. map matching or trajectory compression), a model of the trip producing activity or the chaining of trips is unlikely to have any meaningful affect on the evaluation. Instead, what is needed to evaluate such algorithms are tracks which faithfully mirror what we expect to observe and the exact modeling approach for obtaining such tracks is likely irrelevant.

Furthermore, trajectory data often does not come with any additional information which could be used to derive the information necessary for an activity model. Thus, while activity-based techniques for modeling mobility are certainly more prevalent than in times

[4]The specification of a vehicle type adds no further constraint and can be set to a default value if no information is available.

past, trip-based models are still the predominant model type in use today by transportation practitioners [38] [206]. Indeed, trip-based models have been used for many years now in a variety of contexts [206]. The desire to use trajectories and maintain a data-driven framework with minimal assumptions leads us to utilize the trip-based modeling approach[5].

Switching to an implementation standpoint, one drawback of the **libsumor** component is the fact that, with the exception of the *wrapper* mode, it requires that many of the simulation files be written to disk. This limits the performance of some aspects of the component to that of I/O speeds as well as requiring enough disk space for the task at hand. As physical storage has reduced in cost substantially and performance has improved, such concerns are not as relevant as they were in years past, but their relevance should not be overlooked either.

Additionally, at the time of writing, coverage of the *wrapper* mode was not complete and was limited to those functions necessary to confirm validity of the approach. As such, before the *wrapper* mode module of the **libsumor** component can be extensively used it will need to be extended in order to support the larger collection of options available in SUMO.

Finally, while a reasonable level of filtering is available when operating in the *synthetics* mode, a more robust and analyst extensible filtering module would be desirable so that the analyst can perform custom filtering and avoid having to post-process the resulting synthetics.

4.2 Quantitative

To quantitatively evaluate the proposed framework, we focus on demonstrating the ability to calibrate simulation demand from inferred demand quantities. The premise of this argument is that if the simulation used to produce the synthetics can be calibrated from quanti-

[5] See section 3.2.1 for our argument in favor of using a trip-based approach.

ties inferred from the reference trajectories, the resulting synthetics will be more reflective of reality. As demand calibration procedures are extremely varied, a worthwhile rank comparison would be extremely difficult as each calibration method may try to optimize various elements. On something as complex as a traffic simulation, extracting meaningful results would be extremely hard if not impossible.

Instead, because the **geoaware** component is meant to be general to a variety of simulation situations, we believe a much more informative discussion centers around demonstrating the ability to capture various desirable demand quantities for calibration and illustrating how such a capability compares to the literature and commonly employed techniques. As such, we do not seek to produce quantifiable *metrics* but instead wish to quantifiably justify the *ability* of the framework to infer various demand quantities from the reference trajectories. We present our evaluation as a series of propositions and (i) demonstrate the ability to calibrate the quantity, (ii) explore how other approaches in literature and practice calibrate such a quantity, and (iii) compare our approach to the reviewed approaches.

In synthetics production, commonly used techniques (particularly MOGs) overlook the importance of seasonality in the modeling process. As argued earlier in this work, human mobility exhibits strong seasonal patterns and therefore, being able to capture such seasonality is important and marks an improvement over those techniques which are unable to do so. The proposed framework explicitly incorporates seasonality when constructing the the macro-level demand, micro-level departure time profiles and micro-level edge weights for each block.

The **geoaware** component of the proposed framework makes contributions to three calibration metrics when calibrating the demand of a traffic simulation. The first contribution focuses on demonstrating the ability to incorporate seasonal trends into the macro-level movement between TAZs. The second contribution extends such seasonality to departure time and edge weight selection. The third and final contribution demonstrates the ability to

replicate paths from the reference trajectories. We begin by demonstrating the incorporation of macro-level seasonality.

4.2.1 Replication of Macro-level Demand

Rational, Macro-level Simulation Demand

Proposition 1

The proposed demand model produces statistically rational, macro-level simulation demand

As has been cited previously, the aim of the GeoAware framework is not to replicate the collection of reference trajectories (provided to the framework as input) but rather to infer transferable characteristics from such trajectories which can be used to produce synthetic traffic incorporating such properties. Inferring the requisite macro-level demand information from trajectories is straightforward as they can be aggregated to any spatial and temporal resolution and the quantity of departing trips between each block[6] can be easily extracted. If the proposed model were to stop here, the results would merely be a historical time series of departures not a model of agent demand.

Instead, the proposed approach models demand by fitting a seasonally-aware model to each block. Seasonally modeling demand (i) enables the incorporation of historical demand, (ii) allows the model to be used in situations which are yet to be observed, and (iii) captures typical demand behavior. The **geoaware** component utilizes the SDSBM presented in [184] and outlined in section 3.2.3 as it provides a solid, statistically-backed model for incorporating the affects of seasonality. Much literature has demonstrated the influence of seasonal cycles (such as a week) on mobility patterns [199] [142] [87] [184] [157] and the SDSBM has been shown to better model seasonal phenomena when compared to models which lack explicit consideration of seasonality [184]. Furthermore, be-

[6]Recall, that a block is a directed pair of TAZs. In a network with n TAZs, n^2 blocks exist.

cause the signal and noise terms of the SDSBM are assumed to be Gaussian distributed in our framework, the model is the minimum mean square error (MMSE) estimator of the demand[7] [117]. For the evaluation of the SDSBM, the reader is encouraged to consult [184] or [183].

Seasonal Simulation Demand

> **Proposition 2**
>
> *Calibrating macro-level simulation demand using the proposed demand model captures seasonal demand patterns*

To demonstrate that the **geoaware** component is capable of replicating demand which exhibits seasonality, a 40 element synthetic demand signal is produced that follows a sinusoidal pattern and has a periodicity of length four (4). Without loss of generality, each element of the signal is assumed to represent the demand over a period of one day. Additive white Gaussian noise ($\sigma = 1000$) is added to perturb the signal. Synthetic trips are generated according to the quantity specified by the demand signal and were simulated using the *synthetics* mode of the **libsumor** component with only the onset and terminus location of each trip retained. Once the synthetic trips are produced, the records were fed into the **geoaware** component and the the block demands generated.

A comparison between the original demand signal and the inferred demand signal is presented in figure 4.1. As is clearly evident by the figure, the seasonal demand used in generating the trajectories is captured by the **geoaware** component, demonstrating its ability to capture seasonal patterns exhibited in the reference demand[8].

Current Approaches

Traffic planners are often concerned with confirming that a traffic model can successfully approximate field data, particularly that of local traffic counts and travel surveys [206].

[7] The interested reader is encouraged to consult Chapter 13 of [117] for the derivation.

[8] Those interested in a quantitative comparison between the SDSBM and other block models concerning the ability to recover seasonal phenomena are encouraged to consult [184] or [183].

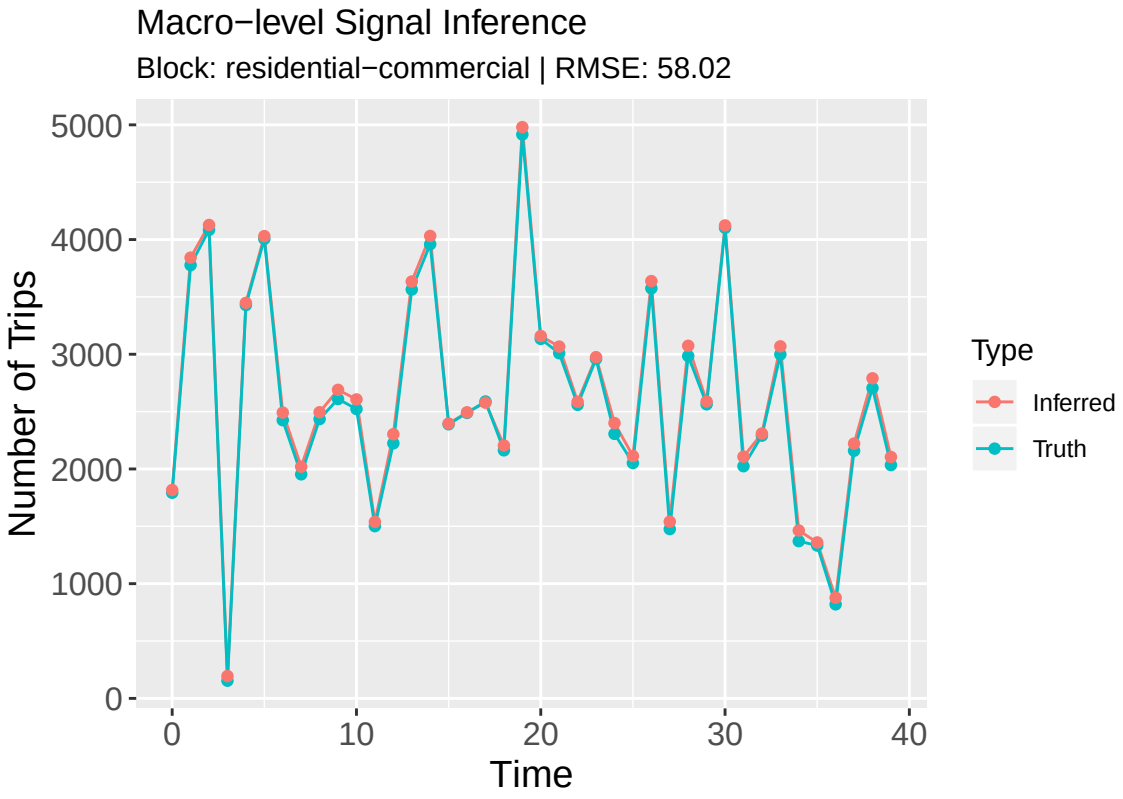


Figure 4.1: Comparison between the truth demand signal and the inferred, macro-level (block) demand signal. The initial values of the observation noise and transition noises for the MLE procedure were naïvely set to 1 as no prior information concerning these quantities is known [183]. To perform MLE, the Nelder-Mead optimization algorithm was used with the maximum number of iterations set to 500. See the *References* section of `R`'s `optim` function (`?optim`) for the algorithm details. Data Source: synthetic

To aid in this process, many traffic simulators and literature pieces provide algorithms for tuning the path choices of the agents such that the simulated edge counts approximate those observed in the field [21]. Typically, when one refers to "demand calibration" in a traffic planning context, such a *path-level* algorithm is envisioned[9]. In the proposed framework, path-level calibration is not of primary interest[10] but instead capturing patterns of mobility that reflect what is typically observed. The word "calibration" is not typically used to describe this procedure and instead such a process may be referred to as "demand generation". Despite the naming semantics, such a process is a calibration procedure as it makes the simulation more reflective of reality.

The UTP procedure (see section 3.2) on which our work is based, generates demand during the first two steps of the procedure, namely the trip generation and distribution steps (see section 3.2.3 for details). Recall, that the trip generation step focuses on creating a pair of trip productions and trip attractions for each TAZ while the trip distribution step allocates trips between directed TAZ pairs known as blocks. Trip productions are calculated per TAZ and typically use a cross-classification technique that tabulates rate parameters for certain household categories and applies those rates to the projected number of trips in that category for some target interval [206] [226] [48]. Natural sources for the household-level socio-economic data are census data and travel surveys [48] [226]. Trip attractions are also calculated per TAZ but are often harder to estimate accurately due to a general lack of survey data quantifying the effects of different establishment types on attracting trips [206]. This lack of detailed information typically leads to the use of a linear regression model on socioeconomic factors indicative of activity, such as employment or school enrollment [206]. Temporal affects can be incorporated at the trip generation step by constructing separate trip generation models for each time period [188] [33].

Even though cross-classification [206] [129] [5] and linear regression [206] [188]

[9]The calibration process need not be restricted to calibrating path selection using edge counts. The DynaMIT simulator presents a general approach for calibrating demand *and* supply quantities [28].

[10]Although, a path-level calibration technique could certainly be coupled with such a macro-level technique.

[129] [5] are the typically employed methodologies, other techniques include[11]: (i) using a Tobit model to handle negative trip rates (caused by negative regression factors) [59] [207], (ii) defining trip generation using logistic regression [107], (iii) capturing the discrete nature of trip rates using Poisson [22] or negative binomial regression [209], (iv) using a multiple classification technique that attempts to eliminate drawbacks of classical cross-classification [202], (v) a neural network approach [113], and (vi) using cell phone data [43].

In the typical UTP framework, once trip productions and trip attractions have been generated, a trip distribution procedure optimally allocates trips between TAZs subject to a cost function on attributes involving attractiveness, cost and perception [44]. Examples of such factors include employment, number of households, travel time, walk time after arrival, monetary cost, destination parking rates, toll roads, income level and area safety [44]. Gravity models [206] [58] [44] are typically employed for the assignment procedure although random utility models [27] [206], intervening opportunities methodologies [203] [58], hybrid gravity-opportunity mixtures [223] [9], state space models [12] [55], Bayesian-based approaches [75] [134], and activity-based[12] techniques [36] [24] [149] are also valid options[13].

Many studies have extended the traditionally static trip distribution step to model the affects of time on trip distribution. The most common approach is to segment the generated (pre-distribution) or distributed (post-distribution) trips according to departure percentages [206] [191]; however, other approaches present a completely overhauled model that explicitly considers time dependence [12] [55] [75] [36] [24].

While the above discussion covers the wide gamut of demand generation procedures explored in the literature, the procedures employed for demand calibration by modern traf-

[11] The interested reader is encouraged to consult [48] for a detailed comparison of some of these methodologies.

[12] Activity-based approaches actually model in terms of tours which are a sequence of activities that begin and end at home [36] and can be translated into TAZ pairings by formatting as origin-destination matrices [185].

[13] An excellent review of trip distribution models is provided in [44].

fic simulators[14] is typically much more restricted. Most modern traffic simulators either assume that demand already exists in some usable format or implements a demand generation procedure using the traditional cross-classification, regression and gravity models of the UTP procedure. Such an approach is not surprising, as the four-step UTP procedure is the most common traffic planning framework, making any tool that implements models for such a framework immediately more useful. Even the fact that many traffic simulators assume demand information is not unwarranted as there is a clear distinction between the modeling of demand and the routing of such demand through a network in terms of both the models and software required.

In our review of related MOGs (moving object generators; see chapter 2), we were unable to find any work that explicitly incorporated seasonality when generating synthetics. While [102] and [170] do not explicitly mention seasonality, because they construct synthetics from patterns, a seasonal pattern could be established and used to generate representative trajectories. Additionally, [108] does not model seasonal effects but it does infer a collection of distributions from field data in order to model "typical" behavior.

Comparison Against Current Approaches

Having presented a review of the current approaches to macro-level demand, both in the literature and in practice, we contrast the GeoAware framework with the current. The GeoAware framework differs from the current in three significant areas, namely, (i) its source of calibration data, (ii) its effectiveness of calibration, and (iii) its focus on transferability.

Data Source. UTP-based models rely on various socioeconomic variables to indicate the quantity and distribution of trips throughout the network. The use of socioeconomic variables for establishing demand is well studied and is a widely employed technique that exhibits a great deal of rationalism. Variables such as car availability, population levels,

[14]The list of reviewed simulators was: Aimsun, Cube Voyager, DRACULA, DynaMIT, MatSIM, MITSIM-Lab, Paramics, SUMO, and TSIS-CORSIM. These traffic simulators where chosen due to their prevalence in the literature and where information could not be found publicly, user documentation was consulted.

employment levels and other such factors can be easily seen to affect the quantity and distribution of trips [44].

Just like any model, however, such models have their drawbacks. In the context of travel demand, the reliance on socioeconomic variables (or any indicator variable) requires that the analyst (i) know the set of variables affecting demand, (ii) be able to capture such variables in the field, and (iii) construct a proper model incorporating such variables. Certainly, models have been constructed for such socioeconomic indicators that have proved useful in practice, but a fundamental question always remains concerning how reflective the model is of reality.

In contrast, the **geoaware** component constructs a predictive model based on direct observations captured through trajectories thereby avoiding the cumbersome process of identifying relevant socioeconomic variables and establishing a model from such data. When one is merely trying to obtain the quantity and distribution of trips and is not attempting to understand the reasoning for such such demand, trajectory data proves a far superior choice over that of indicator variables due to its correspondence with a real-world actor. With trajectory data, any model of demand can be compared with the original trajectories. In models based on socioeconomic variables, no direct link exists to a real-world actor due to the aggregation of the statistics and thus the demand model cannot be compared with actual observations.

Some MOGs do support the use of trajectory data (for instance [102] or [170]) for inferring patterns but the ease of incorporating information from such trajectories depends on the framework. In general, the process is not straightforward and often requires an analyst to define an inference procedure to obtain the relevant information. Such a task places an extra burden on analyst.

Given the prevalence of location acquisition systems nowadays [143], calibration of macro-level demand from trajectories is much more realistic than in times past, however, we do not wish to imply that the aforementioned approaches are obsolete. As noted previ-

ously, the regression and gravity models of the trip generation and trip distribution steps, respectively, are quite common in practice. Such techniques are vital when trying to model how changes in scenario characteristics, such as land use changes, population adjustments, or demographic shifts, change the resulting demand. In such situations, trajectories cannot be directly utilized as they often lack the meta-information necessary to ascertain such information.

Calibration. The second area of difference between the reviewed approaches and the GeoAware framework is the focus the GeoAware framework places on producing and utilizing calibrated, macro-level demand. In many works, demand calibration is rightfully cited as a necessary step in simulation calibration but as noted in our review of relevant traffic simulators, many modern simulation frameworks do not have a mechanism to generate calibrated demand. Separating the calibration from the actual simulation of demand makes sense as the two processes are mostly independent, however, it is valid to question how relevant the results of a traffic simulation are if they are not reflective of reality. While the simulator must be based on a sound traffic model in order to be useful when analyzing real-world situations, it is equally important to have demand that reflects reality. The lack of a technique for natively generating demand in many traffic simulators means that additional work is put on the analyst to derive calibrated, macro-level demand; an effort which might be tangential to the task at hand.

In those traffic simulators which do provide a process for generating trip-based traffic, the approach typically relies on the aforementioned procedures of the trip generation and distributions steps. As noted previously, because of the historical and continued dominance of the UTP procedure, many modern traffic simulators, particularly those aimed at commercial applications, support the UTP procedure for generating traffic demand. Built-in UTP support allows an analyst to quickly generate demand but requires that socioeconomic data exist. In situations where all an analyst wishes to do is to generate traffic that reflects what is typically observed, the collection of various (and relevant) socioeconomic data presents

a hurdle which the analyst must overcome in order to utilize the UTP procedure.

Often, if MOGs support the creation of calibrated synthetics, the quantities and distributions driving such a procedure are assumed to be exogenously defined. Very few specifically detail procedures for producing the necessary quantities and distributions from field data [108] [84].

In contrast, the GeoAware framework is designed for comparatively rapid[15] generation of calibrated trip demand. In the proposed framework, rather than use socioeconomic variables which must be researched, gathered and integrated into a model, trajectory data is used to infer the macro-level mobility patterns and requires little processing once captured by a location acquisition device. Because trajectory data implicitly carries onset and terminus information (as well as a host of other information), aggregate socioeconomic variables are not needed to try and predict such quantities. Additionally, the **geoaware** component outlines a complete procedure for inferring relevant quantities from the trajectory data. By lessening the hassle of synthetics production, an analyst can focus on the task for which he/she needs the synthetics rather than focus on obtaining inputs which will produce sufficiently realistic results.

Transferable Patterns. The **geoaware** component also focuses on ensuring the patterns inferred are transferable. Often, traffic studies argue that a traffic model is valid by demonstrating that it is able to recreate known traffic conditions (with a certain degree of accuracy) of some base year [206]. In studies attempting to address how environmental and societal changes affect demand, establishing the ability to recreate base-year conditions is critical to the argument of the researchers as it legitimizes the conclusion that changes in environmental and societal characteristics produced the changes in travel demand. But as [172] argue, the preoccupation with recreating base-year conditions may limit the utility of the simulation in situations which do not resemble such base-year conditions. If the model is only capable of accurately recreating base-year conditions, such a model will be of limited

[15] By "rapid" we do not necessarily mean fast but instead are highlighting how little overhead is required to generate synthetic demand using the GeoAware framework.

utility.

Furthermore, the reliance of many modern traffic simulators on socioeconomic variables when calibrating macro-level demand may also limit how applicable the simulation is in other situations. Because such models are constructed from a finite collection of socioeconomic variables, the relevance of the variables and the accuracy of the model in non base-year conditions determines how applicable the simulation is in such situations. If the socioeconomic data, traffic counts, speed or any other field-data source are specific to certain conditions, it is senseless to try and make such data representative of "typical" data. Indeed, even when data is supposedly representative of "average" conditions, one must investigate the generation procedure for such data. For instance, path-level demand calibration often relies on traffic counts but traffic counts are often communicated as averages and may have been expanded from counts taken over a shorter duration [118]. How accurate the process is at expanding short-duration counts to annual averages will dictate how faithful such data is to reality. As [21] points out, mismatches between "average" and "observed" demand will likely result in flow that is unrepresentative of reality.

Instead of focusing on base-year recreation, the GeoAware framework focuses on calibrating behavioral properties which are applicable in non base-year conditions. The macro-level demand calibration component of the GeoAware framework seeks to extract mobility patterns by observing historical trends in pickup and dropoff locations of the trajectories. By incorporating seasonal patterns, the resulting simulation is calibrated based upon the trends observed over multiple time periods rather than from a single snapshot or "average" conditions. The incorporation of such seasonal patterns constitutes a major advantage of the GeoAware framework over other MOGs which do not explicitly consider seasonality when producing synthetics.

Because the demand model is produced directly from the observed trajectories, the analyst need not be concerned with ensuring that the produced model is representative. Of course, the analyst must ensure that the trajectories represent the conditions he/she wishes

to recreate but unlike the typically heuristic MOGs or traffic simulators which calibrate using socioeconomic variables, one does not have to check that the resulting demand model accurately represents real world demand. In the GeoAware framework, the data-driven approach ensures that the resulting model is reflective of the underlying data source by capturing trends rather than focusing on base-year recreation. This results in a simulation that is transferable to other situations which meet the modeling assumptions.

4.2.2 Replication of Departure Times

Proposition 3

Calibrating simulation demand using the proposed demand model captures seasonal, micro-level departure time patterns

To demonstrate that the GeoAware framework is capable of capturing seasonal departure time patterns, we construct a simulation scenario similar to that presented in section 4.2.1. Recapping the experimental setup presented there, a 40 element synthetic demand signal is produced by adding white Gaussian noise ($\sigma = 1000$) to a sinusoidal signal having a periodicity of length four (4). Without loss of generality, each element of the demand signal is assumed to represent the demand for an entire day.

Once the demand has been constructed, we define a departure time profile for each time period in the seasonal interval (i.e. each element of the demand is seasonally related to other other demand elements and all related elements share the same time profile). Because the periodicity of the demand signal is of length four, four departure time profiles are defined. We split each profile into 96 data points (i.e. 15 minute intervals) and construct departure time profiles for the period according to (i) uniform, (ii) Gaussian, (iii) sinusoidal, and (iv) composite distributions. Synthetic trips are then produced according to the quantity specified by the demand signal with the departure times for the demand distributed according to the departure time profiles associated with each demand element. For

example, the thirteenth element of the demand signal (day thirteen) would belong to the second time period in the seasonal cycle ($13 \bmod 4 = 1$, where the index is zero-based). The synthetic trips are simulated using the *synthetics* mode of the **libsumor** component.

Once the synthetic trips are simulated, the mobility records produced are fed into the **geoaware** component with the seasonality and simulation granularity parameters coming from those used when generating the truth demand. Once the block-level demand is inferred, the departure time profiles are inferred. A comparison between the departure time profiles used when generating the truth demand and those inferred from the generated trajectories is presented in figure 4.2. As one can see, the empirically derived departure time profiles captured the seasonality present in the original truth data. Because we simulate multiple days and each day is seasonally related with nine other days, we effectively perform four Monte-Carlo simulations where the time departures in each period of the seasonal cycle is simulated ten times.

Current Approaches

A survey of transportation planning literature reveals that the models which consider departure time are often activity-based. The association of departure time selection with activity-based models makes sense as the commonly used, trip-based traffic planning models tend to be more macro-focused whereas activity-based models are micro-level and seek to justify the observed travel patterns based on behavioral attributes [206]. [217], [197], and [2] present early approaches to departure time modeling from an activity-based perspective.

Typically, in transportation planning contexts, departure time models are constructed as a discrete choice model from cross-sectional data deemed relevant to departure time choice [50] [2] [14] [232]. In the literature, factors such as travel time, travel cost, transportation mode, arrival flexibility, gender, age, family status and occupation have all been used when fitting a model of departure time. As some of these factors are subject to the affects of congestion, some traffic assignment works use such models to assign departure times based upon the prevailing traffic conditions within the simulation [71] [28].

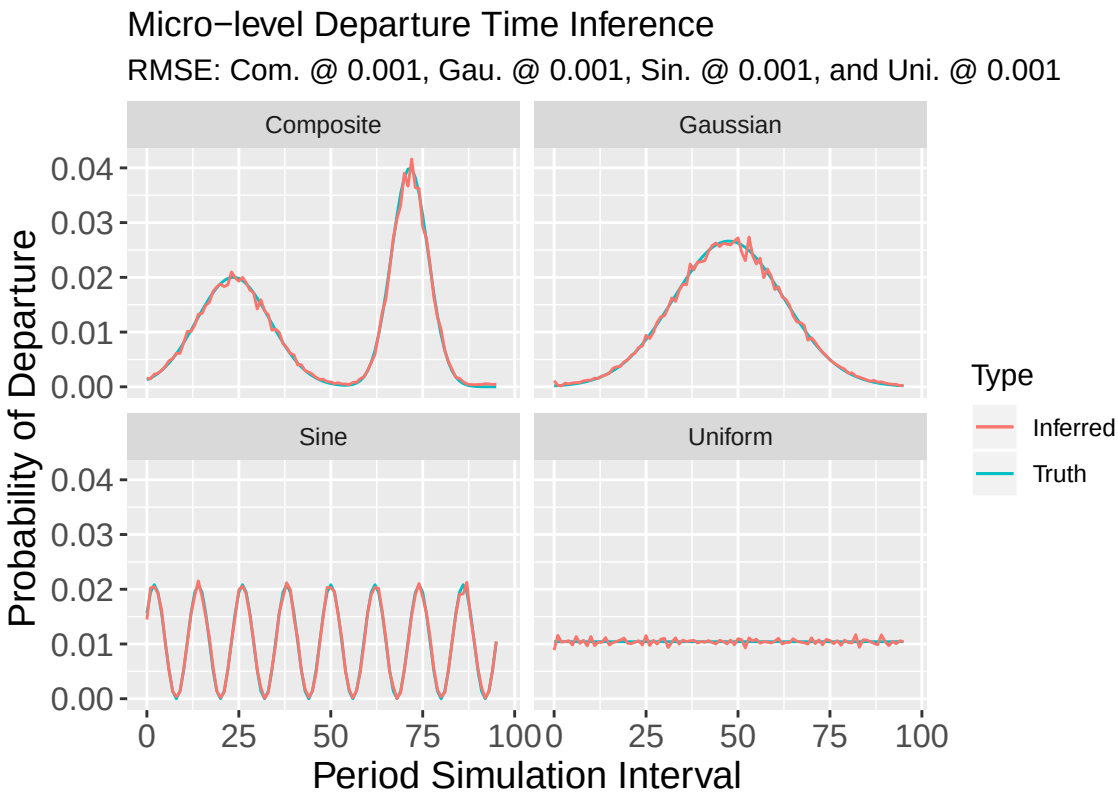


Figure 4.2: Comparison between the truth departure time signals and the inferred departure time signals. The x-axis indicates the simulation interval within the period. In the experiment, a seasonality length of four was used. The initial values of the observation noise and transition noises for the MLE procedure were naïvely set to 1 as no prior information concerning these quantities is known [183]. To perform MLE, the Nelder-Mead optimization algorithm was used with the maximum number of iterations set to 500. See the *References* section of `R`'s `optim` function (`?optim`) for the algorithm details. Data Source: synthetic

Although utilizing a discrete choice framework on cross-sectional data is the most prevalent technique, other approaches to modeling departure time profiles do exist in the literature. Examples include constructing an input-output N-curve from flow data [103], reverse engineering departure times from a calibrated OD matrix [125], and using GPS data to estimate door-to-door travel time and its affect on departure time choice [168].

With the exception of the DynaMIT traffic simulator, the reviewed traffic simulators[16] do not focus on calibrating micro-level departure trends. The DynaMIT traffic simulator calibrates driver responses (such as departure time) through behavioral models based on historical patterns and incoming surveillance data [28]. While not many traffic simulators support calibration of trip departures, some traffic simulators, such as AIMSUN [4] or DRACULA [139], allow an analyst to specify a distribution to use for departure times.

Many MOGs support the use of statistical distributions to specify various temporal properties, however, only a few specifically detail methodologies for inferring temporal properties from field data. Because [102] and [170] generate by example and construct their synthetics from mobility patterns, such works are capable of modeling departure times pattern in data. Despite this, however, neither work outlines an algorithm for creating such patterns from data and therefore an analyst must construct such mobility patterns. [108] and [84] use field data for calibrating temporal properties and provide methodologies for incorporating this information into the framework. [108] uses such field data to produce synthetic CDRs while [84] uses the data to construct a synthetic population which is simulated to produce symbolic trajectories. [200] present a technique that uses regression on trajectory data to produce synthetic demand that is then routed by SUMO.

Comparison Against Current Approaches

The **geoaware** component is easily distinguishable from the approaches in the literature and practice for three reasons. First, of the widely utilized traffic simulators, only one supports

[16]The same collection of reviewed simulators presented earlier. For reference, the list of reviewed simulators was: Aimsun, Cube Voyager, DRACULA, DynaMIT, MatSIM, MITSIMLab, Paramics, SUMO, and TSIS-CORSIM.

any form of calibration of departure time based upon data. While it is likely that such an absence is a result of the typically macro-level analysis in which traffic simulators are employed, it is important to note that microscopic traffic simulators are applicable to a wide variety of scenarios which extend beyond supplying data for macroscopic analysis. Thus, the inability of most traffic simulators to utilize calibration data to make the departure trends more reflective of reality is discouraging. The **geoaware** component stands in complete contrast to such an approach as it explicitly constructs its model of demand by inferring micro-level departure patterns from the collection of reference trajectories.

Second, in contrast to the approaches outlined above, the **geoaware** component does not rely on cross-sectional data. One common source of cross-sectional data are travel surveys or census products. Unfortunately, surveys are (i) not routinely performed, (ii) feature a limited sample size [50] [17], and (iii) have noisy responses [17]. Additionally, as such data may not explicitly provide information on trip departures, a model must be constructed from such factors. In contrast, modeling departure times in the **geoaware** component uses trajectory data that is ubiquitous nowadays, can often be captured passively[17], and provides micro-level information — including departure time.

Third, in contrast to available MOGs, the GeoAware framework not only allows departure time to be specified by a statistical distribution, it provides a method for inferring such a distribution from trajectory data. As we showed in our review, such an approach is not common. Additionally, the empirical PMFs constructed by the proposed framework heuristically incorporate seasonality, ensuring that the PMF constructed is representative of "average" or typical conditions.

4.2.3 Replication of Incident Edge Weights

[17] We do not wish to construe the idea that such data should be collected without a user's permission but instead simply note that often capturing trajectory data does not require the user to do anything (e.g. a camera-based vehicle tracking system).

Proposition 4

Calibrating simulation demand using the proposed demand model captures micro-level, onset and terminus edge selection weights

Similar to the previous experiments, to demonstrate that the **geoaware** framework is capable of capturing seasonal edge weights associated with the onset and terminus locations of a trip, a 40 element, sinusoidal demand signal with additive white Gaussian noise ($\sigma = 1000$) is constructed. Once again, the periodicity of the signal is of length four (4) and each demand element is assumed (without loss of generality) to represent the demand over an entire day.

Now, however, instead of defining departure time profiles for each period in the seasonal cycle, edge weight profiles are defined for the onset and terminus edges. Because an edge weight profile had to be constructed for both the onset and terminus edges of each period in the seasonal cycle, a total of eight edge weight profiles are defined and consist of uniform, Gaussian, composite and sinusoidal distributions. The number of bins in each distribution is dependent upon the number of edges which belong to the onset and terminus TAZs, respectively. Synthetic trips adhering to edge weight distributions are produced and simulated using the *synthetics* mode of the **libsumor** component.

Once the truth trajectories are generated, they are fed into the **geoaware** component with seasonality and simulation granularity parameters matching those used when creating the synthetic trajectories. Once the block-level demand is inferred, the edge selection weights were inferred. Figures 4.3 and 4.4 present the empirically inferred quantities alongside the truth distributions used when generating the synthetic trajectories. Once again, as in the demand and departure time experiments, the inferred demand accurately captures seasonally distributed edge weights. Because we simulate multiple days and each day is seasonally related with nine other days, we effectively perform four Monte-Carlo simula-

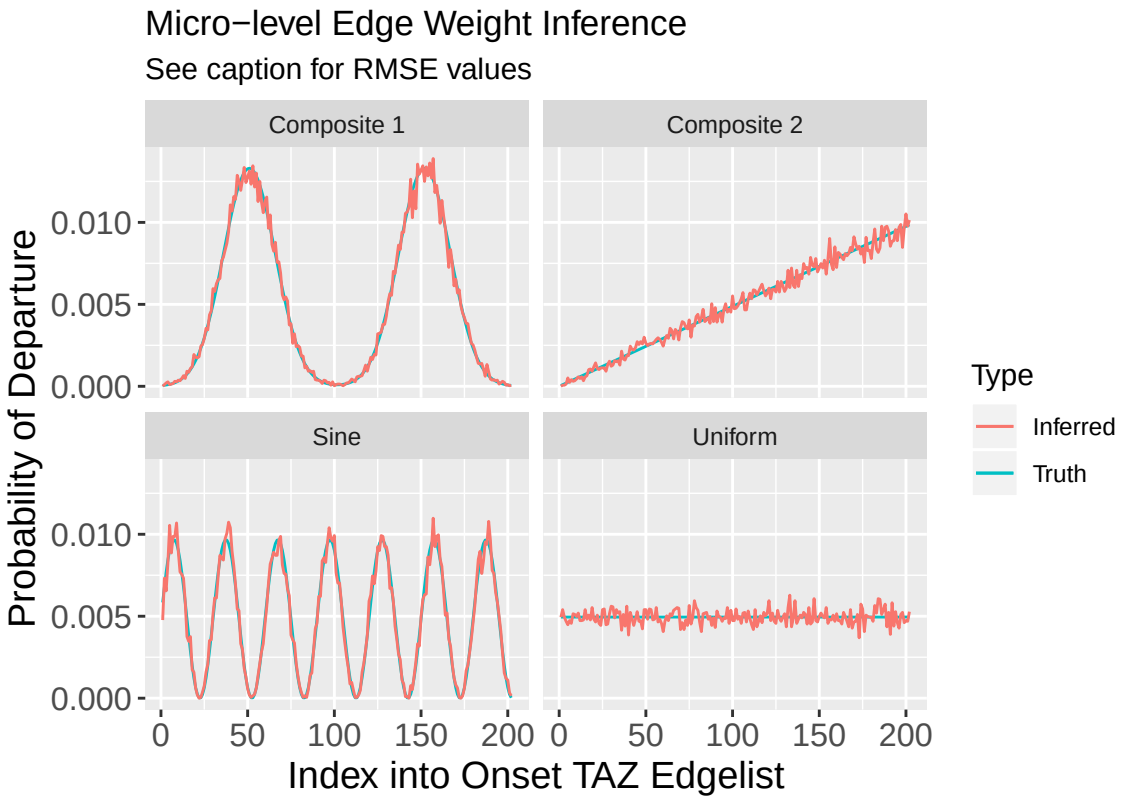


Figure 4.3: Comparison between the truth edge weights and the inferred departure edge weights for the onset TAZ. The x-axis indicates the index into the list of edges associated with the onset TAZ. Uniform RMSE: 0.0004531695. Composite 1 RMSE: 0.0004189938. Composite 2 RMSE: 0.000384571. Sine RMSE: 0.0004959786. Total of $N = 103391$ trip records. The initial values of the observation noise and transition noises for the MLE procedure were naïvely set to 1 as no prior information concerning these quantities is known [183]. To perform MLE, the Nelder-Mead optimization algorithm was used with the maximum number of iterations set to 500. See the *References* section of `R`'s `optim` function (`?optim`) for the algorithm details. Data Source: synthetic

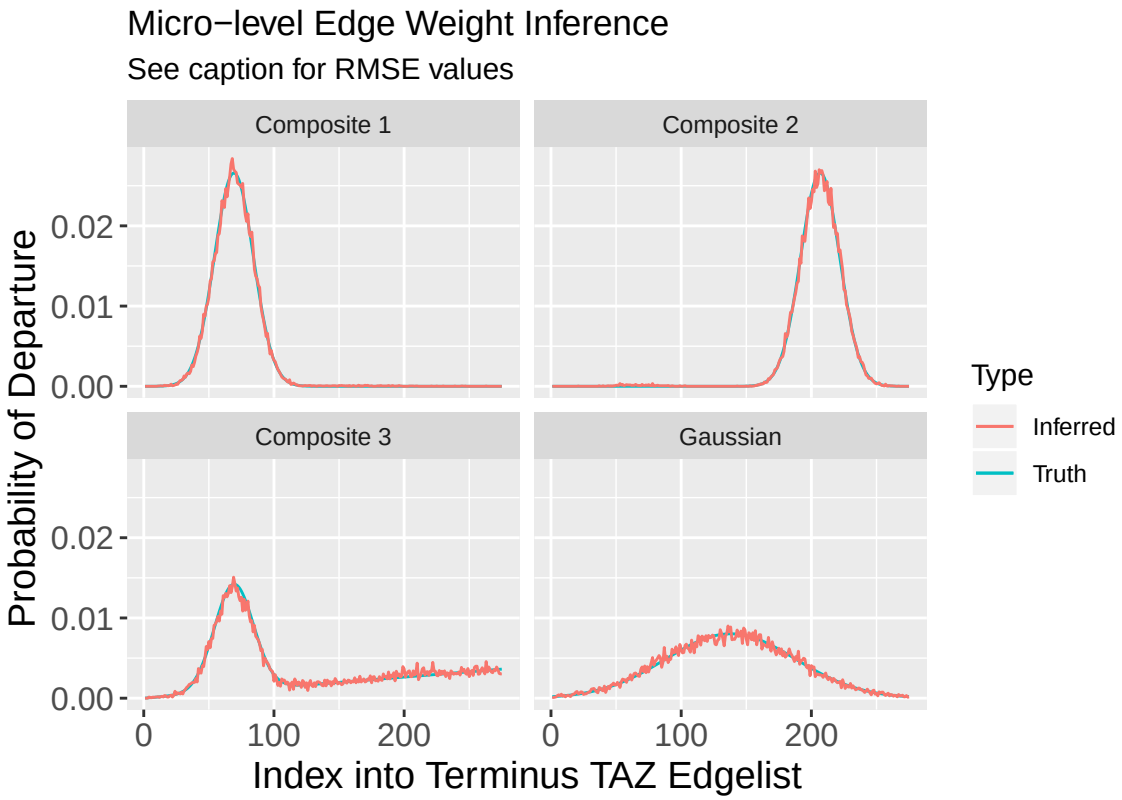


Figure 4.4: Comparison between the truth edge weights and the inferred arrival edge weights for the terminus TAZ. The x-axis indicates the index into the list of edges associated with the terminus TAZ. Gaussian RMSE: 0.0004317771. Composite 1 RMSE: 0.0003508453. Composite 2 RMSE: 0.0003736785. Composite 3 RMSE: 0.0004666017. Total of $N = 103391$ trip records. The initial values of the observation noise and transition noises for the MLE procedure were naïvely set to 1 as no prior information concerning these quantities is known [183]. To perform MLE, the Nelder-Mead optimization algorithm was used with the maximum number of iterations set to 500. See the *References* section of `R`'s `optim` function (`?optim`) for the algorithm details. Data Source: synthetic

tions where the selection of edges in each of the days is simulated ten times.

Current Approaches

The most common techniques employed in the literature for estimating the locations associated with a trip tend to be associated with that of activity-based, discrete choice frameworks. Similar to the activity-based, discrete choice frameworks reviewed in our discussion of departure time, such models are often based on behavioral data. [29] [131] [232] typify the approaches which are prevalent in the literature. Often, the destination choice is the quantity of interest while the onset location is assumed exogenously.

As most metropolitan planning tools represent traffic at the TAZ-level (a geospatial polygon), agents are typically inserted and received at a single centroid location [206] or at several locations throughout the zone [91]. The nodes are assumed to be connected to a roadway within the TAZ. Thus, in these models, when one discusses an onset and terminus location, one does so at the zonal level.

Some approaches break from convention and model travel at a sub-TAZ level but such approaches are rare in the literature. [109] and [145] perform sub-TAZ modeling by dividing the TAZ into subareas which can be chosen evenly or based upon socioeconomic variables. [218] takes a different approach and performs sub-TAZ modeling by using a discrete choice model to model the route selection process from home to the access points of the transportation network.

Upon review of a representative collection of modern traffic simulators[18], none were found to specifically incorporate the modeling of onset or destination edges from calibration data. Modern traffic simulators often support macro-level TAZs but the exact methodology by which such simulators select individual edges is certainly not standardized. Microscopic traffic simulators such as VISSIM [175], AIMSUN [4] and SUMO [140] allow an analyst to specify the quantity of trips at particular edges but such quantity information

[18]The same collection of reviewed simulators considered throughout this section. For reference, the list of reviewed simulators was: Aimsun, Cube Voyager, DRACULA, DynaMIT, MatSIM, MITSIMLab, Paramics, SUMO, and TSIS-CORSIM.

must be externally defined.

Additionally, it is also worth noting that since microscopic simulators often rely on OD matrices for demand generation, one way to model individual location demand would be to use individual locations rather than aggregate zones when crafting the OD matrices. Of course, such a procedure is only applicable if the data provided supports such a fine-grained representation.

Many MOGs support the ability to specify a distribution that governs the selection of onset and terminus edges. Nevertheless, only a few specifically detail a procedure for setting such weights from relevant field data. [108] uses real CDRs or other relevant field data to construct distributions governing the selection of various location information, namely home, commute distance, work, and hourly population density distributions. [64] uses statistical data to set the home and work regions in the *BerlinMOD* framework. The ST-ACTS framework uses field data to construct a synthetic population which is then simulated to produce symbolic trajectories [84]. [102] and [170] are also able to recreate edge patterns if the general mobility patterns on which the synthetics are based captures such information.

Comparison Against Current Approaches

As noted above, the predominant technique for modeling onset and terminus choice is using discrete choice models. While insight can gathered from understanding what behavioral attributes drive the selection of incident locations, such behavioral data is often difficult to obtain as many factors are unseen and requires specifying or identifying a model of mobility using such behavioral data [50] [192]. In contrast, the GeoAware framework enables a user to quickly infer incident location weights through the construction of an empirical PMF, eliminating the need to find and model socioeconomic data. Of course, the drawback to such an approach is that the model is not explanatory but the benefit of such an approach is the fact that there is no concern over ensuring that the model is representative of the data.

Additionally, the macro-level zonal approach of modern traffic simulators poses se-

rious problems when trying to perform low-level simulation or when trying to accurately represent demand [91]. Despite the power of microscopic traffic simulators, modern traffic simulators often rely on *macroscopic* routing principles. The reliance on macroscopic principles is likely due to the focus of travel practitioners on macro-level analysis and the fact that, in the past, commonly available data sources tended to be aggregate in nature. In contrast, the **geoaware** component extracts micro-level patterns from the reference trajectories and uses the extracted patterns to perform sub-zonal edge selection based upon the inferred patterns.

While many MOGs support the ability to calibrate edge selection from field data, the source of information is typically aggregate in nature and therefore, often does not provide the necessary, low-level details. In contrast, when trace-level trajectories are provided to the GeoAware framework, seasonally-aware PMFs can be constructed for choosing various edges based upon the block to which an agent belongs. Such a PMF enables the weights to be time dependent and makes the selection of onset and terminus locations dependent on each other. Not many MOGs support the ability to make the selection of onset and terminus edges dependent on each other.

4.2.4 Replication of Paths

Proposition 5

Calibrating simulation demand using the proposed demand model permits a certain percentage of trips to duplicate the paths observed in the reference trajectories

Another technique an analyst can use on trace-level trajectory data is the path replication algorithm provided by the **geoaware** component. The goal of the path replication algorithm is to increase the realism of the demand obtained by requiring that a certain percentage of the generated demand follow the paths observed in the reference trajectories rather than being routed by the simulator. By replicating a certain percentage of the paths

Quarter	Truth	Inferred
Quarter 1	0.25	0.28
Quarter 2	0.50	0.52
Quarter 3	0.75	0.76
Quarter 4	1	1

Table 4.2: Results of path replication experiment.

in the reference trajectories, an analyst can control how much of the routing in a simulation is performed by the routing algorithm versus duplicating paths in the reference trajectories. Clearly, if the simulated vehicles are routed according to the paths from the reference trajectories, that percentage of trajectories is guaranteed to follow the paths observed in the real world (as captured by the trajectories).

To quantitatively demonstrate that the simulated trips produced by the **geoaware** component can clone a certain percentage of the reference trajectories, we construct a simple simulation scenario and demonstrate the ability to create demand where a certain percentage of the trips have paths which mirror those observed in the reference trajectories.

The experiment begins by creating a collection of synthetic trips. Once the synthetic trajectories are created, a **geoaware** object is instantiated using the synthetic trajectories as the reference trajectories. Prior to running the trip generation procedure, we set the percentage of trips which should have paths mimicking those in the reference trajectories for each period in the seasonal cycle.

Next, the trip generation procedure is called. The reader may recall from the methodology section (see section 3.2.5) that it is within this procedure that a certain percentage of the trips are assigned paths from the reference trajectories while the remaining trips are distributed according to the relevant departure time and edge weight distributions. The results of the path replication procedure is demonstrated in table 4.2. The table shows the ability of the **geoaware** component to replicate paths from the reference trajectories. As the framework code takes the ceiling when obtaining the quantity of trips to generate, it is reasonable to expect the inferred quantities to be slightly higher than truth.

As noted in 3.2.5, this technique assumes two things. First, in order to replicate path-level details of a trajectory, the reference trajectories must be at such a resolution. Check in/out trajectories will not work because they lack the low-level information necessary to identify the path taken between two points. Second, this process assumes that a map matching procedure of sufficient quality exists to take the raw GPS locations and convert them into the series of edges within a network. How well the map matching algorithm performs this task directly translates into how usable the resulting trajectories are. As map matching is obviously beyond the scope of this work, for this evaluation we rely on the open source Fast Map Matching algorithm [231].

Current Approaches

In modern traffic simulations, the ability to infer the paths taken by real-world actors from calibration data is not present. To be sure, many support the ability to detail predefined routes but such procedures require that the paths be defined exogenous to the framework [140] [4] [175]. The inability of modern traffic simulators to infer the paths taken by agents via calibration data is likely due to the fact that modern traffic simulators typically perform such calibration with aggregate statistics (such as traffic counts or speed-flow diagrams) which do not contain the information necessary for path-level inference.

While no reviewed MOG specifically attempts to replicate path-level details, the [102] and [170] frameworks are able to perform path replication if the mobility patterns used for synthetics generation are properly configured.

Comparison Against Current Approaches

As noted previously, the **geoaware** component is primarily focused on inferring transferable demand patterns. This results in a process that is focused on *reproducing mobility patterns* not *recreating base conditions*. Nevertheless, at times it can be helpful to inject known demand conditions into an otherwise pattern-calibrated simulation. The ability to recreate paths from the reference trajectories is beneficial in situations where an analyst wishes to ensure that a certain percentage of simulated vehicles follow the routing captured

by the reference trajectories but does not need (or does not wish) to construct a behavioral model of such route choice characteristics. Thus, path replication can be thought of as an aid for avoiding a much lengthier modeling process. Modern traffic simulators typically support the specification of demand through external procedures [140] [4] [175] but, to the best of the author's knowledge, none support such a procedure natively from a calibration source. Additionally, no reviewed MOG presents a method for calibrating such information directly from data.

4.3 Chicago Use Case

With the qualitative and quantitative evaluation now complete, we conclude the evaluation chapter by presenting a tutorial that details how to create synthetic trajectories from publicly available taxicab pickup/dropoff data for the city of Chicago, Illinois.

INPUT

Creating a collection of synthetic trajectories using the GeoAware framework begins by defining the necessary inputs. We specify these inputs below.

Reference Trajectories. The reference trajectories come from the publically available collection of taxicab pickup and dropoffs made available by the city of Chicago[19]. We specifically retain the pickup and drop-off community areas along with the the timestamps associated with each of these events. The data is cleaned to remove entries for which we do not have both a pickup and drop-off area and entries with erroneous timestamps. We use a month of trajectories from January 2016.

Roadway Network. The roadway network is derived from TIGER/Line shapefiles for the Chicago area[20]. We specifically retain only the records with MTFCC code "S1400" which corresponds to local neighborhood, rural and city roadways. To aid in computational fea-

[19]The latest collection of records can accessed at: https://data.cityofchicago.org/Transportation/Taxi-Trips/wrvz-psew

[20]TIGER/Line shapefiles are made publicly available by the United States Census Bureau at: https://www.census.gov/geographies/mapping-files/time-series/geo/tiger-line-file.html

sibility, we geo-fence the network to a few community areas. Because TIGER/Line shapefiles are commonly used in transportation studies, the conversion procedure was provided as a utility in the **geoaware** component.

Community Definitions. The reference trajectories are aggregated at two levels. One of the aggregation levels is that of a community area. The geospatial boundaries for these areas are provided by the city of Chicago[21]. We extract the bounding polygons for each area and format them for use by the **geoaware** component.

Timing Parameters. The SDSBM fit intervals are constructed at daily intervals beginning on January 1, 2016 and extending to February 1, 2016. We set the seasonal periodicity to be seven in order to model weekly cycles. The city of Chicago aggregates the pickup and drop-off times to the nearest fifteen minutes and therefore, in order to capture sub-day departure time profiles, the simulation granularity parameter is set to 900 seconds.

Configuration File. A configuration file is specified according to the format specified in section 3.1.5. The configuration file details (i) various SUMO parameters, (ii) **libsumor** settings, and (iii) the location of required files.

CALIBRATING DEMAND

The calibration of demand is handled by the **geoaware** component. An algorithmic outline of this component is presented in algorithm 4. The process begins by initializing a **geoaware** object with the reference trajectories, roadway network, community definitions, and timing parameters[22]. This step is shown in line 2 of algorithm 4. Next, TAZs are constructed detailing the edges which belong to each TAZ and the region that each edge falls into[23]. This step is shown is line 4 of algorithm 4.

After the TAZs have been initialized, one can now infer the macro-level demand patterns exhibited in the reference trajectories. As presented in section 3.2.3, the SDSBM is

[21]The current listing of community areas in Chicago is publicly available here: https://data.cityofchicago.org/Facilities-Geographic-Boundaries/Boundaries-Community-Areas-current-/cauq-8yn6

[22]The configuration file is not needed until the **libsumor** component.

[23]Recall that a TAZ may be comprised of multiple regions. See section 3.1.3 for additional details.

input : (i) trajectory data, (ii) roadway network, (iii) community definitions, (iv) timing parameters, and (v) a config file
output: a model of trip demand

```
1  # instantiate the geoaware component
2  geoaware ← geoaware$new(trajectories, roadway, communities,
     timing)
3  # generate TAZs
4  geoaware$generateTazs()
5  # generate macro-level block demands using SDSBM
6  geoaware$generateBlockDemands(initial_mle_guesses)
7  # extract micro-level heuristics
8  geoaware$generateTimeProfiles()
9  geoaware$generateEdgeWeights()
10 # generate trips from inferred quantities
11 geoaware$generateTrips()
12 # generate scenario
13 geoaware$generateScenario(config)
```

Algorithm 4: Modeling trip demands using the **geoaware** component

used for this process. The SDSBM requires the reference trajectories, the fit intervals and the seasonality length in order to produce a demand signal for each block (a directional pairing of TAZs). This process is handled by line 6 of algorithm 4.

With the macro-level demand now inferred, the calibration turns to extracting micro-level details. The first micro-level detail we seek is a model of departure times. To obtain this model, the reference trajectories are grouped into a bin based upon which seasonal period the record belongs to. The width of each bin is dependent upon the size of the fit intervals used with the SDSBM. For this tutorial, as presented earlier, the fit intervals have a width of one day (86,400 seconds) and the seasonal periodicity is set to seven. This means that each record in the reference trajectories is grouped into one of seven bins based upon what day of the week it occurred on. Once grouped, an empirical PMF of the departure times is constructed for each day at the resolution specified by the simulation granularity parameter. In this tutorial, this results in a 96-bin histogram as we model each day with fifteen-minute resolution (900 seconds). This step in the procedure is shown on line 8 of

algorithm 4.

The next step of the **geoaware** component is to construct the edge weight profiles. Unfortunately, due to privacy restrictions, the taxicab data used does not include such low-level information. Therefore, rather than construct an empirical PMF of such edge weights, we simply set the edge weights associated with each TAZ uniformly. We provide a custom function to uniformly set these weights given the number of edges belonging to a TAZ. This step is shown on line 9 in algorithm 4.

Once the macro- and micro-level details have been inferred, the final step is to specify trips using the inferred information. As the reference trajectories do not include path-level information, we are unable to replicate path-level details and therefore rely exclusively on the assignment algorithms supplied by SUMO. During trip specification, trip departure times and onset and terminus edges are drawn for the number of trips required in each simulation interval. The specification of trips is line 11 of algorithm 4.

The trips specified, the last step of the **geoaware** component is to create a scenario that incorporates the specified trips, the roadway network and configuration file. This task is handled by the method specified on line 13 of algorithm 4.

PRODUCING SYNTHETICS

Once a scenario has been constructed, the **libsumor** component is used to create synthetics with mobility patterns captured from the reference trajectories. Algorithm 5 details this process.

input : a simulation scenario
output: a CSV file containing synthetic trajectories

```
1 # the inputs
2 scenario ← a properly constructed simulation scenario

3 # instantiate synthetics mode
4 simulation ← libsumor(scenario, "synthetics")
5 # generate synthetics
6 csv_synthetics ← simulation::run (end_time)
```

Algorithm 5: Producing synthetic trajectories

As shown on line 4, the process begins by instantiating a **libsumor** object in *synthetics* mode. During instantiation, files detailing the (i) configuration of SUMO, (ii) vehicle types, and (iii) the trips are written out. Afterwards, the synthetics mode is instantiated by inferring the relevant details from the specified configuration file. Once instantiated, the process of generating synthetic trajectories is quite simple and only requires a single call as shown on line 6 of algorithm 5. Once the synthetics have been produced, the CSV file can be used for whatever purposes require synthetic trajectories.

Chapter 5 — Conclusion

Synthetic trajectories are useful for many situations and have been used for privacy protection [94] [126], controlled evaluation of spatiotemporal algorithms (e.g. clustering or map-matching algorithms) [171] [237] [49] [106] [6], and spatial database evaluation [171] [64] [40] [56] [159]. Synthetic trajectories are often used in such situations because finding suitable, real-world trajectories is difficult due to privacy issues, ethical concerns, dataset size, researcher access and sampling frequency [171] [143] [119] [174].

Because synthetic trajectories are used as substitutes for real world data, it is important that the synthetics accurately reflect properties found in the real world. Such calibration is essential if the trajectories are to be reflective of reality. However, as the related literature review showed (chapter 2), often MOGs (moving object generators) resort to heuristic models for obtaining synthetics, resulting in synthetic trajectories that do not contain key characteristics found in real world datasets. The review also noted that while many human mobility models provide improved models of mobility, they are typically not structured for producing synthetics and involve the specification of a model which may leave out relevant factors.

To address the aforementioned modeling concerns when constructing synthetic vehicle trajectories, we presented a data-driven, microscopic traffic simulator-based framework for producing synthetic vehicle traces (see chapter 3). The use of a microscopic traffic simulator enabled the proposed framework to model the complex nature of vehicle traffic at the individual level according to established transportation theory [70]. Being a data-driven

framework means that the the microscopic traffic simulation demand used to produce the desired synthetics has been calibrated from real-world patterns according to macro- and micro-level procedures. As we aim to make our work practical and usable by practitioners, our framework is implemented in an alpha-release package that enables fully reproducible, data-driven vehicle simulations from the `R` programming environment.

As research is an ongoing process, there is always room for improving upon the proposed framework. One area that could easily be extended would be the micro-level inference procedures. In their current form, the procedures are purely heuristic and incorporate seasonality by grouping related entries when constructing their respective empirical PMFs. If an analyst desires that a more robust model be used for constructing such distributions, such a model can be easily supplied through the framework's built-in support for custom, micro-level profiling procedures.

Another notable area of improvement would be to extend the somewhat limited filtering capabilities supplied by the *synthetics* mode. Obviously, as an academic work, such filtering was primarily introduced to aid in post-processing efforts and therefore, the capabilities introduced were those deemed necessary for such an effort. Of course, such filtering capabilities are unlikely to fit the needs of all analysts. Thus, a better approach would be to allow an analyst to provide a custom filtering procedure (similar to the custom, micro-level profiling procedures) which would appropriately filter the produced synthetics according to the analyst's need.

The path assignment algorithm could also be further improved by taking advantage of the low-level information provided by the reference trajectories. With the exception of the path replication piece, the GeoAware framework relies on SUMO for routing vehicles between onset and terminus locations. Such routing procedures do not make use of the implicit routing information contained in the reference trajectories, thereby missing an opportunity to infer path-level mobility patterns that influence the routing of vehicles. In an effort to mitigate such effects, the proposed framework provides the ability to replicate a

certain percentage of tracks but such a solution is not a substitute for actually inferring transferable, path-level patterns.

While we have noted areas of the framework that could be further improved in later research, the proposed framework has been shown to provide an analyst with an accessible approach for generating realistic, synthetic vehicle traces. Our qualitative and quantitative evaluation (chapter 4) of the framework demonstrated the capabilities of the framework and presented our contributions to the current state of practice. In particular, the utility of the proposed framework was evaluated by demonstrating how easy it is for an analyst to use the created alpha-release package to generate fully reproducible vehicle simulations from within the `R` environment. Such an accomplishment is notable as, to the best of the author's knowledge, no such traffic simulation package exists for the `R` programming environment.

The evaluation also demonstrated the ability of the framework to infer seasonal, macro- and micro-level patterns from the reference trajectories to use when calibrating simulation demand. A collection of quantifiable demand metrics was presented and it was shown that the proposed framework calibrated such metrics from inference procedures on the reference trajectories. Additionally, for each metric, we contrasted the proposed approach with the current literature and techniques employed in practice. Our evaluation shows that the GeoAware framework brings many improvements to the field of MOGs.

As an application-orientated work, we concluded the evaluation with a tutorial-like discussion that demonstrated how to use the GeoAware framework to create synthetics for the Chicago, Illinois area. This tutorial is meant to demonstrate the entire modeling and simulation process and serves as a guide for using the framework.

When taken collectively, our various evaluations demonstrate that the GeoAware framework improves upon current MOGs by providing an analyst accessible, statistically-backed framework for producing the synthetic vehicle traces that are often needed in various research endeavors.

5.1 Acknowledgments

This work was supported by industrial and government membership fees to the Center for Surveillance Research, a National Science Foundation Industry and University Cooperative Research Program (I/UCRC).

Bibliography

[1] Mohamed A Abdel-Aty, Ryuichi Kitamura, and Paul P Jovanis. Using stated preference data for studying the effect of advanced traffic information on drivers' route choice. *Transportation Research Part C: Emerging Technologies*, 5(1):39–50, 1997.

[2] Mark D Abkowitz. An analysis of the commuter departure time decision. *Transportation*, 10(3):283–297, 1981.

[3] Federal Highway Administration and Federal Transit Administration. The transportation planning process briefing book, 2019.

[4] Aimsun SL. *aimsun.next Technical Documentation*.

[5] Hashem R Al-Masaeid and Sanaa S Fayyad. Estimation of trip generation rates for residential areas in jordan. *Jordan Journal of Civil Engineering*, 12(1), 2018.

[6] Abdullah Aldwyish, Egemen Tanin, and Shanika Karunasekera. Follow the best: Crowdsourced automated travel advice. In *Proceedings of the 14th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pages 272–281, 2017.

[7] Vassili Alexiadis, Krista Jeannotte, Andre Chandra, Alexander Skabardonis, and Richard Dowling. Traffic analysis toolbox volume i, traffic analysis tools primer.

Technical report, United States. Federal Highway Administration. Office of Research, Development, and Technology, 2004.

[8] Staffan Algers, Jonas Eliasson, and Lars-Göran Mattsson. Is it time to use activity-based urban transport models? a discussion of planning needs and modelling possibilities. *The Annals of Regional Science*, 39(4):767–789, 2005.

[9] Lourdes MW Almeida and Mirian B Gonçalves. A methodology to incorporate behavioral aspects in trip-distribution models with an application to estimate student flow. *Environment and Planning A*, 33(6):1125–1138, 2001.

[10] Constantinos Antoniou, Mosche Ben-Akiva, Michel Bierlaire, and Rabi Mishalani. Demand simulation for dynamic traffic assignment. *IFAC Proceedings Volumes*, 30(8):633–637, 1997.

[11] Akinori Asahara, Kishiko Maruyama, Akiko Sato, and Kouichi Seto. Pedestrian-movement prediction based on mixed markov-chain model. In *Proceedings of the 19th ACM SIGSPATIAL international conference on advances in geographic information systems*, pages 25–33, 2011.

[12] Kalidas Ashok and Moshe E Ben-Akiva. Alternative approaches for real-time estimation and prediction of time-dependent origin–destination flows. *Transportation science*, 34(1):21–36, 2000.

[13] Berkay Aydin, Rafal Angryk, and Karthik Ganesan Pillai. Ermo-dg: Evolving region moving object dataset generator. In *The Twenty-Seventh International Flairs Conference*, 2014.

[14] Nicolás Badiola, Sebastián Raveau, and Patricia Galilea. Modelling preferences towards activities and their effect on departure time choices. *Transportation Research Part A: Policy and Practice*, 129:39–51, 2019.

[15] Ramachandran Balakrishna. *Off-line calibration of dynamic traffic assignment models*. PhD thesis, Massachusetts Institute of Technology, 2006.

[16] Ramachandran Balakrishna, Constantinos Antoniou, Moshe Ben-Akiva, Haris N Koutsopoulos, and Yang Wen. Calibration of microscopic traffic simulation models: Methods and application. *Transportation Research Record*, 1999(1):198–207, 2007.

[17] Hugo Barbosa, Marc Barthelemy, Gourab Ghoshal, Charlotte R James, Maxime Lenormand, Thomas Louail, Ronaldo Menezes, José J Ramasco, Filippo Simini, and Marcello Tomasini. Human mobility: Models and applications. *Physics Reports*, 734:1–74, 2018.

[18] Hugo Barbosa, Fernando B de Lima-Neto, Alexandre Evsukoff, and Ronaldo Menezes. The effect of recency to human mobility. *EPJ Data Science*, 4:1–14, 2015.

[19] Jaume Barceló. Models, traffic models, simulation, and traffic simulation. In *Fundamentals of traffic simulation*, pages 1–62. Springer, 2010.

[20] Jaume Barceló, Esteve Codina, Jordi Casas, Jaume L Ferrer, and David García. Microscopic traffic simulation: A tool for the design, analysis and evaluation of intelligent transport systems. *Journal of intelligent and robotic systems*, 41(2-3):173–203, 2005.

[21] Jaume Barceló et al. *Fundamentals of traffic simulation*, volume 145. Springer, 2010.

[22] Tim Barmby and Jurgen Doornik. Modelling trip frequency as a poisson variable. *Journal of Transport Economics and Policy*, pages 309–315, 1989.

[23] Bekir Bartin, Kaan Ozbay, Jingqin Gao, and Abdullah Kurkcu. Calibration and validation of large-scale traffic simulation networks: a case study. *Procedia computer science*, 130:844–849, 2018.

[24] Aleix Bassolas, José J. Ramasco, Ricardo Herranz, and Oliva G. Cantú-Ros. Mobile phone records to feed activity-based travel demand models: Matsim for studying a cordon toll policy in barcelona. *Transportation Research Part A: Policy and Practice*, 121:56–74, Mar 2019.

[25] Edward Beimborn and Rob Kennedy. Inside the blackbox: Making transportation models work for livable communities, 1996.

[26] Gabriela Beirão and JA Sarsfield Cabral. Understanding attitudes towards public transport and private car: A qualitative study. *Transport policy*, 14(6):478–489, 2007.

[27] Moshe Ben-Akiba, Hugh F Gunn, and Lionel A Silman. Disaggregate trip distribution models. *Doboku Gakkai Ronbunshu*, 1984(347):1–17, 1984.

[28] Moshe Ben-Akiva, Haris N Koutsopoulos, Constantinos Antoniou, and Ramachandran Balakrishna. Traffic simulation with dynamit. In *Fundamentals of traffic simulation*, pages 363–398. Springer, 2010.

[29] Moshe Ben-Akiva and Steven R. Lerman. *Discrete Choice Analysis: Theory and Application to Travel Demand*. The MIT Press, 1985.

[30] Lisa Benton-Short and Nathaniel M. Lewis. Cities of the united states and canada. In Stanley D. Brunn, Maureen Hays-Mitchell, Donald J. Zeigler, and Jessica K. Graybill, editors, *Cities of the World, Regional Patterns and Urban Environments*. Rowman and Littlefield, 2016.

[31] Chandra R Bhat. Analysis of travel mode and departure time choice for urban shopping trips. *Transportation Research Part B: Methodological*, 32(6):361–371, 1998.

[32] Chandra R Bhat. Incorporating observed and unobserved heterogeneity in urban work travel mode choice modeling. *Transportation science*, 34(2):228–238, 2000.

[33] Chandra R Bhat and Jennifer L Steed. A continuous-time model of departure time choice for urban shopping trips. *Transportation Research Part B: Methodological*, 36(3):207–224, 2002.

[34] Roger S Bivand, Edzer J Pebesma, Virgilio Gómez-Rubio, and Edzer Jan Pebesma. *Applied spatial data analysis with R*, volume 747248717. Springer, 2008.

[35] Transportation Research Board. *Metropolitan travel forecasting: Current practice and future direction — Special Report 288*. The National Academies Press, 2007.

[36] John L Bowman and Moshe E Ben-Akiva. Activity-based disaggregate travel demand model system with activity schedules. *Transportation research part a: policy and practice*, 35(1):1–28, 2001.

[37] John Lawrence Bowman. *Activity based travel demand model system with daily activity schedules*. PhD thesis, Massachusetts Institute of Technology, 1995.

[38] David E Boyce and Huw CWL Williams. Urban travel forecasting in the usa and uk. In *Methods and Models in Transport and Telecommunications*, pages 25–44. Springer, 2005.

[39] Luuk Brederode, Adam Pel, Luc Wismans, Erik de Romph, and Serge Hoogendoorn. Static traffic assignment with queuing: model properties and applications. *Transportmetrica A: Transport Science*, 15(2):179–214, 2019.

[40] Thomas Brinkhoff. A framework for generating network-based moving objects. *GeoInformatica*, 6(2):153–180, 2002.

[41] Dirk Brockmann, Lars Hufnagel, and Theo Geisel. The scaling laws of human travel. *Nature*, 439(7075):462–465, 2006.

[42] Hans-Joachim Bungartz, Stefan Zimmer, Martin Buchholz, and Dirk Pflüger. Microscopic simulation of road traffic. In *Modeling and Simulation*, pages 171–201. Springer, 2014.

[43] Andrew Bwambale, Charisma F Choudhury, and Stephane Hess. Modelling trip generation using mobile phone data: A latent demographics approach. *Journal of Transport Geography*, 76:276–286, 2019.

[44] Ennio Cascetta, Francesca Pagliara, and Andrea Papola. Alternative approaches to trip distribution modelling: a retrospective review and suggestions for combining different approaches. *Papers in regional Science*, 86(4):597–620, 2007.

[45] Joe Castiglione, Mark Bradley, and John Gliebe. *Activity-based travel demand models: a primer*. The National Academies Press, 2015.

[46] Robert Cervero. Built environments and mode choice: toward a normative framework. *Transportation Research Part D: Transport and Environment*, 7(4):265–284, 2002.

[47] Frank J Cesario. A combined trip generation and distribution model. *Transportation Science*, 9(3):211–223, 1975.

[48] Justin S Chang, Dongjae Jung, Jaekyung Kim, and Taeseok Kang. Comparative analysis of trip generation models: results using home-based work trips in the seoul metropolitan area. *Transportation Letters*, 6(2):78–88, 2014.

[49] Bertil Chapuis and Benoît Garbinato. Geodabs: Trajectory indexing meets fingerprinting at scale. In *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, pages 1086–1095. IEEE, 2018.

[50] Cynthia Chen, Jingtao Ma, Yusak Susilo, Yu Liu, and Menglin Wang. The promises of big data and small data for travel behavior (aka human mobility) analysis. *Transportation research part C: emerging technologies*, 68:285–299, 2016.

[51] Jie Chen, Zhu Xiao, Dong Wang, Daiwu Chen, Vincent Havyarimana, Jing Bai, and Hongyang Chen. Toward opportunistic compression and transmission for private car trajectory data collection. *IEEE Sensors Journal*, 19(5):1925–1935, 2018.

[52] Xiqun Michael Chen, Chenfeng Xiong, Xiang He, Zheng Zhu, and Lei Zhang. Time-of-day vehicle mileage fees for congestion mitigation and revenue generation: A simulation-based optimization method and its real-world application. *Transportation Research Part C: Emerging Technologies*, 63:71–95, 2016.

[53] Long Cheng, Xuewu Chen, Jonas De Vos, Xinjun Lai, and Frank Witlox. Applying a random forest method approach to model travel mode choice behavior. *Travel behaviour and society*, 14:1–10, 2019.

[54] Yi-Chang Chiu, Jon Bottom, Michael Mahut, Alexander Paz, Ramachandran Balakrishna, Travis Waller, and Jim Hicks. Dynamic traffic assignment: A primer. *Dynamic Traffic Assignment: A Primer*, 2011.

[55] Hsun-Jung Cho, Yow-Jen Jou, and Chien-Lun Lan. Time dependent origin-destination estimation from traffic count without prior information. *Networks and Spatial Economics*, 9(2):145–170, 2009.

[56] Seongjin Choi, Jiwon Kim, and Hwasoo Yeo. Trajgail: Generating urban trajectories using generative adversarial imitation learning. *arXiv preprint arXiv:2007.14189*, 2020.

[57] Andy HF Chow. Properties of system optimal traffic assignment with departure time choice and its solution method. *Transportation Research Part B: Methodological*, 43(3):325–344, 2009.

[58] RA Cochrane. A possible economic basis for the gravity model. *Journal of Transport Economics and Policy*, pages 34–49, 1975.

[59] Adrian V Cotrus, Joseph N Prashker, and Yoram Shiftan. Spatial and temporal transferability of trip generation demand models in israel. *Journal of Transportation and Statistics*, 8(1):37, 2005.

[60] Andrea Cuttone, Sune Lehmann, and Marta C González. Understanding predictability and exploration in human mobility. *EPJ Data Science*, 7:1–17, 2018.

[61] Takao Dantsuji, Daisuke Fukuda, and Nan Zheng. Simulation-based joint optimization framework for congestion mitigation in multimodal urban network: a macroscopic approach. *Transportation*, pages 1–25, 2019.

[62] Felipe de Souza, Omer Verbas, and Joshua Auld. Mesoscopic traffic flow model for agent-based simulation. *Procedia Computer Science*, 151:858–863, 2019.

[63] Chuan Ding, Donggen Wang, Chao Liu, Yi Zhang, and Jiawen Yang. Exploring the influence of built environment on travel mode choice considering the mediating effects of car ownership and travel distance. *Transportation Research Part A: Policy and Practice*, 100:65–80, 2017.

[64] Christian Düntgen, Thomas Behr, and Ralf Hartmut Güting. Berlinmod: a benchmark for moving object databases. *The VLDB Journal*, 18(6):1335, 2009.

[65] Juan M Durán. What is a simulation model? *Minds and Machines*, pages 1–23, 2020.

[66] James Durbin and Siem Jan Koopman. *Time series analysis by state space methods*. Oxford university press, 2012.

[67] Dirk Eddelbuettel. *Seamless R and C++ Integration with Rcpp*. Springer, New York, 2013. ISBN 978-1-4614-6867-7.

[68] Dirk Eddelbuettel and James Joseph Balamuta. Extending extitR with extitC++: A Brief Introduction to extitRcpp. *PeerJ Preprints*, 5:e3188v1, aug 2017.

[69] Dirk Eddelbuettel and Romain François. Rcpp: Seamless R and C++ integration. *Journal of Statistical Software*, 40(8):1–18, 2011.

[70] Lily Elefteriadou. Simulation modeling. In *An introduction to traffic flow theory*, volume 84. Springer, 2014.

[71] Dick Ettema, Guus Tamminga, Harry Timmermans, and Theo Arentze. A micro-simulation model system of departure time using a perception updating model under travel time uncertainty. *Transportation Research Part A: Policy and Practice*, 39(4):325–344, 2005.

[72] Martin Fellendorf and Peter Vortisch. Microscopic traffic flow simulator vissim. In *Fundamentals of traffic simulation*, pages 63–93. Springer, 2010.

[73] Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, and Depeng Jin. Deepmove: Predicting human mobility with attentional recurrent networks. In *Proceedings of the 2018 world wide web conference*, pages 1459–1468, 2018.

[74] Jie Feng, Zeyu Yang, Fengli Xu, Haisu Yu, Mudan Wang, and Yong Li. Learning to simulate human mobility. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3426–3433, 2020.

[75] Gunnar Flötteröd, Michel Bierlaire, and Kai Nagel. Bayesian demand calibration for dynamic traffic simulations. *Transportation Science*, 45(4):541–561, 2011.

[76] Peter Foytik, Craig Jordan, and R Michael Robinson. Exploring simulation based dynamic traffic assignment with a large-scale microscopic traffic simulation model. In *Proceedings of the 50th Annual Simulation Symposium*, pages 1–12, 2017.

[77] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núnez del Prado Cortez. Towards temporal mobility markov chains. In *Proceedings of the first international workshop on dynamicity collecated with OPODIS 2011*, 2011.

[78] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. Next place prediction using mobility markov chains. In *Proceedings of the first workshop on measurement, privacy, and mobility*, pages 1–6, 2012.

[79] Christian Gawron. An iterative algorithm to determine the dynamic user equilibrium in a traffic simulation model. *International Journal of Modern Physics C*, 9(03):393–407, 1998.

[80] Christian Gawron. *Simulation-Based Traffic Assignment. Computing user equilibria in large street networks*. PhD thesis, Universität zu Köln, 1998.

[81] Judith L. Gersting. *Mathematical Structures for Computer Science*. W.H. Freeman and Company, 2014.

[82] Fosca Giannotti, Andrea Mazzoni, Simone Puntoni, and Chiara Renso. Synthetic generation of cellular network positioning data. In *Proceedings of the 13th annual ACM international workshop on Geographic information systems*, pages 12–20, 2005.

[83] Fosca Giannotti, Mirco Nanni, Fabio Pinelli, and Dino Pedreschi. Trajectory pattern mining. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 330–339, 2007.

[84] Gyozo Gidofalvi and Torben Bach Pedersen. St–acts: a spatio-temporal activity simulator. In *Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems*, pages 155–162, 2006.

[85] Anna Goldenberg, Alice X Zheng, Stephen E Fienberg, and Edoardo M Airoldi. A survey of statistical network models. *arXiv preprint arXiv:0912.5410*, 2009.

[86] Marta C Gonzalez, Cesar A Hidalgo, and Albert-Laszlo Barabasi. Understanding individual human mobility patterns. *nature*, 453(7196):779–782, 2008.

[87] Gabriel Goulet-Langlois, Haris N Koutsopoulos, Zhan Zhao, and Jinhua Zhao. Measuring regularity of individual travel patterns. *IEEE Transactions on Intelligent Transportation Systems*, 19(5):1583–1592, 2017.

[88] Ellen F Grumert, Andreas Tapani, and Xiaoliang Ma. Characteristics of variable speed limit systems. *European transport research review*, 10(2):1–12, 2018.

[89] Hugh Gunn. The netherlands national model: a review of seven years of application. *International Transactions in Operational Research*, 1(2):125–133, 1994.

[90] Yuntao Guo, Jian Wang, Srinivas Peeta, and Panagiotis Ch Anastasopoulos. Impacts of internal migration, household registration system, and family planning policy on travel mode choice in china. *Travel Behaviour and Society*, 13:128–143, 2018.

[91] Alex Hagen-Zanker and Ying Jin. Adaptive zoning for efficient transport modelling in urban models. In *International Conference on Computational Science and Its Applications*, pages 673–687. Springer, 2015.

[92] Torsten Hägerstraand. What about people in regional science? *Papers in regional science*, 24(1):7–24, 1970.

[93] Andrew C Harvey. *Forecasting, structural time series models and the Kalman filter*. Cambridge university press, 1990.

[94] Xi He, Graham Cormode, Ashwin Machanavajjhala, Cecilia M Procopiuc, and Divesh Srivastava. Dpt: differentially private trajectory synthesis using hierarchical reference systems. *Proceedings of the VLDB Endowment*, 8(11):1154–1165, 2015.

[95] Rainer Hegselmann, Ulrich Mueller, and Klaus G Troitzsch. *Modelling and simulation in the social sciences from the philosophy of science point of view*, volume 23. Springer Science & Business Media, 1996.

[96] Dirk Helbing, Ansgar Hennecke, Vladimir Shvetsov, and Martin Treiber. Micro- and macro-simulation of freeway traffic. *Mathematical and computer modelling*, 35(5-6):517–547, 2002.

[97] Leonhard Held and Daniel Sabanés Bové. *Bayesian Inference*, pages 167–219. Springer Berlin Heidelberg, Berlin, Heidelberg, 2020.

[98] Jouni Helske. Kfas: Exponential family state space models in r. *Journal of Statistical Software, Articles*, 78(10):1–39, 2017.

[99] Dwayne Henclewood, Wonho Suh, Michael O Rodgers, Richard Fujimoto, and Michael P Hunter. A calibration procedure for increasing the accuracy of microscopic traffic simulation models. *Simulation*, 93(1):35–47, 2017.

[100] Martin Hilliges and Wolfgang Weidlich. A phenomenological model for dynamic traffic flow in networks. *Transportation Research Part B: Methodological*, 29(6):407–431, 1995.

[101] Yaron Hollander and Ronghui Liu. The principles of calibrating traffic microsimulation models. *Transportation*, 35(3):347–362, 2008.

[102] Haibo Hu and Dik-Lun Lee. Gamma: a framework for moving object simulation. In *International Symposium on Spatial and Temporal Databases*, pages 37–54. Springer, 2005.

[103] Xianbiao Hu and Yi-Chang Chiu. A new approach to calibrating time-dependent origin-destination departure profile for traffic simulation models. In *2011 14th In-*

ternational IEEE Conference on Intelligent Transportation Systems (ITSC), pages 139–144. IEEE, 2011.

[104] Chuanlin Huang, Peiquan Jin, Huaishuai Wang, Na Wang, Shouhong Wan, and Lihua Yue. Indoorstg: A flexible tool to generate trajectory data for indoor moving objects. In *2013 IEEE 14th International Conference on Mobile Data Management*, volume 1, pages 341–343. IEEE, 2013.

[105] Dou Huang, Xuan Song, Zipei Fan, Renhe Jiang, Ryosuke Shibasaki, Yu Zhang, Haizhong Wang, and Yugo Kato. A variational autoencoder based generative model of urban human mobility. In *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 425–430. IEEE, 2019.

[106] Chih-Chieh Hung, Wen-Chih Peng, and Wang-Chien Lee. Clustering and aggregating clues of trajectories for mining trajectory patterns and routes. *The VLDB Journal*, 24(2):169–192, 2015.

[107] Leta F Huntsinger, Nagui M Rouphail, and Peter Bloomfield. Trip generation models using cumulative logistic regression. *Journal of urban planning and development*, 139(3):176–184, 2013.

[108] Sibren Isaacman, Richard Becker, Ramón Cáceres, Margaret Martonosi, James Rowland, Alexander Varshavsky, and Walter Willinger. Human mobility modeling at metropolitan scales. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pages 239–252, 2012.

[109] Mansoureh Jeihani and Anam Ardeshiri. Transims implementation for a small network and comparison with enhanced four-step model. *Journal of the Transportation Research Forum*, 53(1), 2014.

[110] Mithilesh Jha, Katie Moore, and Behruz Pashaie. Emergency evacuation planning with microscopic traffic simulation. *Transportation Research Record*, 1886(1):40–48, 2004.

[111] Shan Jiang, Yingxiang Yang, Siddharth Gupta, Daniele Veneziano, Shounak Athavale, and Marta C González. The timegeo modeling framework for urban mobility without travel surveys. *Proceedings of the National Academy of Sciences*, 113(37):E5370–E5378, 2016.

[112] Peter Jones. The role of an evolving paradigm in shaping international transport research and policy agendas over the last 50 years. *Travel behaviour research in an evolving world*, 2009.

[113] Nuriye Kabakuş and Ahmet Tortum. Comparative analysis of trip generation models according to household characteristics for developed, developing and non-developed provinces in turkey. *Sādhanā*, 44(5):122, 2019.

[114] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the AMSE–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[115] Islam R Kamel, Hossam Abdelgawad, and Baher Abdulhai. Transportation big data simulation platform for the greater toronto area (gta). In *Smart City 360*, pages 443–454. Springer, 2016.

[116] Juyoung Kang and Hwan-Seung Yong. Spatio-temporal discretization for sequential pattern mining. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication*, pages 218–224, 2008.

[117] Steven M Kay. *Fundamentals of statistical signal processing*. Prentice Hall PTR, 1993.

[118] McKenzie Keehan. Annual average daily traffic (aadt) estimation with regression using centrality and roadway characteristic variables. *Master's Thesis*, 2017.

[119] Joon-Seok Kim, Hamdi Kavak, Umar Manzoor, Andrew Crooks, Dieter Pfoser, Carola Wenk, and Andreas Züfle. Simulating urban patterns of life: A geo-social data generation framework. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 576–579, 2019.

[120] Kyu-Ok Kim and LR Rilett. Simplex-based calibration of traffic microsimulation models with intelligent transportation systems data. *Transportation Research Record*, 1855(1):80–89, 2003.

[121] Justin Kitzes, Daniel Turek, and Fatma Deniz. *The practice of reproducible research: case studies and lessons from the data-intensive sciences*. Univ of California Press, 2017.

[122] Georgios N Kouziokas. Neural network-based road accident forecasting in transportation and public management. In *The 4th Conference on Sustainable Urban Mobility*, pages 98–103. Springer, 2018.

[123] Daniel Krajzewicz. Traffic simulation with sumo–simulation of urban mobility. In *Fundamentals of traffic simulation*, pages 269–293. Springer, 2010.

[124] Daniel Krajzewicz, Michael Behrisch, and Y Wang. Comparing performance and quality of traffic assignment techniques for microscopic road traffic simulations. In *Proc. DTA2008 International Symposium on Dynamic Traffic Assignment*, 2008.

[125] Ida Kristoffersson and Leonid Engelson. Estimating preferred departure times of road users in a large urban network. *Transportation*, 45(3):767–787, 2018.

[126] Vaibhav Kulkarni and Benoît Garbinato. Generating synthetic mobility traffic using rnns. In *Proceedings of the 1st Workshop on Artificial Intelligence and Deep Learning for Geographic Knowledge Discovery*, pages 1–4, 2017.

[127] Vaibhav Kulkarni, Abhijit Mahalunkar, Benoit Garbinato, and John D Kelleher. On the inability of markov models to capture criticality in human mobility. In *International Conference on Artificial Neural Networks*, pages 484–497. Springer, 2019.

[128] Vaibhav Kulkarni, Natasa Tagasovska, Thibault Vatter, and Benoit Garbinato. Generative models for simulating mobility trajectories. *arXiv preprint arXiv:1811.12801*, 2018.

[129] Ajay Kumar and David Matthew Levinson. Specifying, estimating and validating a new trip generation model: Case study in montgomery county, maryland. *Transportation Research Record*, pages 107–113, 1994.

[130] Masao Kuwahara, Ryota Horiguchi, and Hisatomo Hanabusa. Traffic simulation with avenue. In *Fundamentals of Traffic Simulation*, pages 95–129. Springer, 2010.

[131] William HK Lam and Hai-Jun Huang. Combined activity/travel choice models: time-dependent and dynamic versions. *Networks and Spatial Economics*, 3(3):323–347, 2003.

[132] Larry J LeBlanc, Edward K Morlok, and William P Pierskalla. An efficient approach to solving the road network equilibrium traffic assignment problem. *Transportation research*, 9(5):309–318, 1975.

[133] Ludovic Leclercq, Alméria Sénécat, and Guilhem Mariotte. Dynamic macroscopic simulation of on-street parking search: A trip-based approach. *Transportation Research Part B: Methodological*, 101:268–282, 2017.

[134] Baibing Li. Bayesian inference for origin-destination matrices of transport networks using the em algorithm. *Technometrics*, 47(4):399–408, 2005.

[135] Huan Li, Hua Lu, Xin Chen, Gang Chen, Ke Chen, and Lidan Shou. Vita: A versatile toolkit for generating indoor mobility data for real-world buildings. *Proceedings of the VLDB Endowment*, 9(13):1453–1456, 2016.

[136] Ben Liang and Zygmunt J Haas. Predictive distance-based mobility management for pcs networks. In *IEEE INFOCOM'99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No. 99CH36320)*, volume 3, pages 1377–1384. IEEE, 1999.

[137] Edward B Lieberman. Brief history of traffic simulation. *Traffic and Transportation Simulation*, 17, 2014.

[138] Dung-Ying Lin, Naveen Eluru, S Travis Waller, and Chandra R Bhat. Evacuation planning using the integrated system of activity-based modeling and dynamic traffic assignment. *Transportation research record*, 2132(1):69–77, 2009.

[139] Ronghui Liu, Dirck Van Vliet, and David Watling. Microsimulation models incorporating both demand and supply dynamics. *Transportation Research Part A: Policy and Practice*, 40(2):125–150, 2006.

[140] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie WieBner. Microscopic traffic simulation using sumo. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2575–2582. IEEE, 2018.

[141] Robin Lovelace, Jakub Nowosad, and Jannes Muenchow. *Geocomputation with R*. CRC Press, 2019.

[142] Xin Lu, Erik Wetter, Nita Bharti, Andrew J Tatem, and Linus Bengtsson. Approaching the limit of predictability in human mobility. *Scientific reports*, 3:2923, 2013.

[143] Yongmei Lu and Yu Liu. Pervasive location acquisition technologies: Opportunities and challenges for geospatial studies. *Computers, Environment and Urban Systems*, 36(2):105–108, 2012.

[144] Eleanor Mann and Charles Abraham. The role of affect in uk commuters' travel mode choices: An interpretative phenomenological analysis. *British journal of psychology*, 97(2):155–176, 2006.

[145] WW Mann. B-node model: New subarea traffic assignment model & application. In *Eighth TRB Conference on the Application of Transportation Planning MethodsTransportation Research Board; Texas Department of Transportation; Corpus Christi Metropolitan Planning Organization; Federal Highway Administration; and Federal Transit Administration.*, 2002.

[146] Norman L Marshall. Forecasting the impossible: The status quo of estimating traffic flows with static traffic assignment and the future of dynamic traffic assignment. *Research in Transportation Business & Management*, 29:85–92, 2018.

[147] Brian V Martin, Frederick W Memmott, and Alexander J Bone. Principles and techniques of predicting future demand for urban area transportation. *Mass Inst Tech Dept Pub Wks Jt Hwy Res*, 1961.

[148] Catherine Matias and Stéphane Robin. Modeling heterogeneity in random graphs through latent space models: a selective review. *ESAIM: Proceedings and Surveys*, 47:55–74, 2014.

[149] Michael G. McNally. The activity-based approach. Technical report, UC Irvine: Center for Activity Systems Analysis, 2000.

[150] Bob McQueen. *Big Data Analytics for Connected Vehicles and Smart Cities*. Artech House, 2017.

[151] Mohamed F Mokbel, Louai Alarabi, Jie Bao, Ahmed Eldawy, Amr Magdy, Mohamed Sarwat, Ethan Waytas, and Steven Yackel. Mntg: An extensible web-based traffic generator. In *International Symposium on Spatial and Temporal Databases*, pages 38–55. Springer, 2013.

[152] Elliott W Montroll and George H Weiss. Random walks on lattices. ii. *Journal of Mathematical Physics*, 6(2):167–181, 1965.

[153] Jameson Morgan and Derek Doran. Geonet: A framework for intrinsic geospatial anomaly detection. Poster, 2019.

[154] Sandeep Mudigonda and Kaan Ozbay. Robust calibration of macroscopic traffic simulation models using stochastic collocation. *Transportation Research Procedia*, 9:1–20, 2015.

[155] Fred Nwanganga and Mike Chapple. *Naïve Bayes*, chapter 7, pages 251–275. John Wiley & Sons, Ltd, 2020.

[156] City of Chicago. Taxi trips. Online, 2020 (accessed 5/20/2020). https://data.cityofchicago.org/Transportation/Taxi-Trips/wrvz-psew.

[157] Eduardo Mucelli Rezende Oliveira, Aline Carneiro Viana, Carlos Sarraute, Jorge Brea, and Ignacio Alvarez-Hamelin. On the regularity of human mobility. *Pervasive and Mobile Computing*, 33:73–90, 2016.

[158] Stan Openshaw. Ecological fallacies and the analysis of areal census data. *Environment and planning A*, 16(1):17–31, 1984.

[159] Kun Ouyang, Reza Shokri, David S Rosenblum, and Wenzhuo Yang. A non-parametric generative model for human trajectories. In *IJCAI*, pages 3812–3817, 2018.

[160] Markos Papageorgiou, Ioannis Papamichail, Albert Messmer, and Yibing Wang. Traffic simulation with metanet. In *Fundamentals of traffic simulation*, pages 399–430. Springer, 2010.

[161] Luca Pappalardo, Salvatore Rinzivillo, and Filippo Simini. Human mobility modelling: exploration and preferential return meet the gravity model. *Procedia Computer Science*, 83:934–939, 2016.

[162] Luca Pappalardo and Filippo Simini. Data-driven generation of spatio-temporal routines in human mobility. *Data Mining and Knowledge Discovery*, 32(3):787–829, 2018.

[163] Luca Pappalardo, Filippo Simini, Gianni Barlacchi, and Roberto Pellungrini. scikit-mobility: a python library for the analysis, generation and risk assessment of mobility data. *arXiv preprint arXiv:1907.07062*, 2019.

[164] Luca Pappalardo, Filippo Simini, Salvatore Rinzivillo, Dino Pedreschi, Fosca Giannotti, and Albert-László Barabási. Returners and explorers dichotomy in human mobility. *Nature communications*, 6(1):1–8, 2015.

[165] Byungkyu Park, Hongtu Qi, et al. Development and evaluation of a calibration and validation procedure for microscopic simulation models. Technical report, Virginia Transportation Research Council, 2004.

[166] Byungkyu "Brian" Park, Nagui M Rouphail, and Jerome Sacks. Assessment of stochastic signal optimization method using microsimulation. *Transportation Research Record*, 1748(1):40–45, 2001.

[167] Alexander Paz, Victor Molano, and Javier Sanchez-Medina. Holistic calibration of microscopic traffic flow models: Methodology and real world application studies. In *Engineering and applied sciences optimization*, pages 33–52. Springer, 2015.

[168] Stefanie Peer, Jasper Knockaert, Paul Koster, Yin-Yen Tseng, and Erik T Verhoef. Door-to-door travel times in rp departure time choice models: An approximation method using gps data. *Transportation Research Part B: Methodological*, 58:134–150, 2013.

[169] Srinivas Peeta and Athanasios K Ziliaskopoulos. Foundations of dynamic traffic assignment: The past, the present and the future. *Networks and spatial economics*, 1(3-4):233–265, 2001.

[170] Nikos Pelekis, Christos Ntrigkogias, Panagiotis Tampakis, Stylianos Sideridis, and Yannis Theodoridis. Hermoupolis: a trajectory generator for simulating generalized mobility patterns. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 659–662. Springer, 2013.

[171] Nikos Pelekis and Yannis Theodoridis. Modeling and acquiring mobility data. In *Mobility data management and exploration*, pages 51–73. Springer, 2014.

[172] Ram M Pendyala and Chandra R Bhat. Validation and assessment of activity-based travel demand modeling systems. In *Innovations in Travel Demand Modeling*, volume 2, pages 157–160. The National Academies Press, 2008.

[173] Dieter Pfoser and Yannis Theodoridis. Generating semantics-based trajectories of moving objects. *Computers, Environment and Urban Systems*, 27(3):243–263, 2003.

[174] Matthew Piekenbrock and Derek Doran. Intrinsic point of interest discovery from trajectory data. *arXiv preprint arXiv:1712.05247*, 2017.

[175] PTV Group. *VISSIM*.

[176] Xiaobo Qu, Lu Zhen, Robert J Howlett, and Lakhmi C Jain. *Smart Transportation Systems 2019*, volume 149. Springer, 2019.

[177] David A Quarmby. Choice of travel mode for the journey to work: some findings. *Journal of transport Economics and Policy*, pages 273–314, 1967.

[178] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2020.

[179] Hossein Rahimi-Farahani, Amir Abbas Rassafi, and Babak Mirbaha. Forced-node route guidance system: incorporating both user equilibrium and system optimal benefits. *IET Intelligent Transport Systems*, 13(12):1851–1859, 2019.

[180] Anshika Rawal and Pravati Swain. Prediction of human mobility using mobile traffic dataset with hmm. In *Recent Findings in Intelligent Computing Techniques*, pages 489–495. Springer, 2018.

[181] Wilfred W Recker, Michael G McNally, and Gregory S Root. A model of complex travel behavior: Part ii—an operational model. *Transportation Research Part A: General*, 20(4):319–330, 1986.

[182] Wilfred W Recker, Michael G McNally, and GS Root. A model of complex travel behavior: Part i—theoretical development. *Transportation Research Part A: General*, 20(4):307–318, 1986.

[183] Jace Robinson and Derek Doran. Seasonal stochastic blockmodeling for anomaly detection in dynamic networks. *arXiv preprint arXiv:1712.05359*, 2017.

[184] Jace D Robinson. A model for seasonal dynamic networks. *Master's Thesis*, 2018.

[185] Thomas F Rossi and Yoram Shiftan. Tour based travel demand modeling in the us. *IFAC Proceedings Volumes*, 30(8):381–386, 1997.

[186] Earl R Ruiter and Moshe Ben-Akiva. Disaggregate travel demand models for the san francisco bay area. system structure, component models, and application procedures. *Transportation Research Record*, (673):121–128, 1978.

[187] Jean-Marc Saglio and José Moreira. Oporto: A realistic scenario generator for moving objects. *GeoInformatica*, 5(1):71–93, 2001.

[188] Krishna Saw, Bhimaji K Katti, and Gaurang K Joshi. Ab temporal trip generation modelling for primary activities: A case study of fast growing metropolitan city. *Population (lacs)*, 1971:1981, 1961.

[189] Joachim Scheiner and Christian Holz-Rau. Gendered travel mode choice: a focus on car deficient households. *Journal of Transport Geography*, 24:250–261, 2012.

[190] ECML PKDD 2015 Data Set. Taxi service trajectory - prediction challenge. Online, 2015 (accessed 6/23/2020). http://www.geolink.pt/ecmlpkdd2015-challenge/dataset.html.

[191] Sajjad Shafiei, Ziyuan Gu, and Meead Saberi. Calibration and validation of a simulation-based dynamic traffic assignment model for a large-scale congested network. *Simulation Modelling Practice and Theory*, 86:169–186, 2018.

[192] Alec T Shuldiner and Paul W Shuldiner. The measure of all things: reflections on changing conceptions of the individual in travel demand modeling. *Transportation*, 40(6):1117–1131, 2013.

[193] Anja Simma and Kay W Axhausen. Structures of commitment in mode use: a comparison of switzerland, germany and great britain. *Transport Policy*, 8(4):279–288, 2001.

[194] Adriano F Siqueira, Carlos JT Peixoto, Chen Wu, and Wei-Liang Qian. Effect of stochastic transition in the fundamental diagram of traffic flow. *Transportation Research Part B: Methodological*, 87:1–13, 2016.

[195] Virginia P Sisiopiku. Application of traffic simulation modeling for improved emergency preparedness planning. *Journal of urban planning and development*, 133(1):51–60, 2007.

[196] Jaimison Sloboden, John Lewis, Vassili Alexiadis, Yi-Chang Chiu, Eric Nava, et al. Traffic analysis toolbox volume xiv: Guidebook on the utilization of dynamic traffic assignment in modeling. Technical report, United States. Federal Highway Administration. Office of Operations, 2012.

[197] Kenneth A Small. The scheduling of consumer activities: work trips. *The American Economic Review*, 72(3):467–479, 1982.

[198] Chaoming Song, Tal Koren, Pu Wang, and Albert-László Barabási. Modelling the scaling properties of human mobility. *Nature Physics*, 6(10):818–823, 2010.

[199] Chaoming Song, Zehui Qu, Nicholas Blumm, and Albert-László Barabási. Limits of predictability in human mobility. *Science*, 327(5968):1018–1021, 2010.

[200] Hyewon Song and Okgee Min. Statistical traffic generation methods for urban traffic simulation. In *2018 20th International Conference on Advanced Communication Technology (ICACT)*, pages 247–250. IEEE, 2018.

[201] Xuan Song, Hiroshi Kanasugi, and Ryosuke Shibasaki. Deeptransport: Prediction and simulation of human mobility and transportation mode at a citywide level. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2618–2624, 2016.

[202] Peter R Stopher and KATHIE G McDONALD. Trip generation by cross-classification: an alternative methodology. *Transportation Research Record*, 944:84–91, 1983.

[203] Samuel A Stouffer. Intervening opportunities and competing migrants. *Journal of regional science*, 2(1):1–26, 1960.

[204] Matthias N Sweet and Mengke Chen. Does regional travel time unreliability influence mode choice? *Transportation*, 38(4):625–642, 2011.

[205] Pete Sykes. Traffic simulation with paramics. In *Fundamentals of traffic simulation*, pages 131–171. Springer, 2010.

[206] Cambridge Systematics. *Travel Demand Forecasting: Parameters and Techniques*, volume 716. Transportation Research Board, 2012.

[207] Nursitihazlin Ahmad Termida, Yusak O Susilo, Joel P Franklin, and Chengxi Liu. Understanding seasonal variation in individual's activity participation and trip generation by using four consecutive two-week travel diary. *Travel Behaviour and Society*, 12:52–63, 2018.

[208] Yannis Theodoridis, Jefferson RO Silva, and Mario A Nascimento. On the generation of spatiotemporal datasets. In *International Symposium on Spatial Databases*, pages 147–164. Springer, 1999.

[209] Guang Tian, Keunhyun Park, and Reid Ewing. Trip and parking generation rates for different housing types: Effects of compact development. *Urban Studies*, 56(8):1554–1575, 2019.

[210] Tomer Toledo, Moshe E Ben-Akiva, Deepak Darda, Mithilesh Jha, and Haris N Koutsopoulos. Calibration of microscopic traffic simulation models with aggregate data. *Transportation Research Record*, 1876(1):10–19, 2004.

[211] Tomer Toledo, Haris N Koutsopoulos, Angus Davol, Moshe E Ben-Akiva, Wilco Burghout, Ingmar Andréasson, Tobias Johansson, and Christen Lundin. Calibration and validation of microscopic traffic simulation tools: Stockholm case study. *Transportation Research Record*, 1831(1):65–75, 2003.

[212] JA Tomlin. A mathematical programming model for the combined distribution-assignment of traffic. *Transportation Science*, 5(2):122–140, 1971.

[213] Danique Ton, Dorine C Duives, Oded Cats, Sascha Hoogendoorn-Lanser, and Serge P Hoogendoorn. Cycling or walking? determinants of mode choice in the netherlands. *Transportation research part A: policy and practice*, 123:7–23, 2019.

[214] Jameson L Toole, Carlos Herrera-Yaqüe, Christian M Schneider, and Marta C González. Coupling human mobility and social ties. *Journal of The Royal Society Interface*, 12(105):20141128, 2015.

[215] Theodoros Tzouramanis, Michael Vassilakopoulos, and Yannis Manolopoulos. On the generation of time-evolving regional data. *GeoInformatica*, 6(3):207–231, 2002.

[216] Oskar Blom Västberg, Anders Karlström, Daniel Jonsson, and Marcus Sundberg. A dynamic discrete choice activity-based travel demand model. *Transportation science*, 54(1):21–41, 2020.

[217] William S Vickrey. Congestion theory and transport investment. *The American Economic Review*, 59(2):251–260, 1969.

[218] Jose Osiris Vidana-Bencomo, Esmaeil Balal, Jason C Anderson, and Salvador Hernandez. Modeling route choice criteria from home to major streets: A discrete choice approach. *International journal of transportation science and technology*, 7(1):74–88, 2018.

[219] Akshay Vij, Andre Carrel, and Joan L Walker. Incorporating the influence of latent modal preferences on travel mode choice behavior. *Transportation Research Part A: Policy and Practice*, 54:164–178, 2013.

[220] Kay W Axhausen, Andreas Horni, and Kai Nagel. *The multi-agent transport simulation MATSim*. Ubiquity Press, 2016.

[221] Jinzhong Wang, Xiangjie Kong, Feng Xia, and Lijun Sun. Urban human mobility: Data-driven modeling and prediction. *ACM SIGKDD Explorations Newsletter*, 21(1):1–19, 2019.

[222] Hadley Wickham. *Advanced r*. CRC press, 2019.

[223] Michael J Wills. A flexible gravity-opportunities model for trip distribution. *Transportation Research Part B: Methodological*, 20(2):89–111, 1986.

[224] Alan Wilson. Entropy in urban and regional modelling: Retrospect and prospect. *Geographical Analysis*, 42(4):364–394, 2010.

[225] Alan Geoffrey Wilson. The use of entropy maximising models, in the theory of trip distribution, mode split and route split. *Journal of transport economics and policy*, pages 108–126, 1969.

[226] Mintesnot G Woldeamanuel. *Concepts in urban transportation planning: the quest for mobility, sustainability and quality of life*. McFarland, 2016.

[227] HJ Wootton and GW Pick. A model for trips generated by households. *Journal of Transport Economics and Policy*, pages 137–153, 1967.

[228] Hairuo Xie, Egemen Tanin, Kotagiri Ramamohanarao, Shanika Karunasekera, Lars Kulik, Rui Zhang, and Jianzhong Qi. Generating traffic data for any city using smarts simulator. *SIGSPATIAL Special*, 11(1):22–28, 2019.

[229] Chenfeng Xiong, Zheng Zhu, Xiang He, Xiqun Chen, Shanjiang Zhu, Subrat Mahapatra, Gang-Len Chang, and Lei Zhang. Developing a 24-hour large-scale microscopic traffic simulation model for the before-and-after study of a new tolled freeway in the washington, dc–baltimore region. *Journal of Transportation Engineering*, 141(6):05015001, 2015.

[230] Jianqiu Xu and Ralf Hartmut Güting. Mwgen: a mini world generator. In *2012 IEEE 13th International Conference on Mobile Data Management*, pages 258–267. IEEE, 2012.

[231] Can Yang and Gyozo Gidofalvi. Fast map matching, an algorithm integrating hidden markov model with precomputation. *International Journal of Geographical Information Science*, 32(3):547–570, 2018.

[232] Liya Yang, Guo Zheng, and Xiaoning Zhu. Cross-nested logit model for the joint choice of residential location, travel mode, and departure time. *Habitat International*, 38:157–166, 2013.

[233] Zhenzhen Yao, Guijuan Zhang, Dianjie Lu, and Hong Liu. Data-driven crowd evacuation: A reinforcement learning method. *Neurocomputing*, 366:314–327, 2019.

[234] Jia Yu, Zishan Fu, and Mohamed Sarwat. Dissecting geosparksim: a scalable microscopic road network traffic simulator in apache spark. *Distributed and Parallel Databases*, 38(4):963–994, 2020.

[235] Miao Yu and Wei David Fan. Calibration of microscopic traffic simulation models using metaheuristic algorithms. *International Journal of Transportation Science and Technology*, 6(1):63–77, 2017.

[236] Yunjie Zhao and Adel W Sadek. Large-scale agent-based traffic micro-simulation: Experiences with model refinement, calibration, validation and application. *Procedia Computer Science*, 10:815–820, 2012.

[237] Yu Zheng. Trajectory data mining: an overview. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 6(3):1–41, 2015.

[238] Athanasios K Ziliaskopoulos, S Travis Waller, Yue Li, and Mark Byram. Large-scale dynamic traffic assignment: Implementation issues and computational analysis. *Journal of Transportation Engineering*, 130(5):585–593, 2004.